SPARSE NULL SPACE BASIS PURSUIT AND ANALYSIS DICTIONARY LEARNING FOR HIGH-DIMENSIONAL DATA ANALYSIS

Xiao Bian *Hamid Krim* *Alex Bronstein*[†] *Liyi Dai*[‡]

Department of Electrical Engineering, North Carolina State University, Raleigh, NC, USA
[†]School of Electrical Engineering, Tel Aviv University, Tel Aviv, Israel
[‡]Computing Sciences Division, U.S. Army Research Office, Research Triangle Park, NC, USA

ABSTRACT

Sparse models in dictionary learning have been successfully applied in a wide variety of machine learning and computer vision problems, and have also recently been of increasing research interest. Another interesting related problem based on a linear equality constraint, namely the sparse null space problem (SNS), first appeared in 1986, and has since inspired results on sparse basis pursuit.

In this paper, we investigate the relation between the SNS problem and the analysis dictionary learning problem, and show that the SNS problem plays a central role, and may be utilized to solve dictionary learning problems. Moreover, we propose an efficient algorithm of sparse null space basis pursuit, and extend it to a solution of analysis dictionary learning. Experimental results on numerical synthetic data and real-world data are further presented to validate the performance of our method.

Index Terms— Sparse null space problem, analysis dictionary learning, sparse representation, high dimensional signal processing

1. INTRODUCTION

High dimensional data analysis has been a high focus of research in diverse areas, including machine learning, computer vision, and applied mathematics, on account of its theoretical complexity, and high relevance to big data problems. Dictionary learning has been one of the fundamental methodologies to handle high dimensional data, and has successfully been applied to feature extraction [1], denoising [1] [2] [3], recognition and classification [4], etc.

Specifically, recent research has shown that sparse models are crucial to learning a discriminating and robust dictionary [2]. In particular, data samples are assumed to lie on a union of subspaces (UoS), and a parsimonious constraint on the use of atoms to represent each data point helps to recover the underlying basis of each subspace [7] [8]. Both synthesis models and analysis models were proposed to reveal the sparsity property of data. In synthesis models, we have a synthesis dictionary $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_n]$ such that $\mathbf{x}_i = \sum_{j \in S} \mathbf{d}_j w_{ij}, \|S\|_0 \leq k$, where $\mathbf{x}_i$ is our data, $\mathbf{d}_j$ is the j^{th} atom in the dictionary, and w_{ij} the corresponding coefficient. On the other hand, an operator H is adopted in an analysis model such that $H \circ \mathbf{x}_i$ gives a sparse vector representing $\mathbf{x}_i$ [9][3].

Another interesting problem invoking sparsity, is the sparse null space problem (SNS), first proposed in 1986 by Coleman and Pothen [10]. This is yet to be related to the state-of-the-art methods in dictionary learning. The SNS problem may be stated as finding a sparse basis for the null space of a given matrix $\mathbf{A}$. The derived elegant results of the sparse null space problem, turn out not only useful in helping us understand the dictionary learning problem, but as we further elaborate, also in providing insight for solving practical problems.

In this paper, we hence study the relation between the SNS problem and the dictionary learning problem, and as we shall show next, the SNS problem is equivalent to the analysis dictionary learning (ADL) problem. We then proceed to solve an ADL problem using methods for the SNS problem. Specifically, inspired by the existing results for the SNS problem and the state-of-the-art of sparsity pursuit algorithms, we present an l_1 minimization-based greedy algorithm to solve the SNS problem. In contrast to current mainstream algorithms [2][3][4][11], the convergence of our method is assured by both the convergence of the greedy algorithm and the convex l_1 minimization. Moreover, we demonstrate its superior performance on both synthetic numerical data and real-world data.

The remainder of this paper is organized as follows. In Section 2, we analyze the relation of the SNS problem to the ADL problem, and show their equivalence. In Section 3, building upon the results of the solution of SNS problem, we present an effective method to solve the SNS problem, which can essentially also be used to handle the ADL problem. Finally, in Section 4 we validate our method on the analysis dictionary learning problem by numerical experiments, and

We would like to acknowledge the support of U.S. Army Research Office: Grant # W911NF-04-D-0003-0022

illustrate the effectiveness of our algorithm on texture classification.

