SMOOTHED SIMCO FOR DICTIONARY LEARNING: HANDLING THE SINGULARITY ISSUE

Xiaochen Zhao, Guangyu Zhou, Wei Dai

Department of Electrical and Electronic Engineering, Imperial College London, United Kingdom
{xiaochen.zhao10, g.zhou11, wei.dai1}@imperial.ac.uk

ABSTRACT

Typical algorithms for dictionary learning iteratively perform two steps: sparse approximation and dictionary update. This paper focuses on the latter. While various algorithms have been proposed for dictionary update, the global optimality is generally not guaranteed. Interestingly, the main reason for an optimization procedure not converging to a global optimum is not local minima or saddle points but singular points where the objective function is not continuous. To address the singularity issue, we propose the so called smoothed SimCO, where the original objective function is replaced with a continuous counterpart. It can be proved that in the limit case, the new objective function is the best possible lower semi-continuous approximation of the original one. A Newton CG method is implemented to solve the corresponding optimization problem. Simulations demonstrate the proposed method significantly improves the performance.

1. INTRODUCTION

A good dictionary plays a decisive role in sparse signal representation applications (e.g. signal denoising, image inpainting and data classification). However, in many applications, pre-defined dictionaries may not be available, for example, blind source separation and compressed sensing with imprecise hardware calibration. For those applications, one needs to learn the dictionary from the available training data. Dictionary learning is the technique to find an over-complete dictionary that accurately represent the signals with sparse coefficients.

Dictionary learning algorithms typically involve iteratively solving two problems: sparse coding and dictionary update. Sparse coding aims at finding an optimal sparse approximation of the training samples with a given dictionary. Algorithms including l_1-minimization [1] or greedy algorithms (e.g. OMP [2] and SP [3]) are often used to solve this problem.

The goal of dictionary update is to refine the dictionary for a given sparsity pattern. In MOD designed by Engan, et al. [4], during each iteration the dictionary is updated by fixing the sparse coefficients and solving a least squares problem. In 2006, Aaron, et al. generalized the K-means method and developed K-SVD algorithm [5]. The K-SVD structure is considered where each time a single codeword is updated with the corresponding sparse coefficients. More recently, Dai, et al. designed SimCO algorithm [6] where dictionary update is formulated as an optimization problem on manifolds. SimCO allows a simultaneous update of an arbitrary subset of codewords and the corresponding coefficients. It has been shown in [6] that MOD and K-SVD can be viewed as special cases of SimCO. See [7, 8] and references therein for other techniques.

Unfortunately, all the above dictionary update algorithms do not guarantee the global optimality. The observation in [6] says that the main reason for benchmark algorithms failing to converge to a global minimizer is the singularity issue. That is, the objective function is not continuous and an optimization procedure may get trapped in the neighborhood of singular points. To address this issue, the authors in [6] proposed regularized SimCO, where a regularization term is added to the objective function.

In this paper, we propose a smoothing technique for SimCO, termed *smoothed SimCO*. The major contributions of this paper are:

- A continuous objective function is proposed to replace the original one. This new objective function results in significant improvement according to the numerical tests.
- We prove that the proposed objective function, in the limit, is the best possible lower semi-continuous approximation of the original one. The lower semi-continuity guarantees that the solution set is closed, which is required for a convergence of any optimization procedure. By contrast, the regularized objective function proposed in [6] does not have this property.
- A Newton CG method is designed to minimize the proposed objective function. It turns out that the corresponding computations are highly non-trivial. In this paper, key formulae for implementing the Newton CG method are derived. Numerical tests verify that our implementation achieves a good balance between convergence rate and computational complexity.

The rest of this paper will be organized as following: an introduction of the SimCO algorithm is given in Section 2. Then in Section 3 we present the smoothed objective function and give a discussion on the selection of the parameters. An efficiency of the algorithm implementation is discussed in Section 4. Numerical comparison between our proposed algorithm and the mainstream algorithms are shown in Section 5. Finally we conclude our work in Section 6.

