A NEW ALGORITHM FOR LEARNING OVERCOMPLETE DICTIONARIES

Mostafa Sadeghi[1], *Massoud Babaie-Zadeh*[1], *and Christian Jutten*[2]

[1]Electrical Engineering Department, Sharif University of Technology, Tehran, IRAN
[2]GIPSA-Lab, Grenoble, and Institut Universitaire de France, France.

ABSTRACT

In this paper, we propose a new algorithm for learning overcomplete dictionaries. The proposed algorithm is actually a new approach for optimizing a recently proposed cost function for dictionary learning. This cost function is regularized with a term that encourages low similarity between different atoms. While the previous approach needs to run an iterative limited-memory BFGS (l-BFGS) algorithm at each iteration of another iterative algorithm, our approach uses a closed-form formula. Experimental results on reconstruction of a true underlying dictionary and designing a sparsifying dictionary for a class of autoregressive signals show that our approach results in both better quality and lower computational load.

***Index Terms*—** Compressed sensing, sparse signal approximation, overcomplete dictionary learning

1. INTRODUCTION

1.1. Sparse signal approximation

Sparse approximation of signals has received a lot of attention during the last decade due to its applications in many different areas such as compressed sensing [1] and image processing [2]. Let $\mathbf{D} \triangleq [\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_m]$ be a set of m vectors, called *atoms*, each with dimension n, and $\mathbf{y}$ be a signal of the same dimension. $\mathbf{D}$ is called a *dictionary* or a collection of atoms and it is overcomplete, i.e. $m > n$. The sparse approximation problem is then to represent $\mathbf{y}$ as a linear combination of as few as possible atoms. This amounts to solve the following minimization problem:

$$\mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x}} \|\mathbf{x}\|_0 \quad \text{subject to} \quad \|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2 \leq \epsilon, \tag{1}$$

where $\|.\|_0$ is the so-called ℓ_0 pseudo-norm that counts the number of nonzero components. The above problem needs a combinatorial search and is generally NP-hard [2]. So, alternative methods are used to solve it [2, 3]. One of the most successful ideas is to replace the non-convex sparsity measure $\|.\|_0$ with its best convex approximation $\|.\|_1$ [4] which leads to the following convex problem:

$$\mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x}} \|\mathbf{x}\|_1 \quad \text{subject to} \quad \|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2 \leq \epsilon. \tag{2}$$

1.2. Dictionary learning

In the above sparse approximation problem, the overcomplete dictionary, $\mathbf{D}$, is assumed to be known. However, the impact of a suitable dictionary for a given class of signals, i.e. the one that provides sufficient sparse approximation for all of the signals of that class, is crucial in many applications such as image enhancement and compression [2]. This dictionary can either be chosen as a predetermined set of atoms such as overcomplete DCT dictionary for natural images, or learned from a given set of training signals. The later can be better fitted to a particular class of signals and leads to more promising results in many applications such as image denoising [5], classification tasks [6], and so on.

During the last few years, many dictionary learning algorithms have been introduced to address the problem of learning overcomplete dictionaries [7]. These algorithms can be considered as a generalization of the well-known K-means clustering algorithm [8]. While in K-means, we restrict each signal to use only one atom (called centroid), in the sparse approximation case, each signal is allowed to use more than one atom provided that it uses as few as possible atoms.

Consider a set of signals $\{\mathbf{y}_i\}_{i=1}^L$ where $\forall i : \mathbf{y}_i \in \mathbb{R}^n$. Putting these signals, called training data, as the columns of the matrix $\mathbf{Y}$, the dictionary learning problem is then to solve the following minimization problem:

$$(\mathbf{D}^*, \mathbf{X}^*) = \operatorname*{argmin}_{\mathbf{D}\in\mathcal{D}, \mathbf{X}\in\mathcal{X}} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2, \tag{3}$$

where $\|.\|_F$ is the Frobenius norm, and $\mathcal{D}$ and $\mathcal{X}$ are admissible sets of the dictionary and the coefficient matrix, respectively. In this paper, $\mathcal{D}$ is the set of all dictionaries with unit column-norm. Since we require that each signal has a sparse approximation, $\mathcal{X}$ is the set of all matrices $\mathbf{X}$ with sparse columns.