A brief summary of notations used throughout this paper is as follows: The sparsity of an $m \times n$ matrix $\mathbf{X}$, defined as $\frac{\|\mathbf{X}\|_0}{mn}$, is denoted by $\rho(\mathbf{X})$. We denote by $P_{\mathbf{X}}$ the projection matrix onto $\mathrm{col}(\mathbf{X})$, and by $P_{\mathbf{X}^\perp} = \mathbf{I} - P_{\mathbf{X}}^T P_{\mathbf{X}}$ the projection matrix onto $\mathrm{null}(\mathbf{X})$. Additionally, given a vector $\mathbf{y} \in R^n$, operator $(\cdot)_j$ returns the value of the j^{th} element of $\mathbf{y}$. The adjoint operator of $(\cdot)_j$, denoted as $(\cdot)_j^*$, is hence as follows: $(c)_j^* = \mathbf{v} \in R^n$, such that $(\mathbf{v})_j = c$ and $(\mathbf{v})_i = 0$, if $i \neq j$.

2. FROM SNS TO ADL

In this section, we reformulate the SNS problem and the ADL problem in a matrix form, and then proceed to establish the equivalence by way of the common solution they share.

Given any $m \times n$ matrix $\mathbf{A}$ such that $\mathrm{row}(\mathbf{A}) \subset R^n$, the SNS problem may be defined as follows,

$$\mathrm{SNS}(\mathbf{A}) = \arg\min_{\mathbf{N}} \|\mathbf{N}\|_0, \ s.t. \ \mathrm{col}(\mathbf{N}) = \mathrm{null}(\mathbf{A}). \quad (1)$$

Let $\mathbf{X} = [x_1, \ldots, x_n]$ be a generic data matrix, the ADL problem can then be rewritten as

$$\mathrm{ADL}(\mathbf{X}) = \arg\min_{\mathbf{U}} \|\mathbf{U}\|_0,$$
$$s.t. \ \mathbf{DX} = \mathbf{U}, \mathrm{row}(\mathbf{X}) = \mathrm{row}(\mathbf{U}), \quad (2)$$

where $\mathbf{D}$ is an analysis operator in matrix form, and $\mathbf{U}$ is the corresponding sparse coefficient matrix. To avoid trivial solutions as $\mathbf{U} = 0$, we further require $\mathrm{row}(\mathbf{X}) = \mathrm{row}(\mathbf{U})$. Essentially, this is the maximum information we can preserve for $\mathbf{X}$, since any row of $\mathbf{U}$ is a linear combination of rows in $\mathbf{X}$, and hence $\mathrm{row}(\mathbf{U}) \subseteq \mathrm{row}(\mathbf{X})$. In practice, we may also consider the case that $\mathrm{row}(\mathbf{U}) \subset \mathrm{row}(\mathbf{X})$ by further selecting a subset of d_i in $\mathbf{D}$. We are focusing on the generic formulation, i.e. $\mathrm{row}(\mathbf{X}) = \mathrm{row}(\mathbf{U})$, in this section for the sake of theoretical analysis, and will elaborate on this issue later in the discussion of the detailed algorithm.

We note that finding a sparse representation of null space in Problem (1), is equivalent to sparsifying a given matrix $\hat{\mathbf{N}}$ such that $\mathrm{col}(\hat{\mathbf{N}}) = \mathrm{null}(\mathbf{A})$. This coincides with the goal of Problem (2) where the row space of the data matrix $\mathbf{X}$ is instead invoked. In particular, we have the following theorem,

Theorem 1. *Assume* $\mathrm{null}(\mathbf{A}) = \mathrm{row}(\mathbf{X})$*, then a matrix* $\mathbf{N}$ *is a minimizer of the SNS problem (as shown in (1)), if and only if* $\mathbf{N}^T$ *is a minimizer of the ADL problem (as shown in (2))* .

This essentially tells us that we can solve a sparse dictionary learning problem, should we have access to an effective method to solve the corresponding SNS problem. Specifically, given a data matrix $\mathbf{X} = [x_1, \ldots, x_n]$, the analysis dictionary for $\mathbf{X}$ may be constructed in the following three steps:

1. Build a matrix $\mathbf{A}$ such that $\mathrm{row}(\mathbf{A}) = \mathrm{null}(\mathbf{X})$, i.e. $\mathbf{XA}^T = 0$ and $rank(\mathbf{A}) + rank(\mathbf{X}) = n$.
2. Find the sparse feature vectors $\mathbf{U}^T$ by solving $\mathbf{N} = \mathrm{SNS}(\mathbf{A})$.
3. Construct the analysis operator $\mathbf{D}$ from $\mathbf{DX} = \mathbf{U}$.

