Joint Optimization Projection Matrix and Sparsifying Dictionary via Stochastic Method

Tao Hong

Abstract—An efficient algorithm is proposed in this letter to optimize the Projection Matrix and Sparsifying Dictionary (PMSD) simultaneously on a large training dataset through stochastic method. A closed-from solution is derived for optimizing the projection matrix with a fixed sparsifying dictionary and the stochastic method is used to optimize the sparsifying dictionary with a fixed optimized projection matrix on a large training dataset. Benefiting from training on a large dataset, the proposed method yields a much better performance in terms of signal recovery accuracy than the existing ones. The simulation results on natural images demonstrate its effectiveness and efficiency.

Index Terms—Compressive sensing, stochastic method, projection matrix, sparsifying dictionary, large dataset

I. INTRODUCTION

SPARSE representations of signals have been receiving a lot of attention because of its success applications in image processing, machine learning, pattern recognition, signal detection/classification and compressive sensing (CS) [1], [2] etc. A given signal \( x \in \mathbb{R}^{N \times 1} \) can be represented as a combination of a few atoms of a dictionary \( \Psi \in \mathbb{R}^{N \times L} \) called sparse, i.e.,

\[ x = \Psi \theta + e \]  

(1)

where \( \theta \in \mathbb{R}^{L \times 1} \) is the sparse coefficient vector of \( x \) in \( \Psi \) and \( e \in \mathbb{R}^{N \times 1} \) stands for the sparse representation error (SRE) which is not nil in practical applications but the energy is always small enough. The signal \( x \) is called \( K \)-sparse, i.e., \( \| \theta \|_0 = K \).

The signal recovery methods involve two stages: optimization of the projection matrix \( \Phi \) with the dictionary \( \Psi \) fixed and then go to optimize the dictionary \( \Psi \) with the projection matrix \( \Phi \) fixed. However, their solution in every stage is a local minima. In [13], Bai et al. gave out a closed-form solution for every stage, so their method yields a better promising performance in terms of signal recovery accuracy than the approach given in [11]. However, their method involves too many Singular Value Decompositions (SVDs) which sacrifices the efficiency of the algorithm.

CS is an emerging framework that one can exactly recover the signal \( x \), if it is sparse or sparse under the dictionary \( \Psi \), from a number of linear projections of dimension considerably lower than the number of samples required by the Shannon-Nyquist Theorem [6]. In the original CS theory, people tend to utilize the random matrix as the projection matrix to sample the signal. However, it already knows that an optimized projection matrix can outperform the random one in various cases [7] - [10].

This suggests the proposal of joint optimization projection matrices and dictionary designs in the literatures [11] - [13]. Durate-Carvajalino et al. involved the projection matrix in the dictionary updating procedure by addressing the following problem:

\[
\min_{\Psi \in C, \Phi} \gamma \| X - \Psi \Theta \|_F^2 + \| Y - \Phi \Psi \Theta \|_F^2, \quad \text{s.t.} \| \Theta_k \|_0 \leq K, \forall k
\]  

(3)

where \( \Phi \in \mathbb{R}^{M \times N} \) denotes the projection matrix with \( M \ll N \), \( \gamma \in [0, 1] \) is a trade-off parameter to balance the first term in relation to the second term and \( Y(:,k) = y_k \), \( \forall k \) is a matrix that contains the projector vectors associated with the training signals \( X \), which are related as follow:

\[
y_k \triangleq \Phi x_k = \Phi \Psi \Theta_k + \Phi e_k
\]  

(4)

where \( \Phi e_k \) represents the projection noise caused by SRE.

In their proposed method, they involve iterations between two stages: optimization of the projection matrix \( \Phi \) with the dictionary \( \Psi \) fixed and then go to optimize the dictionary \( \Psi \) with the projection matrix \( \Phi \) fixed. However, their solution in every stage is a local minima. In [13], Bai et al. gave out a closed-form solution for every stage, so their method yields a better promising performance in terms of signal recovery accuracy than the approach given in [11]. However, their method involves too many Singular Value Decompositions (SVDs) which sacrifices the efficiency of the algorithm.