2. PRELIMINARIES: THE SIMCO FRAMEWORK

The SimCO framework is designed for the dictionary update stage and can be briefly summarized as follows. Consider the problem of updating a dictionary $\boldsymbol{D} \in \mathbb{R}^{m\times d}$ from the training samples $\boldsymbol{Y} \in \mathbb{R}^{m\times n}$. Let Ω be the index set of nonzero coefficients, i.e., $\Omega = \{(i,j) : \boldsymbol{X}_{i,j} \neq 0\}$. The SimCO framework assumes that the sparsity pattern Ω is fixed during the update process. Define the feasible set for sparse coefficients $\mathcal{X}(\Omega) = \left\{\boldsymbol{X} \in \mathbb{R}^{d\times n} : X_{i,j} = 0,\ \forall (i,j) \notin \Omega\right\}$. Follow the convention [6] in defining the feasible set for the dictionary:

$$\mathcal{D} = \left\{\boldsymbol{D} \in \mathbb{R}^{m\times d} : \left\|\boldsymbol{D}_{:,k}\right\|_2 = 1,\ \forall k \in [d]\right\},$$

where $[d] = \{1, 2, \ldots, d\}$. The SimCO framework formulates the dictionary update problem as

$$\min_{\boldsymbol{D}\in\mathcal{D}} f(\boldsymbol{D}) = \min_{\boldsymbol{D}\in\mathcal{D}} \underbrace{\min_{\boldsymbol{X}\in\mathcal{X}(\Omega)} \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}\|_F^2}_{f(\boldsymbol{D})},$$

where the inner optimization problem has a closed-form solution $\boldsymbol{X}^*$ given by

$$\boldsymbol{X}^*_{i,j} = 0,\ \forall (i,j) \notin \Omega,\ \boldsymbol{X}^*_{\Omega(:,j),j} = \boldsymbol{D}^{\dagger}_{:,\Omega(:,j)}\boldsymbol{Y}_{:,j},$$

where $\Omega(:,j) = \{i : (i,j) \in \Omega\}$ is the index set of nonzero elements in $\boldsymbol{X}_{:,j}$, and $\boldsymbol{D}^{\dagger}_{:,\Omega(:,j)}$ is the pseudo-inverse of matrix $\boldsymbol{D}_{:,\Omega(:,j)}$. It is worth to mention that the objective function $f(\boldsymbol{D})$ can be decomposed as a summation of atomic functions:

$$f(\boldsymbol{D}) = \sum_{i=1}^{n} \underbrace{\min_{\boldsymbol{X}_{:,i}\in\mathcal{X}(\Omega(:,i))} \|\boldsymbol{Y}_{:,i} - \boldsymbol{D}\boldsymbol{X}_{:,i}\|_F^2}_{f_i(\boldsymbol{D})},$$

where $\mathcal{X}(\Omega(:,i))$ is defined as the set of all vectors satisfying the given sparsity pattern.

As discussed in [6], the objective function $f(\boldsymbol{D})$ is not continuous. The singularity of $f(\boldsymbol{D})$ is the main reason behind the failure of a dictionary update algorithm[1]: it has been shown that when the benchmark algorithms, including MOD, K-SVD and SimCO, fail, they typically get trapped in the neighborhood of singular points. For Regularized SimCO [6], an l_2 penalty term is added to the objective function to address the discontinuity issue. However our analysis shows that local minimum may be generated by regularization. (see the journal version of this paper [9] for more details).

[1]The singularity issue appears even when the true sparsity pattern is known [6]. This suggests that it is inherent in the dictionary update stage.

3. SMOOTHED OBJECTIVE FUNCTION

To address the singularity issue, we propose a smoothed objective function in this section. Different from [6] where a regularization term is added, we introduce multiplicative terms. Our approach is based on the identification of when the objective function $f(\boldsymbol{D})$ becomes discontinuous.

Definition 1. A dictionary $\boldsymbol{D} \in \mathbb{R}^{m\times d}$ is singular under a given sparse pattern Ω if there exists an $i \in [n]$ such that $\boldsymbol{D}_{:,\Omega(:,i)}$ is column rank deficient. Or equivalently, the minimum singular value of $\boldsymbol{D}_{:,\Omega(:,i)}$ is zero.

For compositional convenience, denote the sub-dictionary $\boldsymbol{D}_{:,\Omega(:,i)}$ by $\boldsymbol{D}_i$. The multiplicative terms, referred to as *modulation functions* henceforth, are designed to "filter" out the singular points. For given constants $0 \leq \delta_1 \leq \delta_2$, we define a modulation function[2] $g_{\boldsymbol{\delta}}(\lambda)$ as:

$$g_{\boldsymbol{\delta}}(\lambda) = \begin{cases} 0 & \text{if } \lambda \leq \delta_1, \\ 6a^5(\lambda) - 15a^4(\lambda) + 10a^3(\lambda) & \text{if } \delta_1 < \lambda < \delta_2, \\ 1 & \text{if } \delta_2 \leq \lambda, \end{cases}$$

where $a(\lambda) = (\lambda - \delta_1)/(\delta_2 - \delta_1)$. The constant $\delta_2 - \delta_1$ defines how fast $g_{\boldsymbol{\delta}}(\lambda)$ changes from 0 to 1 as λ increases. This function is designed such that it is the simplest piecewise polynomial that is second order differentiable.