3. AN ITERATIVE SPARSE NULL SPACE PURSUIT

We have discussed the relation of SNS and ADL in Section 2, and have shown that they may be cast in one unified framework of sparse null space pursuit. Nevertheless, solving SNS is itself a difficult problem. Coleman and Pothen [10] have proved that SNS is essentially NP-hard, hence ruling out a polynomial time algorithm. We however show, it is still possible to approximate the sparse null space basis in polynomial time. In this section, we propose an iterative method based on l_1 minimization for sparse null space pursuit.

3.1. A greedy algorithm for the SNS problem

Previous works on the SNS problem have shed light on finding a solution in polynomial time. In [10], the authors proposed a greedy algorithm for the SNS problem. For the convenience of further discussion, we present the greedy algorithm as Algorithm 1. Additionally, it has been proved in [10] that Algorithm 1 can be used to construct a sparse null space basis, as stated in Theorem 2 [10].

Algorithm 1 A greedy algorithm for sparse null space problem

Initialize: matrix $\mathbf{A} \in R^{m \times d}$, $\mathbf{N} = \emptyset$
for i = 1, ... , q **do**
  Find the sparsest null vector $\mathbf{n}_i$ such that $rank(\mathbf{N} \oplus \mathbf{n}_i) = i$.
  $\mathbf{N} = \mathbf{N} \oplus \mathbf{n}_i$
end for

Theorem 2. *A matrix* $\mathbf{N}$ *is a sparsest null basis of* $\mathbf{A}$ *if and only if it can be constructed by the greedy algorithm.*

It is worth noting that the maximum number of iterations q in Algorithm 1 is constrained by the rank of $\mathbf{A}$, i.e. $q = d - rank(\mathbf{A})$. Moreover, this greedy algorithm can find the global optimal solution for the SNS problem. This elegant result amount to finding the sparsest null space basis of $\mathbf{A}$ in exactly q steps. The subproblem of finding a sparsest null vector itself, is however also a NP-hard problem [10]. We therefore next focus on finding a method to solve this subproblem in each iteration of Algorithm 1.

3.2. l_1-based search for sparse null space

We first reformulate the subproblem of finding a sparsest null vector in Algorithm 1 as follows,

$$\min_{\mathbf{n}_i} \|\mathbf{n}_i\|_0, \ s.t. \ \mathbf{An}_i = 0, P_{\mathbf{N}_i^\perp} \mathbf{n}_i \neq 0, \quad (3)$$

where $\mathbf{N}_i$ is the subspace spanned by the constructed null space vectors in the previous $(i-1)^{th}$ iteration. The condition $P_{\mathbf{N}_i^\perp}\mathbf{n}_i \neq 0$ implies that $\mathbf{n}_i$ is not in the current span of $\mathbf{N}$, and hence $rank(\mathbf{N}_i \oplus \mathbf{n}_i) = rank(\mathbf{N}_i) + 1$.

There are two inherent difficulties in this formulation. First, $\|\cdot\|_0$ is of combinatorial nature, hence the reason of the NP-hardness of the problem. Second, the constraint in (3)

$$P_{\mathbf{N}_i^\perp}\mathbf{n}_i \neq 0, \tag{4}$$

defines a region that is neither compact nor convex. To address the first problem, we propose to take advantage of established results on sparsity pursuit via l_1 minimization [12][13]. While for the second one, and in order to have a convex and compact feasible region, we further adjust the condition $P_{\mathbf{N}_i^\perp}\mathbf{n}_i \neq 0$ as follows,

$$\exists j \in \{1, \ldots, d\}, (P_{\mathbf{N}_i^\perp}\mathbf{n}_i)_j = c, \tag{5}$$

where c is a positive constant.