Although the above methods work well in some practical applications, their approaches are hard to extend to the scenario that the size of the training dataset is very large, exceeding \( 10^6 \) patches in natural image situation, or dynamic, e.g., video stream. Due to the results shown in [10], training the dictionary on a large dataset can fit the signal better and yields a better projection matrix which can improve the signal recovery accuracy comparing with designing the projection matrix with a dictionary training on a small dataset. So it is expected that joint optimization the Projection Matrix and Sparsifying Dictionary (PMSD) on a large dataset would receive a better result in terms of signal recovery accuracy. Moreover, developing such an algorithm can let us to optimize...
the PMSD on dynamic data which is more suitable in some video applications. Inspired by the results shown \[14, 15\], an online algorithm is derived in this letter to address the above problems and receives a much better promising performance in terms of signal recovery accuracy comparing with the approaches proposed in \[11, 13\]. The experiments on natural images demonstrate its efficiency and effectiveness.

The rest of this letter is organized as follows. In Section II a closed-form solution for designing the robust projection matrix is derived. Moreover, the formulation of this projection matrix can save computations during the sparsifying dictionary updating procedures which will be shown in Section III. An online learning sparsifying dictionary algorithm with taking the projection noise into account is proposed in Section III and then the joint online optimization algorithm for the design of PMSD is formulated. Some experiments on natural images are carried out in Section IV to demonstrate the efficiency and effectiveness of the proposed algorithm comparing with the state-of-the-art methods. Some conclusions are given in Section V to end this letter.

II. PROJECTION MATRIX DESIGN

In this section, a closed-form solution for robust projection matrix design is derived and the form of this solution is suitable for involving in the dictionary learning procedure to save the computation which will be given in the next section. According to one of the results proposed in \[10\], it is better to solve the following problem to obtain a robust projection matrix when the SRE exists:

\[
\min_{\Phi} \|I_L - \Psi^T \Phi^T \Psi\|_F^2 + \lambda \|\Phi\|_F^2
\]

(5)

where \(I_L\) represents an identity matrix with dimension \(L\) and \(^T\) denotes the transpose operator. Obviously, an optimal \(\lambda\) would yield a better projection matrix. Although \(\lambda\) can be determined through the practical experiments, it is inconvenience to make sure which one is better in joint optimization situation because we have to go visit (5) many times. One of the heuristic methods is to find a set of solutions which leads to a minimal value for \(\|I_L - \Psi^T \Phi^T \Psi\|_F^2\) and then locate a \(\Phi\) among these solutions which has a minimal \(\|\Phi\|_F^2\). Mathematically speaking, the following optimization problem is considered:

\[
\min_{\Phi} \|\Phi\|_F^2 \\
\text{s.t.} \quad G(\Phi) = \varepsilon \\
G(\Phi) \triangleq \|I_L - \Psi^T \Phi^T \Psi\|_F^2 \\
\text{Rank}(\Phi) = M
\]

(6)

where \(\text{Rank}(\Phi)\) is utilized to calculate the rank of the matrix \(\Phi\) and \(\varepsilon\) is the minimal value of the function \(G(\Phi)\) with \(\text{Rank}(\Phi) = M\).

Typically, such an optimization problem is hard to solve. However, due to the special structure in (6), an analytic solution can be specified through the following Lemma.

**Lemma 1.** Let \(\Phi = U \Psi \Lambda \Psi^T \Phi^T \Psi\) with an SVD of \(\Psi\), where \(\text{Rank}(\Psi) = \tilde{N}\) and \(\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_{\tilde{N}}) > 0\) with \(\lambda_1 > \lambda_2 > \ldots > \lambda_{\tilde{N}}\). The optimal solution for (6) is specified by

\[
\Phi = U [I_M 0] \begin{bmatrix} V^T \Lambda^{-1} 0 \\ 0 0 \end{bmatrix} U_{\Psi}^T
\]

(7)

where \(U \in \mathbb{R}^{M \times M}\) denotes an arbitrary orthonormal and \(V = [J_M 0] \in \mathbb{R}^{N \times \tilde{N}}\) with \(J_M\) a permutation matrix and \(J_{\tilde{N}-M}\) an arbitrary orthonormal matrix.

