SIAM J. Imaging Sciences
Vol. 9, No. 3, pp. 1107–1126

Sparsity and Nullity: Paradigms for Analysis Dictionary Learning*

Xiao Bian†, Hamid Krim†, Alex Bronstein‡, and Liyi Dai§

Abstract. Sparse models in dictionary learning have been successfully applied in a wide variety of machine learning and computer vision problems, and as a result have recently attracted increased research interest. Another interesting related problem based on linear equality constraints, namely the sparse null space (SNS) problem, 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 (ADL) 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 (SNS-BP) and extend it to a solution of ADL. Experimental results on numerical synthetic data and real-world data are further presented to validate the performance of our method.

Key words. dictionary learning, sparse coding, sparse null space problem, union of subspaces

AMS subject classifications. 65F10, 90C25

DOI. 10.1137/15M1030376

1. Introduction.

High-dimensional data analysis has been the focus of research in diverse areas, including machine learning, computer vision, and applied mathematics, on account of its theoretical complexity and great relevance to big data problems. Dictionary learning has been one of the key methodologies in addressing high-dimensional data, and has successfully been applied in feature extraction [15], signal denoising [15], [13], [32], [36], image construction [19], pattern recognition and classification [26], etc.

Specifically, a set of atoms learned from a given dataset are considered as a dictionary and are expected to have the potential to analyze unknown incoming data. In order to construct an effective dictionary, signal models play a key role. One common assumption is that high-dimensional data is concentrated in a low-dimensional manifold embedded in a high-dimensional space. Dimension reduction was hence a natural way to characterize the data, and was subsequently extended to a larger family of algorithms, and often referred to as nonlinear dimension reduction [31], [12]. With a different perspective on the same issue, recent research has shown that sparse models may also be very useful for learning discriminating and robust dictionaries from data (see [13] and references therein). While these were broadly

*Received by the editors July 13, 2015; accepted for publication (in revised form) May 16, 2016; published electronically August 9, 2016. The research of the first, second, and fourth authors was supported in part by U.S. Army Research Office under agreements W911NF-04-D-0003 and W911NF-16-2-0005.
http://www.siam.org/journals/siims/9-3/M103037.html

†Department of Electrical Engineering, North Carolina State University, Raleigh, NC 27606 (xbian@ncsu.edu, ahk@ncsu.edu). The research of these authors was supported in part by MDA.

‡School of Electrical Engineering, Tel Aviv University, Tel Aviv 69978, Israel (bron@eng.tau.ac.il). The research of this author was supported by the ERC StG 335491 (RAPID).

§Computing Sciences Division, U.S. Army Research Office, Research Triangle Park, NC 27703 (liyi.dai.civ@mail.mil).

applicable, a particularly well adapted structure of data, the so-called union of subspaces (UoS), revealed the power of such approaches. In particular, using a parsimony constraint on the number of atoms to represent the data at hand helps effectively recover the underlying basis of each subspace [17], [3]. In dictionary learning, *synthesis* and *analysis* models were proposed, and their respective strengths and limitations in obtaining sparse representations of data are also of interest. On the one hand, synthesis models seek a synthesis dictionary $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_n]$ such that $\mathbf{x}_i = \sum_{j \in S} \mathbf{d}_j w_{ij}, \|W\|_0 \leq k$, where $\mathbf{x}_i$ is the data of interest, $\mathbf{d}_j$ is the jth atom in the dictionary, and w_{ij} the associated coefficient. On the other hand, an operator H is sought in an analysis model, so that $H \circ \mathbf{x}_i$ yield a sparse coefficient vector for representing $\mathbf{x}_i$, where "$\circ$" represents a linear operation such as matrix-vector multiplication or convolution [28], [2], [32], [9], [34].

Another interesting and seemingly unrelated problem invoking sparsity is the sparse null space (SNS) problem, first proposed in 1986 by Coleman and Pothen [10]. As we elaborate further in this paper, the SNS problem may be stated as searching for a sparse basis for the null space of a given matrix $\mathbf{A}$. We demonstrate that the SNS solution is instrumental in helping understand the analysis dictionary learning problem, and in providing sufficient insight into achieving systematic and applicable solutions.

In this paper, we examine the relation between the SNS problem and the dictionary learning problem, and we prove that the SNS problem is equivalent to the analysis dictionary learning (ADL) problem. We then proceed to solve the ADL problem by building on the SNS problem results. Specifically, inspired by these existing results as well as state-of-the-art sparsity pursuit algorithms, we present an l_1 minimization-based greedy algorithm to solve the SNS problem. In contrast to current mainstream algorithms [13], [32], [26], [21], 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.