Note that problem (3) is non-convex with respect to both $\mathbf{D}$ and $\mathbf{X}$. Up to our best knowledge, almost all dictionary learning algorithms solve (3) alternatively: Starting with an initial dictionary and coefficient matrix, the following two stages are repeated several iterations,

1. **Sparse approximation:** with a fixed $\mathbf{D}$, solve (3) for $\mathbf{X}$,

2. **Dictionary update:** with a fixed $\mathbf{X}$, update the dictionary to reduce the approximation error.

The first stage is simply an ordinary sparse approximation problem where the dictionary is known. So, the main difference between dictionary learning algorithms is their approach for performing the second stage. Generally, this dictionary update problem is as follows:

$$\min_{\mathbf{D}\in\mathcal{D}} \|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2. \tag{4}$$

One of the simplest dictionary learning algorithms is the Method of Optimal Directions (MOD) [9] which firstly finds the unconstrained minimum of $\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2$ and then projects the solution onto the set $\mathcal{D}$. This leads to the following closed-form expression:

$$\mathbf{D} = \mathbf{Y}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}, \tag{5}$$

followed by normalizing the columns of D.

Another well-known dictionary learning algorithm is K-SVD [8]. K-SVD solves (4) by updating the atoms sequentially, i.e. for updating each atom, the others are kept fixed. In K-SVD along with each atom, the nonzero entries of the corresponding row vector in the coefficient matrix are also updated. This problem leads to a matrix rank-1 approximation that can be solved by Singular Value Decomposition (SVD).

1.3. Low mutual coherence dictionaries

An important desired property of a dictionary is its low mutual coherence. The mutual coherence of a dictionary is defined as the maximum absolute value of the cross-correlations between the atoms [10]. Specifically, let $\mathbf{G} = [g_{ij}] = \mathbf{D}^T\mathbf{D}$ be the Gram matrix of $\mathbf{D}$ (with normalized columns). The mutual coherence of $\mathbf{D}$ is then defined as:

$$\mu(\mathbf{D}) = \max_{i\neq j} |g_{ij}|. \tag{6}$$

The mutual coherence of a dictionary has an important role in sparse approximation problems [2, 10]. Smaller values of $\mu(\mathbf{D})$ implies less similarity between atoms which in turn results in improved performance of many sparse recovery algorithms including BP [11] and Orthogonal Matching Pursuit (OMP) [12].

Very recently, Sigg *et al.* [13] proposed a dictionary learning method that instead of using (4) for stage 2, regularizes (4) with a term that encourages low *self-coherency* of the dictionary, i.e. the pairwise similarity of the atoms. They proposed the following problem to be solved for stage 2:

$$\min_{\mathbf{D}\in\mathcal{D}} \|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2 + \frac{\lambda}{2}\|\mathbf{D}^T\mathbf{D}-\mathbf{I}\|_F^2. \tag{7}$$

By varying the value of λ, we can make a trade-off between minimizing the approximation error and minimizing the self coherence of the dictionary. To solve (7), [13] proposes to use the limited-memory BFGS (l-BFGS) algorithm [14]. In other words, at each iteration between stages 1 and 2, stage 2 itself has to be solved using an iterative optimization algorithm (l-BFGS).

In this paper, we derive a new dictionary learning algorithm based on solving (7) as its second stage. Our algorithm uses a simple closed-form formula for stage 2, and has a better performance, both in terms of quality and speed, as will be shown by our simulations in Section 3.

The rest of the paper is organized as follows. In Section 2 we describe our proposed algorithm in details. Then Section 3 presents the experimental results.