With the definition of the modulation function, we propose to replace the original objective function with the following one

$$\tilde{f}(\boldsymbol{D}) = \sum_i \underbrace{f_i(\boldsymbol{D}) \cdot g_{\boldsymbol{\delta}_i}(\lambda_{\min}(\boldsymbol{D}_i))}_{\tilde{f}_i(\boldsymbol{D}) \text{ or } \tilde{f}_i(\boldsymbol{D}_i)}.$$

Here the notation $\lambda_{\min}(\boldsymbol{D}_{:,i})$ represents for the minimum singular value of $\boldsymbol{D}_{:,i}$, and the subscript $\boldsymbol{\delta}_i \triangleq \left(\delta_1^{(i)}, \delta_2^{(i)}\right)$ emphasizes that the thresholds $0 < \delta_1^{(i)} < \delta_2^{(i)}$ are different for different $i \in [n]$. (The choices of $\boldsymbol{\delta}_i$ are specified later.) It is also worth to note that each atomic function $\tilde{f}_i$ is a function of the sub-dictionary $\boldsymbol{D}_i$, and hence a function the overall dictionary $\boldsymbol{D}_i$. The dictionary update problem is then formulated as $\min_{\boldsymbol{D}\in\mathcal{D}} \tilde{f}(\boldsymbol{D})$.

The new objective function $\tilde{f}(\boldsymbol{D})$ has several properties especially convenient for the problem at hand.

[2]In practice, we only consider two cases: $0 < \delta_1 < \delta_2$ or $\delta_1 = \delta_2 = 0$. The presented definition works for $0 < \delta_1 < \delta_2$. When $0 = \delta_1 = \delta_2$, $g_{\boldsymbol{\delta}}(\lambda) = 1$ for $\lambda > 0$ and $g_{\boldsymbol{\delta}}(\lambda) = 0$ for $\lambda = 0$.

Theorem 1.

1. When $0 < \delta_1^{(i)} < \delta_2^{(i)}, \forall i$, $\tilde{f}(\boldsymbol{D})$ is continuous.
2. Consider the limit case where $\delta_1^{(i)}, \delta_2^{(i)} \to 0$ with $0 < \delta_1^{(i)} < \delta_2^{(i)}, \forall i$. The following hold.
   (a) $\tilde{f}(\boldsymbol{D})$ and $f(\boldsymbol{D})$ differ only at the singular points.
   (b) $\tilde{f}(\boldsymbol{D})$ is the best possible lower semi-continuous approximation of $f(\boldsymbol{D})$.

Due to the space constraint, the proof is omitted here and will be presented in the journal version of this paper [9]. The properties for the limit case distinguish the proposed $\tilde{f}$ and the one in [6].

The effect of adding the modulation functions, intuitively speaking, is to open "tunnels" for the optimization process to pass through. The smaller $\boldsymbol{\delta}_i$'s are, the better the function $\tilde{f}$ approximates the function f, but the narrower the tunnels are, and the slower the convergence rate is. The next subsection discusses a particular way to choose the parameters.

3.1. Choices of the Thresholds

We use random matrix theory to choose $\boldsymbol{\delta}_i$'s.

We first argue that for different i, $\boldsymbol{\delta}_i$ should be different. Consider the case where $m = 100$, $|\Omega(:,1)| = 2 \ll m$ and $|\Omega(:,2)| = m$. Suppose that the dictionary $\boldsymbol{D}$ is randomly generated from the uniform distribution on $\mathcal{D}$.[3] It is clear that with high probability $\lambda_{\min}(\boldsymbol{D}_1)$ centers around 1 but $\lambda_{\min}(\boldsymbol{D}_2)$ is close to zero. Intuitively, the thresholds $\boldsymbol{\delta}_i$'s should be chosen such that the modulation functions take effect (i.e., $g_{\boldsymbol{\delta}_i} < 1$) with small but positive probability.