**Proof.** According to \[10\] Theorem 2], a class of solutions for minimizing \(G(\Phi)\) with \(\text{Rank}(\Phi) = M\) is specified by

\[
\Phi = U [I_M 0] \begin{bmatrix} V^T \Lambda^{-1} 0 \\ 0 0 \end{bmatrix} U_{\Psi}^T
\]

where \(V\) is an arbitrary orthonormal matrix. Now, we tend to prove \(V\) should be equal to \(V\) to make (8) have a minimal \(\|\Phi\|_F^2\).

Denote \(\text{Tr}(\cdot)\) as the trace operator. Solving (7) is equal to address the following problem:

\[
\min_{\Phi} \|\Phi\|_F^2 = \text{Tr} (\Phi^T \Phi)
\]

\[
= \text{Tr} \left( \begin{bmatrix} V^T \Lambda^{-2} V & 0 \\ 0 & 0 \end{bmatrix} [I_M 0] \right)
\]

\[
= \text{Tr}(\Delta(1 : M, 1 : M))
\]

\[
= \sum_{i=1}^{M} \Delta(i,i)
\]

\[
\geq \sum_{i=1}^{M} \frac{1}{\lambda_i^2}
\]

where \(\Delta = V^T \Lambda^{-2} V\) and the last inequality is from \[2\ Corollary 3.39, pp. 248\]. The equality is satisfied if and only if the former \(M\)-th diagonal elements in \(\Delta\) are equal to its former \(M\)-th eigenvalues ordered as ascending, i.e., \(\frac{1}{\lambda_1^2}, \frac{1}{\lambda_2^2}, \ldots, \frac{1}{\lambda_M^2}\). To satisfy this, the matrix \(V\) have to equal to \(V\). This completes the proof.

The matrices \(U\) and \(V\) can be directly set to the identity matrix here because the identity matrix is also an orthonormal and permutation matrix. Moreover, utilizing the identity matrix here, some computations can be saved. Now, one of the solutions for (6) is

\[
\Phi = [A_M^{-1} 0] U_{\Psi}^T
\]

(8)

where \(A_M = \Lambda(1 : M, 1 : M)\). Clearly, \(A_M\) is a diagonal matrix and the calculation of its inversion is cheap. In effect, one time SVD of the dictionary \(\Psi\) dominates the main complexity in the projection matrix updating procedure. However, compared with the methods shown in \[11, 13\] which need the eigenvalue decomposition or SVD many times, it already saves many computations.

III. ONLINE DICTIONARY LEARNING APPROACH WITH CONSIDERING PROJECTION NOISE

In this section, an online algorithm is proposed to deal with (3) and then the joint optimization algorithm is formulated. The proposed method for (3) contains two stages. Firstly, the sparse coefficient vectors in \(\Theta\) are calculated with
a fixed $\Psi$ and then the sparsifying dictionary $\Psi$ is updated with a fixed $\Theta$. A novel online algorithm is utilized to solve (3) and the detailed steps are summarized in Algorithm 1.

Algorithm 1 Online Dictionary Learning

Initialization:
Training data $X \in \mathbb{R}^{N \times P}$, trade-off parameter $\gamma$, initial projection matrix $\Phi$ and dictionary $\Psi_0$, batch size $\eta \geq 1$, the sparsity level $K$, the power parameter $p$, number of iterations $\text{Iter}_{dic}$.

Output:
Dictionary $\Psi_i$.