Algorithm 1 The proposed algorithm

```
1: Task: Dictionary learning for training data $\mathbf{Y}$.
2: Initialization: Set $\mathbf{D} = \mathbf{D}^{(0)}$, $k = 0$.
3: The main loop: Repeat until convergence:
4: Sparse Approximation: $\mathbf{X}^{(k+1)} = \mathrm{SA}(\mathbf{Y}, \mathbf{D}^{(k)})$.
5: Dictionary Update: Set $\mathbf{X} = \mathbf{X}^{(k+1)}$, $\mathbf{D} = \mathbf{D}^{(k)}$,
   $\hat{\mathbf{G}} = \mathbf{D}^T\mathbf{D} - \mathbf{I}$, then: $\mathbf{D}^{(k+1)} = \mathbf{Y}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\hat{\mathbf{G}})^{-1}$
   Normalize each column, and set $k \leftarrow k + 1$.
```

2. THE PROPOSED ALGORITHM

To solve (7), we set its gradient equal to zero. This leads to the following equation:

$$-(\mathbf{Y}-\mathbf{D}\mathbf{X})\mathbf{X}^T + \lambda\mathbf{D}(\mathbf{D}^T\mathbf{D}-\mathbf{I}) = 0. \tag{8}$$

An exact closed-form solution for this equation would be tricky. However, noting that (7) is itself inside an iterative algorithm (that is, it has to be solved as stage 2 of an iterative algorithm between stages 1 and 2), we propose to replace the term $\mathbf{D}^T\mathbf{D}$ with the Gram matrix of the current dictionary, $\mathbf{D}^{(k)}$. In other words, we solve the following equation for $\mathbf{D}$:

$$-(\mathbf{Y}-\mathbf{D}\mathbf{X})\mathbf{X}^T + \lambda\mathbf{D}(\mathbf{G}^{(k)}-\mathbf{I}) = 0, \tag{9}$$

where k indicates alternation number between the two dictionary learning stages and $\mathbf{G}^{(k)}$ is the Gram matrix of $\mathbf{D}^{(k)}$. Finally, the updated dictionary is obtained as follows[1]:

$$\mathbf{D}^{(k+1)} = \mathbf{Y}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\hat{\mathbf{G}}^{(k)})^{-1}, \tag{10}$$

where $\hat{\mathbf{G}}^{(k)} = \mathbf{G}^{(k)} - \mathbf{I}$ with its diagonal elements all equal to zero. Although norm of each atom is also penalized, a column normalization is done. Note the similarity between (10) and that of MOD in (5).

Algorithm 1 shows a description of the proposed algorithm, in which $\mathrm{SA}(\mathbf{Y}, \mathbf{D})$ denotes the coefficient matrix found by sparsely approximating of $\mathbf{Y}$ in $\mathbf{D}$ via any sparse

[1]We have omitted the alternation number, k, from $\mathbf{X}$ for convenience in the notation.

recovery algorithms (OMP in our experiments). As will be seen in the next section, the computational burden of our proposed algorithm is much less than those of K-SVD and [14] (henceforth named as the l-BFGS).

3. EXPERIMENTAL RESULTS

In this section, we evaluate the efficiency of our proposed algorithm with two experiments. These experiments are similar to those in [15, 8]. For the sparse approximation stage, we use OMP. In the first experiment, we compare the ability of our proposed algorithm to K-SVD[2] and l-BFGS algorithms in reconstruction of a known dictionary. In the second experiment, we consider training vectors from a class of autoregressive (AR) signals. In this case, there is no underlying known dictionary, but as stated in [15], the only goal is to minimize the approximation error for the training set[3].

Our simulations were performed in MATLAB R2010b environment on a system with 3.8 GHz CPU and 8 GB RAM, under Microsoft Windows 7 operating system. As a rough measure of complexity, we will mention the run times of the above algorithms.

3.1. Reconstruction of a known dictionary

We generated a random matrix of size 20×50 as the generating dictionary, with zero mean and unit variance independent and identically distributed (i.i.d.) Gaussian entries. A collection of 2000 training data $\{\mathbf{y}_i\}_{i=1}^{2000}$ were produced, each as a linear combination of s different columns of the dictionary, with zero mean and unit variance i.i.d. Gaussian coefficients in random and uniformly independent positions. We varied s from 3 to 6. We then added white Gaussian noise with Signal to Noise Ratio (SNR) levels of 10, 20, 30, and 100 dB. We applied the three algorithms onto this noisy training data, and compared the resulting recovered dictionaries to the generating dictionary as follows. Assume that $\mathbf{d}_i$ is a generating atom and $\bar{\mathbf{d}}_i$ is the atom in the recovered dictionary that best matches $\mathbf{d}_i$ among the others. We say that the recovery is successful if $|\mathbf{d}_i^T \bar{\mathbf{d}}_i|$ is above 0.99 [8]. The percentage of the correct recovery is used as the measure of successfully reconstructing the generating dictionary.