Generally speaking, it is difficult to quantify the probability of $\lambda_{\min}(\boldsymbol{D}_i)$'s. Nevertheless, when m and $s_i := |\Omega(:,i)|$ approaches infinity with a constant ratio, the distribution of $\lambda_{\min}$ will converge a distribution only dependent of the ratio s_i/m. In particular,

Proposition 1. *For any given m and s_i such that $s_i \leq m$, define $\mathcal{D}_{m,s_i}$ as the set containing all the matrices with unit norm columns. Randomly generate $\boldsymbol{D}_i$ from the uniform distribution on $\mathcal{D}_{m,s_i}$. Then as m, $s_i \to \infty$ simultaneously with $s_i/m \to c_i < 1$, the minimum singular values $\lambda_{\min}(\boldsymbol{D}_i)$ converges to $\tau_i \triangleq 1 - \sqrt{c_i}$ in probability.*

The proof will be detailed in the journal version of this paper [9]. Though the results are asymptotic, they provide a good approximation for finite m and s_i. In our implementation, we set $\delta_2^{(i)} = \alpha\tau_i$ where $\alpha \in (0,1)$ is a constant independent of i, and $\delta_1^{(i)} = \delta_2^{(i)}/200$.

[3] The uniform distribution is well defined as $\mathcal{D}$ is a compact manifold.

4. ALGORITHM IMPLEMENTATION

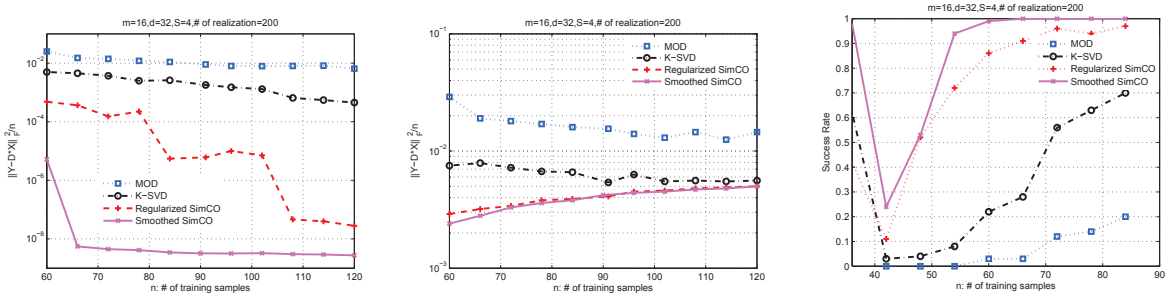
In this section, we present a Newton CG implementation to minimize the objective function $\tilde{f}(\boldsymbol{D})$. Most optimization methods are based on the so called line search strategy. The dictionaries at the beginning and the end of the k-th iteration, denoted by $\boldsymbol{D}^{(k)}$ and $\boldsymbol{D}^{(k+1)}$ respectively, can be related by $\boldsymbol{D}^{(k+1)} = \boldsymbol{D}^{(k)} + \alpha^{(k)}\boldsymbol{\eta}^{(k)}$ where $\alpha^{(k)}$ is the appropriately chosen step size and $\boldsymbol{\eta}_k$ is the search direction. The step size $\alpha^{(k)}$ can be determined by using criteria presented in [10, 11]. The search direction $\boldsymbol{\eta}^{(k)}$ plays the key role in determining the convergence rate. Generally speaking, a Newton direction is preferred (compared with the gradient descent direction) [11]. In a standard Newton method, the computation of the Newton direction requires the Hessian of the objective function. Note that in the problem at hand, the variable $\boldsymbol{D}$ has size $m \times d$ and hence the corresponding Hessian has size $md \times md$. To compute the Hessian explicitly, it requires large computational resource as well as extra-ordinary storage resource. By contrast, Newton CG provides a means to compute the Newton direction without explicitly computing the Hessian.

More specifically, the Newton CG method starts with the gradient descent direction $\boldsymbol{\eta}_0$ and iteratively refines it towards the Newton direction. The detailed steps in finding a good search direction are given in Algorithm 1. Denote the gradient of $\tilde{f}(\boldsymbol{D})$ as $\nabla\tilde{f}(\boldsymbol{D})$. Denote $\nabla_{\boldsymbol{\eta}}\left(\nabla\tilde{f}(\boldsymbol{D})\right) \in \mathbb{R}^{m\times d}$ as the directional derivative of $\nabla\tilde{f}(\boldsymbol{D})$. In each iteration of the proposed algorithm, instead of computing the Hessian $\nabla^2\tilde{f} \in \mathbb{R}^{md\times md}$ explicitly, one only needs to compute $\nabla_{\boldsymbol{\eta}}\left(\nabla\tilde{f}\right)$. The required computational and storage resources are therefore much less than working with the Hessian directly. Due to the space constraint, we postpone the computation details of $\nabla\tilde{f}$ and $\nabla_{\boldsymbol{\eta}}\left(\nabla\tilde{f}\right)$ to the journal version of this paper [9].