1: $A_0 \leftarrow 0$, $B_0 \leftarrow 0$, $C_0 \leftarrow 0$, $i \leftarrow 1$
2: for $t = 1$ to $\text{Iter}_{dic}$ do
3: if $i + \eta \leq P$ then
4: $X_i \leftarrow X(:,i:i + \eta - 1)$, $Y_i \leftarrow \Phi X_i$
5: $i \leftarrow i + \eta$
6: else
7: Shuffle $X$, $i \leftarrow 1$
8: $X_i \leftarrow X(:,i:i + \eta - 1)$, $Y_i \leftarrow \Phi X_i$
9: $i \leftarrow i + \eta$
10: end if
11: Sparse coding: compute using Orthogonal Matching Pursuit (OMP)
\begin{align*}
\Theta_i = \arg\min_{\Theta_i} \left\| \sqrt{\gamma}X_i - \sqrt{\gamma}\Psi_{i-1} - \Phi \Theta_i \right\|_F^2 \\
\text{s.t.} \left\| \Theta_i(:,k) \right\|_0 \leq K, \forall k
\end{align*}
12: $A_i \leftarrow \left(1 - \frac{1}{t}\right)A_{i-1} + \frac{1}{t}\Theta_i \Psi_i^T$
13: $B_i \leftarrow \left(1 - \frac{1}{t}\right)B_{i-1} + \frac{1}{t}X_i \Theta_i^T$
14: $C_i \leftarrow \Phi B_i$
15: Compute $\Psi_i$ using Algorithm 2 with $\Psi_i$ as warm restart, that is,
\begin{align*}
\Psi_i &= \arg\min_{\Psi_i \in C} \frac{1}{2} \sigma(\Psi_i) \\
\sigma(\Psi_i) &= \frac{1}{2} \sum_{j=1}^p \left( \| \sqrt{\gamma}X_i - \Psi_i \Theta_i \|_F^2 \\
&+ \| Y_j - \Phi \Theta_i \|_F^2 \right) \\
&= \frac{1}{2} \text{Tr} \left( \Psi_i^T \Psi_i (\gamma A_i) \right) - \text{Tr} \left( \Psi_i^T \gamma B_i \right) + \frac{1}{2} \text{Tr} \left( \Psi_i^T \Phi \Psi_i \Phi A_i \right) - \text{Tr} \left( \Psi_i^T \Phi \Phi C_i \right)
\end{align*}
16: end for
17: return $\Psi_{\text{iter}_{dic}}$ (learned dictionary)

For simplicity, OMP is used as to act the sparse coding role in Algorithm 1 which is the same as in (11) - (13). In the dictionary updating procedure, we intend to utilize the block-coordinate descent with warm restart. Concretely, each column of $\Psi$ is sequentially updated. The gradient of (11) with respect to $j$-th column of $\Psi$ is
\begin{align}
\frac{\partial \sigma(\Psi_j)}{\partial \Psi_j} &= \gamma \Psi_j - \gamma b_j + \Phi^T \Phi \Psi_j - \Phi^T c_j
\end{align}
where $\Psi_j$, $a_j$, $b_j$, and $c_j$ are the $j$-th column of the matrices $\Psi$, $A_j$, $B_j$ and $C_j$, respectively. To make (11) equal to zero, the $j$-th column of $\Psi$ should be updated as in (13). The matrices $\Xi_1$, $\Xi_2$ and $\Xi_3$ are equal to $(I_N - \Phi^T \Phi)^{-1}$, $(I_N - \Phi^T \Phi)^{-1} \Phi^T$, and $(I_N - \Phi^T \Phi)^{-1}$, respectively. Due to the special structure of $\Phi$ shown in (3), the matrices $\Xi_1$, $\Xi_2$ and $\Xi_3$ can be calculated simply as follows
\begin{align}
\Xi_1 &= U_\Psi \left[ (\gamma^{-1} A^2_M + I_M)^{-1} \right]_{:,1} \\
\Xi_2 &= U_\Psi \left[ (\gamma^{-1} A^2_M + A_M)^{-1} \right]_{:,1} \\
\Xi_3 &= U_\Psi \left[ (\gamma^{-1} I_M + A^2_M)^{-1} \right]_{:,1}
\end{align}
Although we have to calculate the inversion of matrices in (12), the computational burden becomes cheaper than the previous one because the related matrices here are diagonal matrices. The detailed steps related the dictionary updating are given in Algorithm 2. In order to avoid the trivial solution, each column of the dictionary is normalized to have a unit $\ell_2$ norm directly.