We performed 100 alternations between sparse approximation and dictionary update stages for all three algorithms. The initial dictionary was made by randomly choosing different signals from the training set followed by a normalization. For all of these experiments we fixed $\lambda = 0.5$ in our algorithm and the l-BFGS algorithm. We repeated each trial 50 times and averaged the results. Figure 1 shows the resulting

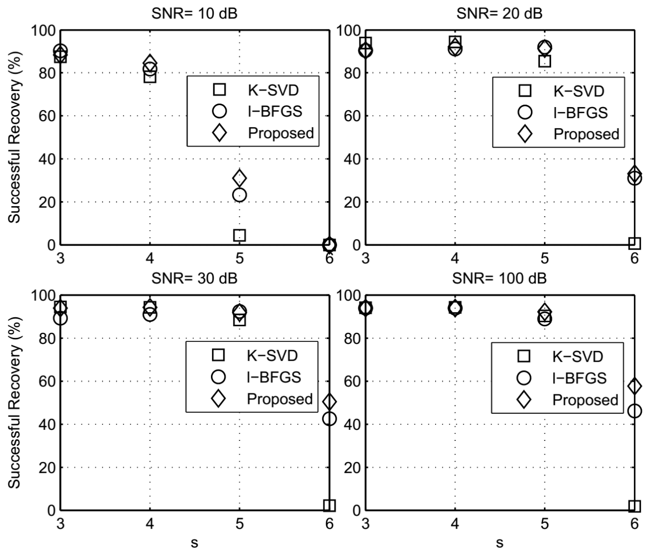

Fig. 1. Results of the experiment on synthetic data. Each figure corresponds to a certain noise level.

successful recovery ratios. As can be seen, the performance of the proposed algorithm is comparable and even slightly better than the other two algorithms. The average execution times of K-SVD, l-BFGS, and our algorithm were 26.88, 13.67, and 2.96 seconds, respectively.

3.2. Sparse Approximation of an AR(1) signal

In this experiment, we consider an AR(1) signal (according to [15]), that is generated as $v(k) = 0.95v(k-1) + e(k)$, where $e(k)$ is Gaussian noise with zero mean and unit variance. A collection of $L = 2000$ training data were made by chopping this signal into vectors of length $n = 20$. Number of atoms was set to $m = 40$ and $s = 5$ atoms were used to approximate each training vector. For all of the three algorithms 100 alternations between stages 1 and 2 were done. As in [15], we computed SNR as $\text{SNR} = 10 \log \|\mathbf{Y}\|_F^2 / \|\mathbf{Y} - \mathbf{DX}\|_F^2$.

To find the best value of the regularization parameter (λ) for the setup of this experiment, we repeated our algorithm and l-BFGS for a sequence of λ's. For each value of λ we repeated the two algorithms 50 times and averaged the results. The final SNR for these two algorithms is plotted as a function of λ in Fig. 2. As can be seen, for our algorithm $\lambda = 85$, and for the l-BFGS $\lambda = 40$ results in the best SNR, although these algorithms are not too sensitive to it (this is especially the case for our algorithm). SNR versus alternation number (averaged over 50 trials) is plotted in Fig. 3. This figure shows that the proposed algorithm outperforms the l-BFGS and K-SVD in the sense that it reaches higher SNR value. From Fig. 2 and Fig. 3 we see that our algorithm has a very smooth behaviour, while it is not the case for the l-BFGS. The average execution times of K-SVD, l-BFGS, and our algorithm were

[2]For K-SVD and OMP we have used K-SVD-Box v10 and OMP-Box v10 available at `http://www.cs.technion.ac.il/~ronrubin/software.html`

[3]As pointed out in the previous works, e.g. [15], the performance of MOD and K-SVD is very similar in these experiments. So, we omitted MOD from the simulations.