5. EMPIRICAL TESTS

The settings for the numerical tests are as follows. The training samples are generated according to $\boldsymbol{Y} = \boldsymbol{D}_{\text{true}}\boldsymbol{X}_{\text{true}} + \boldsymbol{W}$ where $\boldsymbol{W} \in \mathbb{R}^{m\times n}$ are Gaussian noise ($\boldsymbol{W} = \boldsymbol{0}$ for the noiseless case). The dictionary $\boldsymbol{D}_{\text{true}}$ is randomly generated from the uniform distribution on $\mathcal{D}$. Regarding the sparse coefficients, we assume that each column of $\boldsymbol{X}_{\text{true}}$ contains exactly s many non-zero elements of which the locations are randomly generated from the corresponding uniform distribution. The nonzero elements of $\boldsymbol{X}_{\text{true}}$ are randomly generated from the standard Gaussian distribution. To separate the effect of sparse coding, we also assume that the sparse coding stage is perfect, i.e., the true sparsity pattern Ω_{true} is available. The more realistic scenario where sparse coding is combined with dictionary update is tested in the journal version [9] but here.

Both noiseless and noisy case are considered in the tests. Let $\hat{\boldsymbol{D}}$ and $\hat{\boldsymbol{X}}$ be the learned dictionary and the



(a) Noiseless case: normalized distortion. (b) Noisy case: normalized distortion. (c) Noiseless case: success rate.

Fig. 4.1: The performance comparison.

Algorithm 1 The Newton CG algorithm: find the search direction.

Input: $\boldsymbol{D}$; **Output**: $\boldsymbol{\eta}$.

Define: $\mathcal{P}\left(\boldsymbol{\eta}_{:,i}\right) = \left(\boldsymbol{I} - \boldsymbol{D}_{:,i}\boldsymbol{D}_{:,i}^T\right)\boldsymbol{\eta}_{:,i}$.

For $k = 0,\ 1,\ 2,\ ...$

Define tolerance $\epsilon_k = \min\left(0.5, \sqrt{\left\|\nabla\tilde{f}\right\|}\right)\left\|\nabla\tilde{f}\right\|$.

Set $\boldsymbol{z}_0 = \boldsymbol{0}$, $\boldsymbol{r}_0 = \nabla\tilde{f}$, $\boldsymbol{d}_0 = -\boldsymbol{r}_0 = -\nabla\tilde{f}$.

For $j = 0,\ 1,\ 2,\ ...$

Set $\boldsymbol{H}_j = \nabla_{\boldsymbol{d}_j}\left(\nabla\tilde{f}\right)$.

$\forall i$, let $\left(\boldsymbol{H}_j\right)_{:,i} = \mathcal{P}\left(\left(\boldsymbol{H}_j\right)_{:,i}\right)$.

If $tr\left(\boldsymbol{d}_j^T\boldsymbol{H}_j\right) \geqslant 0$

If $j = 0$

return $\boldsymbol{\eta} = -\nabla\tilde{f}$.

else

return $\boldsymbol{\eta} = \boldsymbol{z}_j$.

Set $\alpha_j = tr\left(\boldsymbol{r}_j^T\boldsymbol{r}_j\right)/tr\left(\boldsymbol{d}_j^T\boldsymbol{H}_j\right)$.

Set $\boldsymbol{r}_{j+1} = \boldsymbol{r}_j + \alpha_j\boldsymbol{H}_j$.

If $\|\boldsymbol{r}_{j+1}\| < \epsilon_k$

return $\boldsymbol{\eta} = \boldsymbol{z}_{j+1}$.

Set $\beta_{j+1} = tr\left(\boldsymbol{r}_{j+1}^T\boldsymbol{r}_{j+1}\right)/tr\left(\boldsymbol{r}_j^T\boldsymbol{r}_j\right)$.