Algorithm 2 Dictionary Update

Initialization:
Dictionary $\Psi_i$.

1: repeat
2: for $j = 1$ to $L$ do
3: Update the $j$-th column to optimize (10):
\begin{align}
\Psi_{i-1}(:,j) &\leftarrow \Xi_1 \left[ \frac{b_j - \Psi_{i-1} a_j}{A_{i-1}(j,j)} + \Psi_j \right] + \Xi_2 c_j \\
&+ \Xi_3 \left[ \frac{1}{\gamma} \Psi_j - \frac{1}{\gamma} \Psi_{i-1}(j,j) \right] \\
\Psi_{i-1}(:,j) &\leftarrow \frac{\Psi_{i-1}(:,j)}{\| \Psi_{i-1}(j,:) \|_2}
\end{align}
4: end for
5: until convergence
6: return $\Psi_{i-1}$ (updated dictionary)

Now, the joint optimization PMSD can be formulated as in Algorithm 2. Although we utilize the same framework as given in (11), (13), the proposed algorithm is more suitable for working on a large training dataset and yields a better promising performance in terms of signal recovery accuracy. The simulation results in the next section illustrate its efficiency and effectiveness.
Algorithm 3 Joint Optimization Projection Matrix and Sparsifying Dictionary

**Initialization:**
- Initial projection matrix $\Phi_0$ and dictionary $\Psi_0$, number of iterations $\text{Iter}_{\text{predic}}$.
- Output:
  - The projection matrix $\Phi$ and the sparsifying dictionary $\Psi$.

1. for $i = 1$ to $\text{Iter}_{\text{predic}}$ do
2.   Update the projection matrix utilizing (8) and compute the three matrices $\varepsilon_1, \varepsilon_2, \varepsilon_3$.
3.   Call Algorithm 1 to update the dictionary.
4. end for
5. return $\Phi$ and $\Psi$

IV. SIMULATION RESULTS

Some experiments on natural images are posed in this section to illustrate the performance of the proposed Algorithm 3 denoted as CS$_{Alg3}$. We also compare our method with the ones given in [11, 13] which also utilize the same framework as ours but based on the batch method. Although [13] developed the closed-form solutions for every updating procedures, we will show it is still inefficient for the case when the training dataset is large. The methods given in [11, 13] are denoted as CS$_{S-DCS}$ and CS$_{BL}$, respectively. Both training and testing data are extracted from the LabelMe database [17]. All of the experiments are carried out on a laptop with Intel(R) i7-6500 CPU @ 2.5GHz and RAM 8G.

The signal reconstruction accuracy is evaluated in terms of Peak Signal to Noise Ratio (PSNR) given in [2]

$$p_{\text{psnr}} \triangleq 10 \times \log_{10} \left( \frac{2^r - 1}{\rho_{\text{mse}}} \right) \text{dB}$$

with $r = 8$ bits per pixel and $\rho_{\text{mse}}$ is defined as

$$\rho_{\text{mse}} \triangleq \frac{1}{N \times P} \sum_{k=1}^{P} \| \hat{x}_k - x_k \|_2^2$$

where $x_k$ is the original signal, $\hat{x}_k = \Psi \hat{\theta}_k$ stands for the recovered signal and $P$ is the number of patches in an image or testing data. The training and testing data are obtained through the following method.

**Training data** A set of $8 \times 8$ non-overlapping patches is obtained by randomly extracting 400 patches from each of the images in the whole LabelMe training dataset, with each patch of $8 \times 8$ arranged as a vector of $64 \times 1$. A set of $400 \times 2920 = 1.168 \times 10^6$ training samples is received for training.