Algorithm 2 Sparse Null Space Basis Pursuit

```
Initialize: matrix A, N = ∅
for i = 1, ... , p do
    for j = 1, ... , d do
        Find n_i^j = arg min ||n||_1,
        s.t. An = 0, (P_{N^⊥} n)_j = c
    end for
    n_i = arg min ||n_i^j||_0
    N = N ⊕ n_i
end for
```

This is tantamount to solving the following optimization problem for each j in Algorithm 2,

$$\begin{aligned} &\min_{\mathbf{n}} \|\mathbf{n}\|_1, \\ &s.t.\ \mathbf{A}\mathbf{n} = 0, (P_{\mathbf{N}^\perp}\mathbf{n})_j = c. \end{aligned} \tag{6}$$

It is worth noting that the exact recovery of each $\mathbf{n}$ via (6) is determined by the incoherence of the linear operator defined by the two constraints and the sparsity of each $\mathbf{n}$. To solve (6), we adopt the framework of augmented Lagrange method (ALM) on account of its superior performance on matrix-norm minimization problems [14] [15]. Specifically, we have the augmented Lagrange function of (6) as

$$\begin{aligned} L(\mathbf{n}, \mathbf{Y}_1, \mathbf{Y}_2, \mu) = \|\mathbf{n}\|_1 + \langle \mathbf{Y}_1, \mathbf{A}\mathbf{n}\rangle + \langle \mathbf{Y}_2, (P_{\mathbf{N}^\perp}\mathbf{n})_j - c\rangle \\ + \frac{\mu}{2}\|\mathbf{A}\mathbf{n}\|^2 + \frac{\mu}{2}\|(P_{\mathbf{N}^\perp}\mathbf{n})_j - c\|^2. \end{aligned} \tag{7}$$

The primal variable $\mathbf{n}$ is first updated in each iteration with fixed dual variables Y_1, Y_2 and μ. By introducing an auxiliary variable η, we have

$$\mathbf{n}_{k+1} = \mathcal{T}_{\frac{1}{\mu_k \eta}}\left(\mathbf{n}_k - \frac{\mathbf{n}_k^1 + \mathbf{n}_k^2}{\eta}\right), \tag{8}$$

where $\mathcal{T}$ is the soft-thresholding operator, and $\|\eta\|^2 \geq \|\mathbf{A}\|^2 + \|P_{\mathbf{N}^\perp}\|^2$ [15], and

$$\mathbf{n}_k^1 = \mathbf{A}^T\left(\mathbf{A}\mathbf{n}_k + \frac{\mathbf{Y}_1^k}{\mu_k}\right), \tag{9}$$

$$\mathbf{n}_k^2 = P_{\mathbf{N}^\perp}\left((P_{\mathbf{N}^\perp}\mathbf{n})_j - c + \frac{\mathbf{Y}_2^k}{\mu_k}\right)_j^*. \tag{10}$$

Next, the dual variables $\mathbf{Y}_1$, $\mathbf{Y}_2$ and μ are updated as

$$\mathbf{Y}_1^{k+1} = \mathbf{Y}_1^k + \mu_k\left(\mathbf{A}\mathbf{n}_{k+1}\right), \tag{11}$$

$$\mathbf{Y}_1^{k+1} = \mathbf{Y}_1^k + \mu_k\left((P_{\mathbf{N}^\perp}\mathbf{n})_j - c\right), \tag{12}$$

$$\mu_{k+1} = \min\{\rho\mu_k, \mu_{max}\}. \tag{13}$$

The strategy of linearized ALM method provides a fast convergence rate [15]. We utilized this strategy and therefore have an method (Algorithm 2), named Sparse Null Space Basis Pursuit (SNS-BP), to solve the SNS problem efficiently.

Furthermore, in Section 2, we formulate the ADL problem with the constraint $\mathrm{row}(\mathbf{U}) = \mathrm{row}(\mathbf{X})$. However, when a more compact representation of $\mathbf{X}$ is preferred, we may allow $\mathrm{row}(\mathbf{U}) \subset \mathrm{row}(\mathbf{X})$, by which the dimension of the original data space is further reduced. In particular, when $\mathbf{X}$ has been already separated as desired, we may stop SNS-BP before the pursuit null space reaches the maximum dimension.

4. EXPERIMENTS AND VALIDATIONS

In this section, we conduct a series of experiments, using both numerical synthetic data and real-world data, to evaluate our algorithm. In the first part, we use synthetic numerical data that fit the formulation of SNS and ADL to validate the efficacy of our algorithm. In the second part, we show that our algorithm SNS-BP has the potential of solving real-world problems such as textural image classification.