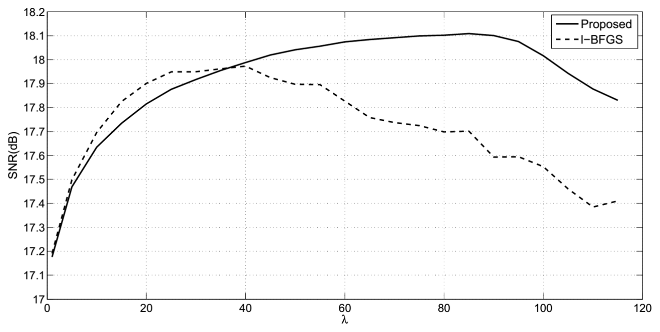

Fig. 2. The final SNR as a function of λ in our algorithm and the l-BFGS algorithm.

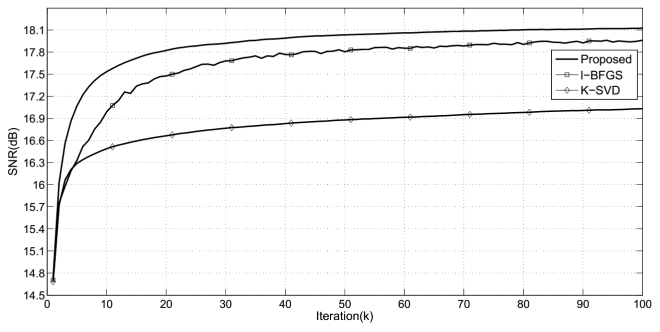

Fig. 3. Results of the experiment on AR(1) signal. SNR is plotted as a function of the alternation number, k, during the learning process. The regularization parameter for our algorithm and the l-BFGS algorithm is $\lambda = 85$, and $\lambda = 40$, respectively.

15.88, 6.69, and 1.25 seconds, respectively.

4. CONCLUSION

In this paper, we derived a new dictionary learning algorithm. Our algorithm is based on solving a recently proposed dictionary learning cost function that controls the trade-off between the approximation error and the self-coherency of the dictionary [13]. Contrary to [13] that proposes to use an iterative limited-memory BFGS (l-BFGS) algorithm inside the alternating minimization iterations, we proposed a closed-form formula to be used inside these iterations. In this way, our algorithm has no additional input parameters. Experimental results on recovery of a known dictionary and designing a sparsifying dictionary for an AR(1) signal showed that compared to l-BFGS and K-SVD algorithms, not only our algorithm has a lower computation load, but also it results in a better quality.

5. REFERENCES

[1] D. L. Donoho, "Compressed sensing," *IEEE Trans. on Information Theory*, vol. 52, no. 4, pp. 1289–1306, April 2006.

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

[3] H. Mohimani, M. Babaie-Zadeh, and Ch. Jutten, "A fast approach for overcomplete sparse decomposition based on smoothed ℓ^0 norm," *IEEE Trans. on Signal Processing*, vol. 57, pp. 289–301, 2009.

[4] S. S. Chen, D. D. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Rev.*, vol. 43, pp. 129–159, 2001.

[5] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. on Image Processing*, vol. 15, no. 12, pp. 3736 – 3745, 2006.

[6] 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.

[7] 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.

[8] 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.

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

[10] D. L. Donoho and X. Huo, "Uncertainty principles and ideal atomic decomposition," *IEEE Trans. Information Theory*, vol. 47, no. 7, pp. 2845–2862, 2001.

[11] J. A. Tropp, "Just relax: Convex programming methods for identifying sparse signals in noise," *IEEE Transactions on Information Theory*, vol. 52, no. 3, pp. 1030–1051, March 2006.

[12] J. A. Tropp, "Greed is good: algorithmic results for sparse approximation," *IEEE Trans. Inform. Theory*, vol. 50, no. 10, pp. 2231–2242, 2004.

[13] C. D. Sigg, T. Dikk, and J. M. Buhmann, "Learning dictionaries with bounded self-coherence," *IEEE Signal Processing Letters*, vol. 19, no. 12, pp. 861–864, 2012.

[14] D. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Math. Progr.*, vol. 45, no. 1, pp. 503–528, 1989.

[15] K. Skretting and K. Engan, "Recursive least squares dictionary learning algorithm," *IEEE Trans. on Signal Processing*, vol. 58, pp. 2121 – 2130, 2010.