**Testing data** The testing data is extracted from the LabelMe testing dataset. Here, we randomly extract 15 patches from 400 images and each sample is an $8 \times 8$ non-overlapping patch. Finally, we obtain 6000 testing samples.

$8 \times 10^4$ and $6 \times 10^3$ patches are randomly chosen from the $1.168 \times 10^6$ training data for CS$_{S-DCS}$ and CS$_{BL}$, respectively, because these two methods cannot stand too large training patches. In order to show the influence of the size of the training dataset, the same $6 \times 10^3$ patches which is prepare for CS$_{BL}$ are also utilized by CS$_{S-DCS}$. For convenience, this case is called CS$_{S-DCS - small}$. The parameters in these two methods are chosen as recommended in their papers. To keep the same dimensions in $\Phi$, $\Psi$ and sparsity level as given in [13], $M$, $L$ and $K$ are set to 20, 256 and 4, respectively, in CS$_{Alg3}$. The parameters $\gamma, \eta, \text{Iter}_{\text{dic}}$ and $\text{Iter}_{\text{predic}}$ are set to $\frac{1}{3}, 128, 1000$ and $10$ in the proposed Algorithm 3. The initial projection matrix and dictionary for [11, 13] are random one and DCT dictionary, respectively. The initial sparsifying dictionary in the proposed algorithm is randomly chosen from the training data and the corresponding projection matrix is obtained through the method shown in Section II. The signal recovery accuracy of the aforementioned methods on testing data is shown in Fig. 2. The corresponding CPU time of the four cases in seconds are given in Table II.

![Fig. 1. The $p_{\text{psnr}}$ of the four different cases versus the iteration number on testing data.](image)

| Case         | CPU Time (Seconds) |
|--------------|--------------------|
| CS$_{S-DCS - small}$ | $2.79 \times 10^3$ |
| CS$_{S-DCS}$    | $1.32 \times 10^4$ |
| CS$_{BL}$       | $4.33 \times 10^4$ |
| CS$_{Alg3}$     | $1.54 \times 10^4$ |

Benefiting from the large training dataset, CS$_{S-DCS}$ yields a better performance in terms of $p_{\text{psnr}}$ than CS$_{S-DCS - small}$. This indicates that enlarging the training dataset can lead to a better PMSD. Compared with CS$_{S-DCS - small}$, CS$_{BL}$ has a higher $p_{\text{psnr}}$ which meets the same result as shown in [13]. However, CS$_{BL}$ needs many SVDs in the algorithm which make it inefficient and become hard to extend to the situation when the training dataset is large. This concern can be observed from Table II that CS$_{BL}$ needs much more CPU time even only 6000 training patches. Although CS$_{BL}$ has a better performance than CS$_{S-DCS - small}$, this advantage will disappear if we enlarge the size of the training dataset in CS$_{S-DCS}$. It can be seen from Fig. 2 that CS$_{S-DCS}$ has a similar performance to CS$_{S-DCS - small}$.

4 According to our experiments, the initial value chosen as this strategy can receive a better performance in our method compared with the initial setting suggested in [11, 13]. However, the different initial values will not influence the final result much in our experiments.
performance with CS$_{BL}$, but it has a shorter training time, see Table II. To the proposed algorithm CS$_{Alg3}$, it has a better performance in terms of $\rho_{psnr}$ compared with other methods. Meanwhile, CS$_{Alg3}$ has a shorter CPU time but has the largest training dataset. It indicates that Algorithm 3 is suitable for training the PMSD on large training dataset. Clearly, training on a large dataset can obtain a better PMSD and the proposed Algorithm 3 is a best choice which combines the efficiency and effectiveness simultaneously.

We also investigate the performance of the above obtained PMSD to the natural images. Due to the limited space, the simulation results are given as a supplemental material. The supplemental material will be uploaded online in the future.

V. CONCLUSION

In this letter, an efficient algorithm which can train the PMSD on a large dataset is proposed. Training the PMSD on a large dataset can receive a better promising performance and the proposed method in this letter which considers the efficiency and effectiveness simultaneously is a suitable choice for such a task.