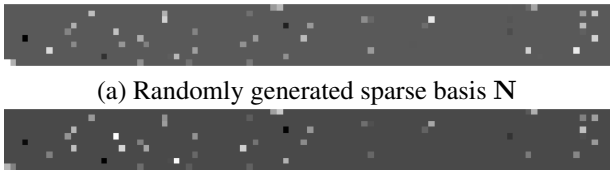
4.1. Numerical experiments on SNS

For a better assessment of the capability of our algorithm, we synthesize data that is compatible with the model of SNS and ADL, and show that SNS-BP is able to reconstruct the sparse null space basis of the SNS problem/the sparse coefficients of the ADL problem.

First, we synthesize a $d \times q$ sparse matrix $\mathbf{N}$ as the sparse null space basis of some matrix $\mathbf{A}$, where $\mathbf{A}$ can be constructed by considering the singular value decomposition of $\mathbf{N}$ and keep the left singular vectors of nonzero singular values as the rows of $\mathbf{A}$, i.e. $\mathrm{row}(\mathbf{A}) = \mathrm{null}(\mathbf{N})$. All elements in $\mathbf{N}$ follow a binomial distribution as zero/nonzero entries. Moreover, the amplitude of each nonzero element is generated from a gaussian distribution.

The matrix $\mathbf{A}$ can then be seen as the input to SNS-BP, and we may therefore compare the recovered null space basis $\hat{\mathbf{N}}$ with the ground truth $\mathbf{N}$. In Fig.1, we show one example of

exact recovery of a sparse null space basis up to permutation and scale.



(a) Randomly generated sparse basis $\mathbf{N}$

(b) Recovered sparse null space basis $\hat{\mathbf{N}}$

Fig. 1. An example of the result of SNS-BP

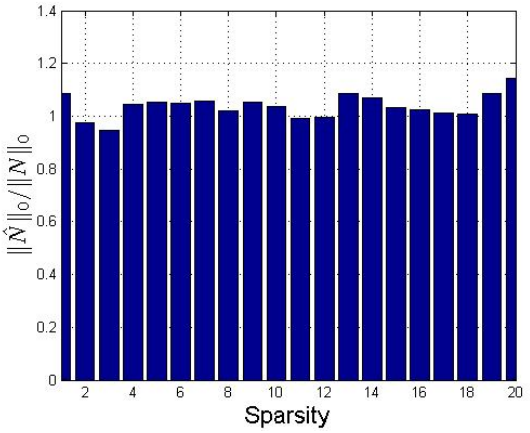


Fig. 2. $\|\hat{\mathbf{N}}\|_0/\|\mathbf{N}\|_0$ vs Sparsity

In Fig.2, we present the sparsity level of $\hat{\mathbf{N}}$ with the sparsity of $\mathbf{N}$ varying from 0.01 to 0.2, i.e. 1% nonzero to 20% nonzero. If our method works well, we would expect it to find the sparsest basis, and therefore $\rho(\hat{\mathbf{N}}) \approx \rho(\mathbf{N})$, i.e. the relative sparsity $\rho(\hat{\mathbf{N}})/\rho(\mathbf{N}) \approx 1$. In Fig.2, 10 experiments have been carried out and the average sparsity is calculated. We can see that the sparse bases discovered by SNS-BP have similar sparsity with $\mathbf{N}$, with $\rho(\mathbf{N})$ from 0.01 to 0.2. Additionally, define the relative error of $\hat{\mathbf{N}}$ as

$$err(\hat{\mathbf{N}}) = \frac{\|\hat{\mathbf{N}}\mathbf{P}\mathbf{\Gamma} - \mathbf{N}\|_F}{\|\mathbf{N}\|_F}, \tag{14}$$

where $\mathbf{P}$ is an arbitrary permutation matrix, and $\mathbf{\Gamma}$ is a diagonal matrix representing the scales of each sparse basis. The average relative error of all the experiments with the sparsity of $\mathbf{N}$ from 0.01 to 0.2 is 1.69%.

4.2. Applications on real-world data

In this part, we further explore the potential of our method on images. The performance of our algorithm is evaluated on texture images from Brodatz database [16]. Each texture image is partitioned into a set of patches, and then the analysis operator learned from patches of different textures is applied to incoming data, which is also segmented into sets of patches. The properties of various textures may lead to different patterns of the corresponding sparse coefficients. We therefore apply the learned operator to incoming data, and compare the distribution of the associated coefficients with those from training sets.