Our contribution is primarily twofold: on the one hand, we show, to the best of our knowledge, for the first time the connection between ADL and SNS, and further exploit an SNS-inspired approach to solve the ADL problem; on the other hand, we propose a novel sparsity pursuit algorithm for solving the SNS/ADL problem and show its efficacy on the subspace clustering problem. Note that in a short conference paper [4], we have shown the preliminary results of our work. In this paper, we provide a more detailed analysis of ADL and SNS, including the proof of the exact recovery condition of our proposed algorithm SNS-BP for solving SNS. Additionally, we further show comprehensive experimental results compared to the state-of-the-art methods such as analysis k-SVD [33].

The remainder of this paper is organized as follows. In section 2, we discuss the current state of the art in the area of dictionary learning, specifically in ADL. In section 3, we analyze the relationship between the SNS problem and ADL, and show their equivalence. In section 4, we present an effective method to solve the SNS problem which essentially also efficiently solves the ADL problem. Finally, in section 5 we validate our method on the ADL problem by numerical experiments, and illustrate the effectiveness of our algorithm on texture classification.

1.1. Notation. The notational conventions used throughout this paper are as follows: For an $m \times n$ matrix $\mathbf{X}$, the space spanned by its rows is denoted as row($\mathbf{X}$), and that spanned by its columns col($\mathbf{X}$). Its null space is denoted by null($\mathbf{X}$), and the direct sum of two subspaces is denoted by $\oplus$. The sparsity of $\mathbf{X}$, defined as $\frac{\|\mathbf{X}\|_0}{mn}$, is denoted by $\rho(\mathbf{X})$. Moreover, we denote by $P_{\mathbf{X}}$ the projection matrix onto col($\mathbf{X}$), and by $P_{\mathbf{X}^\perp} = \mathbf{I} - P_{\mathbf{X}}^T P_{\mathbf{X}}$ the projection matrix onto null($\mathbf{X}$). Additionally, given a vector $\mathbf{y} \in R^n$, operator $(\cdot)_j$ returns the value of the jth element of $\mathbf{y}$. The adjoint operator of $(\cdot)_j$, denoted as $(\cdot)_j^*$, is hence $(c)_j^* = \mathbf{v} \in R^n$ such that $(\mathbf{v})_j = c$ and $(\mathbf{v})_i = 0$, if $i \neq j$. Finally, we denote the set of all sparse vectors $\{\mathbf{x} | \|\mathbf{x}\|_0 \leq k\}$ by Σ_k.

2. Related work.

2.1. Sparse null space problem. The SNS problem has been proved to be an NP-hard problem in [10]. Different algorithms have since been proposed to find approximate solutions of SNS. For the sake of simplicity, we formulate SNS as finding sparse null vectors of $\mathbf{A}$.

In [11] the authors have proposed two algorithms for the SNS problem. The algorithms have two phases: First, a minimal dependent set of columns is found by bipartite graph matching; second, nonzero entries in the null vector are computed accordingly. When applied to the ADL problem, the computational cost of the algorithm is $\mathcal{O}(d^3)$, where d is the dimension of the row vectors of matrix $\mathbf{A}$. Additionally, the algorithm does not guarantee the reconstruction of the sparse null vectors, which is evident from the experimental evaluation presented in [11].

In [18] the authors outlined several families of SNS algorithms, both combinatorial and noncombinatorial in nature. Typically, the combinatorial structure of the matrix is used to guide the search for sparse null vectors, and noncombinatorial numerical methods are used to decide linear dependence. An important open question raised in this paper is the trade-off between conditioning and sparsity. In [29] SNS algorithms for several types of matrices arising in structural optimization have been studied. Since most of the SNS algorithms require certain structures of the null matrix such as orthogonality, the resulting null matrix may be much denser than an arbitrary null matrix of the same rank.

In contrast to these previous algorithms, the algorithm proposed in this paper does not make any assumption on the structure of $\mathbf{A}$ or that of its corresponding sparse null vectors. By segmenting the constraint domain into convex subregions, we naturally address the issue of linear dependency of the sparse null vectors. In fact, this approach has no constraint on the conditioning of the sparse null matrix and hence does not have the trade-off between conditioning and sparsity. Additionally, we improve the computational complexity to $\mathcal{O}(d^2)$, allowing us to apply it to ADL problems with data of moderate dimensions.