One of the possible directions for future research is to develop an accelerated algorithm to make the proposed method faster. Involving the Sequential Subspace Optimization (SESOP) in the algorithm may belong to one of the possible methods to realize the accelerated purpose.

REFERENCES

[1] S. Mallat, A wavelet tour of signal processing: the sparse way, Academic press, 2008.
[2] M. Elad, Sparse and Redundant Representations: from theory to applications in signal and image processing, Springer Science & Business Media, 2010.
[3] I. Tosic and P. Frossard, “Dictionary Learning,” IEEE Signal Process. Mag., vol. 28, pp. 27-38, Mar. 2011.
[4] K. Engan, S. O. Aase, and J. H. Hakon-housoy, “Method of optimal direction for frame design,” Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., vol. 5, pp. 2443-2446, Mar. 1999.
[5] M. Aharon, M. Elad, and A. Bruckstein, “K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation,” IEEE Trans. Signal Process., vol. 54, pp. 4311-4322, Nov. 2006.
[6] E. J. Candès and M. B. Wakin, “An introduction to compressive sampling,” IEEE Signal Process. Mag., vol. 25, pp. 21-30, Mar. 2008.
[7] M. Elad, “Optimized projections for compressed sensing,” IEEE Trans. Signal Process., vol. 55, pp. 5695-5702, Dec. 2007.
[8] T. Hong, H. Bai, S. Li and Z. Zhu, “An efficient algorithm for designing projection matrix in compressive sensing based on alternating optimization,” Signal Process., vol. 125, pp. 9-20, Aug. 2016.
[9] G. Li, X. Li, S. Li, H. Bai, Q. Jiang and X. He, “Designing robust sensing matrix for image compression,” IEEE Trans. Image Process., vol. 24, pp. 5389-5400, Dec. 2015.
[10] T. Hong and Z. Zhu, “Robust Projection Matrix Design and Its Application in Compression,” https://arxiv.org/abs/1609.08281, 2016.
[11] J. M. Durate-Carvajalino and G. Sapiro, “Learning to sense sparse signals: simultaneously sensing matrix and sparsifying dictionary optimization,” IEEE Trans. Image Process., vol. 18, pp. 1395-1408, Jul. 2009.
[12] W. C. Chan and M. R. D. Rodrigues, “Dictionary learning with optimized projection design for compressive sensing applications,” IEEE Signal Process. Lett., vol. 20, pp. 992-995, Oct. 2013.
[13] H. Bai, G. Li, S. Li, Q. Li, Q. Jiang, and L. Chang, “Alternating optimization of sensing matrix and sparsifying dictionary for compressed sensing,” IEEE Trans. Signal Process., vol. 63, pp. 1581-1594, Mar. 2015.
[14] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, “Online dictionary learning for sparse coding,” Proceedings of the 26th international conference on machine learning, ACM, pp. 689-696, Jun. 2009.
[15] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, “Online learning for matrix factorization and sparse coding,” Journal of Machine Learning, vol. 11, pp. 19-60, Jan. 2010.
[16] M. Zibulevsky and M. Elad, “$\ell_1$-$\ell_2$ Optimization in Signal and Image Processing,” IEEE Signal Process. Mag., vol. 27, pp. 76-88, May 2010.
[17] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, “LabelMe: A Database and Web-Based Tool for Image Annotation,” International Journal of Computation Vision, vol. 77, pp. 157-173, May 2008.
[18] E. Richardson, R. Herskovitz, B. Ginsburg, and M. Zibulevsky, “SEBOOST-Boosting stochastic learning using subspace optimization techniques,” accepted by Advanced in Neural Information Process. Systems, NIPS, 2016.
VI. ADDITIONAL SIMULATION RESULTS