Specifically, we segment each texture image into 10×10 patches, and randomly pick a subset of 120 patches as the training set from each texture image, and the rest of the patches are used as a testing set. In our experiment, we first train the analysis operator by using half of the data in the training set without knowing the label of each patch, and then calculate the distribution of the coefficients P_i of the rest of the patches from the i^{th} class of texture in the training set. In the next testing stage, texture images are used as a set of patches, and we compare the distribution of $\mathbf{U}_j = \mathbf{D}\mathbf{X}_j$ with all P_i, and assign $\mathbf{X}_j$ to the class with the closest distribution, such as

$$class(\mathbf{X}_j) = \arg\min_i d(P_i, P_{\mathbf{U}_j}). \tag{15}$$

We use the total variation distance in ((15), as defined in [17]

$$d(p,q) = \|p - q\|_{TV} = \frac{1}{2}\sum_{x\in\Omega} |p(x) - q(x)|. \tag{16}$$

In this experiment, the classification rate is 94.78%. The performance is higher than known state of the art methods based on predesigned features, such as [18] with a 86.63% classification rate, and comparable to the supervised dictionary learning algorithm [19]. It is worth noting that, the training set is only composed of around 1% of the dataset. A less stringent training set implies a lower computational cost. This also demonstrates the scalability of our method, in light of the competitive classification performance.

5. CONCLUSION

We have proposed in this paper, a novel approach for sparse null space problem, and have proved the equivalence of the sparse null space problem and the analysis dictionary learning. We have presented SNS-BP as an iterative algorithm based on l_1 minimization, to pursue the solution of the SNS problem. We have further applied this algorithm to analysis dictionary learning, and show the efficacy of our approach by experiments on both synthetic dataset and real-world data in texture classification.

Future work may include exploring the potential application of ADL on other high dimensional database, such as image/video classification.

6. REFERENCES

[1] Michael Elad, Mario AT Figueiredo, and Yi Ma, "On the role of sparse and redundant representations in image processing," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 972–982, 2010.

[2] Michael Elad, "Sparse and redundant representation modeling: What next?," *IEEE Signal Processing Letters*, vol. 19, pp. 922–928, Dec. 2012.

[3] Ron Rubinstein, Tomer Faktor, and Michael Elad, "K-svd dictionary-learning for the analysis sparse model," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 5405–5408.

[4] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009, pp. 689–696.

[5] Sam T Roweis and Lawrence K Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[6] David L Donoho and Carrie Grimes, "Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data," *Proceedings of the National Academy of Sciences*, vol. 100, no. 10, pp. 5591–5596, 2003.

[7] Ehsan Elhamifar and René Vidal, "Sparse subspace clustering," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 2790–2797.

[8] Xiao Bian and Hamid Krim, "Robust Subspace Recovery via Bi-Sparsity Pursuit," *ArXiv e-prints*, Mar. 2014.

[9] Xiao Bian and Hamid Krim, "Optimal operator space pursuit: a framework for video sequence data analysis," in *Computer Vision–ACCV 2012*, pp. 760–769. Springer, 2013.

[10] Thomas F Coleman and Alex Pothen, "The null space problem i. complexity," *SIAM Journal on Algebraic Discrete Methods*, vol. 7, no. 4, pp. 527–537, 1986.

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

[12] Emmanuel J. Candès, Justin K. Romberg, and Terence Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information.," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[13] Emmanuel J Candès, Justin K Romberg, and Terence Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on pure and applied mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[14] Zhouchen Lin, Minming Chen, and Yi Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," *arXiv preprint arXiv:1009.5055*, 2010.

[15] Zhouchen Lin, Risheng Liu, and Zhixun Su, "Linearized alternating direction method with adaptive penalty for low-rank representation," in *Advances in Neural Information Processing Systems*, 2011, pp. 612–620.

[16] Trygve Randen and John Hakon Husoy, "Filtering for texture classification: A comparative study," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 21, no. 4, pp. 291–310, 1999.

[17] David Asher Levin, Yuval Peres, and Elizabeth Lee Wilmer, *Markov chains and mixing times*, AMS Bookstore, 2009.

[18] Gehong Zhao, Guangmin Wu, Yue Liu, and Janming Chen, "Texture classification based on completed modeling of local binary pattern," in *Computational and Information Sciences (ICCIS), 2011 International Conference on*. IEEE, 2011, pp. 268–271.

[19] Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman, "Supervised dictionary learning," in *NIPS*, 2008, pp. 1033–1040.