Set $\boldsymbol{d}_{j+1} = -\boldsymbol{r}_{j+1} + \beta_{j+1}\boldsymbol{d}_j$.

end

$\forall i$, let $\boldsymbol{\eta}_{:,i} = \mathcal{P}\left(\boldsymbol{\eta}_{:,i}\right)$.

corresponding sparse coefficients, respectively. The normalized learning error is defined as $\left\|\boldsymbol{Y} - \hat{\boldsymbol{D}}\hat{\boldsymbol{X}}\right\|_F^2/n$. The criteria for success learning are designed for both cases using the normalized learning error: in the noiseless case, a success is claimed when $\left\|\boldsymbol{Y} - \hat{\boldsymbol{D}}\hat{\boldsymbol{X}}\right\|_F^2/n \leq \epsilon_e\|\boldsymbol{Y}\|_F^2$ where the constant ϵ_e is ideally zero but set to 10^{-6} in practice; for the noisy case, the criterion for a successful learning is given by $\left\|\boldsymbol{Y} - \hat{\boldsymbol{D}}\hat{\boldsymbol{X}}\right\|_F^2/n \leq \epsilon_n\|\boldsymbol{Y}\|_F^2$ where $\epsilon_n := \|\boldsymbol{W}\|_F^2/n/\|\boldsymbol{D}_{true}\boldsymbol{X}_{true}\|_F^2$.

In the tests, four algorithms, namely MOD, K-SVD, regularized SimCO, and smoothed SimCO, are compared. For each of these algorithms, the maximum number of iterations is set to 1000. For regularized SimCO, the regularization constant is initially set as $\mu = 0.1$ and then reduced to $\mu/10$ after every 100 iterations. In smoothed SimCO, the thresholds $\boldsymbol{\delta}_i$'s are set to $(0.001, 0.2)$ for the first 500 iterations and then to $(0, 0)$ for the rest 500 iterations. (Note that $\boldsymbol{\delta}_i = \boldsymbol{\delta}_j$ due to the simulation setting.)

The simulation results are presented in Figure 4.1, where the first two sub-figures compare the normalized distortion and the last one focuses on the success rate. The advantage of the proposed smoothed SimCO is clear for both noiseless and noisy cases. In terms of success rate, smoothed SimCO reaches 100% success rate when the number of training samples $n > 60$ while MOD and K-SVD could not achieve 100% success rate even when $n \geq 84$. It is also interesting to observe the dip in the success rate when n is in the middle-range (Figure 4.1c). This is expected. On one hand, the success rate should increase when the number of training samples becomes larger. On the other hand, when the number of training samples is extremely low, for example, $n = 1$, the learning problem becomes trivial. Hence, the most difficult case is when n is in the middle-range.

6. CONCLUSION

We presented a new method for dictionary learning based on the SimCO framework to address the singularity problem occurred in the dictionary update stage. We rigorously analyzed the proposed objective function and proved certain good properties. A Newton CG method is implemented to achieve a good balance between convergence rate and computational complexity. Numerical results of the proposed algorithms demonstrate the significant performance improvement.

7. REFERENCES

[1] E. Candes and T. Tao, "Decoding by linear programming," *IEEE Trans. Inf. Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[2] Y. C. Pati, R. Rezaiifar and P. S. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition," *IEEE Asilomar Conf. Signals, Syst., Comput.*, pp. 40–44, 1993.

[3] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE Trans. Inf. Theory*, vol. 55, pp. 2030–2249, 2009.

[4] K. Engan, S. Aase and J. H. Husoy, "Method of optimal directions for frame design," *IEEE Int. Conf. Acoustics, Speech, Signal Process.*, vol. 5, pp. 2443–2446, 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, no. 11, pp. 4311–4322, 2006.

[6] W. Dai, T. Xu and W. Wang, "Simultaneous codeword optimization (SimCO) for dictionary update and learning," *IEEE Trans. Signal Process.*, vol. 60, pp. 6340–6353, 2012.

[7] B. Mailhe and M. D. Plumbley, "Dictionary learning with large step gradient descent for sparse representations," *IEEE Int. Conf. LVA/ICA*, pp. 231–238, 2012.

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

[9] X. Zhao, G. Zhou and W. Dai, "Dictionary learning: a singularity problem and how to handle it," In preparation.

[10] A. Edelman, T. A. Arias and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM J. Matrix Anal. Appl.*, vol. 20, no. 2, pp. 303–353, 1999.

[11] J. Nocedal and S. J. Wright, *Numerical Optimization*, New York: Springer, 1999.