In here, we investigate the performance of the four different cases mentioned in this letter on ten natural images. The Structural Similarity Index (SSIM) is also involved in comparing the recovered natural images by the different methods mentioned in this letter. The results are shown in Table II. Clearly, the proposed Algorithm 3 has the highest PSNR and SSIM. Compared with $CS_{S-DCS} - small$, $CS_{S-DCS}$ has a higher PSNR and SSIM on all of the ten tested natural images. This meets the result given in this letter that enlarging the size of the training dataset is significant. This can also be illustrated by the methods between $CS_{BL}$ and $CS_{S-DCS}$. Note that $CS_{BL}$ works better than $CS_{S-DCS} - small$ when they have the same small size of training data. However, the performance of $CS_{S-DCS}$ will exceed $CS_{BL}$ when the size of the training data is enlarged. All of these told us training the projection matrix and the corresponding sparsifying dictionary on a large dataset is significant. Moreover, the proposed Algorithm 3 is a good choice to attack such a mission. To examine the visual effect clearly, the recovered performance of two natural images, i.e., ‘Lena’ and ‘Mandril’ in Fig. 2, are shown in Fig.s 3 and 4.

Fig. 2. The original tested images. (a) Lena, (b) Mandril.

REFERENCES

[1] G. Li, Z. H. Zhu, D. H. Yang, L. P. Chang, and H. Bai, “On projection matrix optimization for compressive sensing systems,” IEEE Trans. Signal Process., vol. 61, pp. 2887-2898, Jun. 2013.
[2] R. A. Horn and C. R. Johnson, Matrix Analysis, Cambridge University, Second Edition, 2012.
[3] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” IEEE Trans. Image Process., vol. 13, pp. 600-612, Apr. 2004.
TABLE II
PERFORMANCE EVALUATED WITH DIFFERENT ALGORITHMS SHOWN IN THIS LETTER. (LEFT: PSNR, RIGHT: SSIM. THE HIGHEST IS MARKED WITH BOLD.)

|       | CS\textsubscript{S}−DCS − small | CS\textsubscript{S}−DCS | CS\textsubscript{BL} | CS\textsubscript{Alg3} |
|-------|----------------------------------|-------------------------|---------------------|-----------------------|
| Lena  | 33.0566 0.9089                   | 33.1859 0.9114         | 33.3059 0.9111      | 34.6557 0.9281        |
| Elane | 32.3970 0.8713                   | 32.6393 0.8945         | 32.4076 0.8944      | 33.1756 0.9244        |
| Man   | 31.1978 0.8738                   | 31.5599 0.8866         | 31.4686 0.8782      | 32.5941 0.8999        |
| Mandril | 23.4291 0.7598                | 23.8411 0.7823         | 23.8221 0.7746      | 24.3753 0.8007        |
| Peppers | 28.9462 0.8877                | 29.6975 0.9005         | 29.4145 0.8925      | 30.6859 0.9169        |
| Boat  | 29.7350 0.8561                   | 30.3488 0.8679         | 30.1027 0.8580      | 31.2858 0.8837        |
| House | 31.5166 0.8842                   | 32.0602 0.8985         | 32.0707 0.8956      | 33.0146 0.9158        |
| Cameraman | 26.2240 0.8581              | 26.8272 0.8716         | 26.4545 0.8673      | 27.4254 0.8877        |
| Barbara | 25.6148 0.8229                | 25.9153 0.8316         | 25.5165 0.8168      | 26.0835 0.8393        |
| Tank  | 30.7233 0.8252                   | 30.8210 0.8309         | 31.1403 0.8361      | 31.7813 0.8576        |
| Averaged | 29.2842 0.8485                | 29.7838 0.8609         | 29.5703 0.8534      | 30.5078 0.8754        |

Fig. 3. The recovered tested image ‘Lena’. (a) CS\textsubscript{S}−DCS − small, (b) CS\textsubscript{S}−DCS, (c) CS\textsubscript{BL}, (d) CS\textsubscript{Alg3}.

Fig. 4. The recovered tested image ‘Mandril’. (a) CS\textsubscript{S}−DCS − small, (b) CS\textsubscript{S}−DCS, (c) CS\textsubscript{BL}, (d) CS\textsubscript{Alg3}.