2.2. Analysis dictionary learning. Much of the research in dictionary learning has primarily focused on synthesis models such as [1], [26], [25], [21], among many others. Remarkable performance has been achieved in learning effective dictionaries which are well adapted to specific datasets, especially on imagery data. The contribution of the atoms in the representation of the training data was constrained to be sparse. The same sparseness is exploited to recover an input signal from corrupted data [1] or to classify signals into different clusters [20].

Specifically, the learned dictionary and the corresponding sparse coefficients are alternately

updated until convergence or until a target performance attainment. With the sparse coefficients in hand, learning each atom may be solved by the gradient descent or its variations such as the stochastic gradient descent [5]. With the dictionary atoms discovery, different approaches have been proposed to determine the sparse coefficients. In k-SVD [1], for example, orthogonal matching pursuit (OMP) is used to recover the sparse coefficient, while Lasso-LARS is chosen in online dictionary learning for sparse coding [26].

A well-known issue with synthesis dictionary learning is the poor stability in its signal representation [33], [14]. This is primarily due to the difficulty in controlling the coherence of the learned dictionary, which may, in turn, lead to multiple representations of the same signal [6]. While this phenomenon does not particularly adversely affect denoising [14], it may significantly impact a consistent classification or clustering performance. In addition, the computational cost to process new incoming data points often calls for a procedure such as a matching pursuit procedure [1], or a sparse coding routine such as Lasso [26], and is too significant for large high-dimensional datasets.

While ADL is known to be difficult to train, it is free of the above-mentioned SDL (synthesis dictionary learning) limitations [33]. Additionally, upon learning the dictionary $\mathbf{D}$, the representation of any data $\mathbf{x}$ is unique, as it is the result of one matrix-vector multiplication. The latter fact also yields a low computational cost of processing new data which is linear in both the data dimension and the sample size.

Analysis k-SVD [33] provides, to the best of our knowledge, the current state-of-the-art solution to the ADL problem. In the learning procedure, a framework similar to k-SVD is designed to alternately update the dictionary and the sparse coefficients. At each iteration, each atom is independently updated by minimizing the covariance between the atom and a subset of data samples that are "almost orthogonal" to the atom. The minimization may be formulated as searching for the singular vectors corresponding to the minimal singular value of the data matrix of the subset of samples. Analysis k-SVD generally achieves the best known performance in the recovery of the original data space.

Additionally, in [9] the ADL problem has been connected to filter-based MRF models such as the field of experts model. By casting ADL into MRF models, the analysis operator has no explicit constraint in the optimization problem, which simplifies the solution procedure. In [36], the assumption of a uniform normalized tight frame is imposed to constrain the ADL problem and avoid the trivial solution.

In [19] the ADL problem is formulated as an optimization problem on the set of full-rank matrices with normalized columns. Specifically, the sparsity of output features and the variance of sparsity of output features are both minimized under the constraint that the analysis operator is of full rank and has no linear dependent rows. The resulting highly nonconvex optimization problem is solved via the geometric conjugate gradient method that is claimed to be efficient.

As a departure from analysis k-SVD, we present in this paper an alternative ADL algorithm which is closely related to the SNS problem discussed next. By decomposing the DL problem into a set of convex optimization subproblems, we can guarantee the convergence of our proposed algorithm. Additionally, and in contrast to analysis k-SVD, our proposed method can naturally adapt to represent data of varying underlying sparsity with no prior knowledge.

3. From sparse null space to analysis dictionary learning. In this section, we reformulate the SNS and ADL problems in matrix form, and then establish their equivalence.

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

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

Let $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ be a generic data matrix; the ADL problem can then be formulated as

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

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

It should be noted that one of the first formulations of ADL discussed in [35] suggested to solve

$$\min_{\mathbf{D}} \|\mathbf{DX}\|_1 \quad \text{s.t. } \mathbf{D} \in \mathcal{F}, \tag{3}$$

where $\mathcal{F}$ is a uniform normalized tight frame. The primary purpose of the latter constraint is to avoid the trivial solutions $\mathbf{D} = 0$. Our formulation is free of this rather arbitrary constraint.

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

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

The proof of Theorem 1 is straightforward. Specifically, assume $\mathbf{N}$ is a minimizer of (1); then, the constraints in (1) ensure that $\text{null}(\mathbf{A}) = \text{col}(\mathbf{N})$. Since $\text{col}(\mathbf{X}^T) = \text{row}(\mathbf{X}) = \text{null}(\mathbf{A})$, we have

$$\text{col}(\mathbf{X}^T) = \text{col}(\mathbf{N}). \tag{4}$$

We next consider any minimizer $\mathbf{U}$ of problem (2); note that $\mathbf{D}$ is fully determined from $\mathbf{U}$. Since $\text{row}(\mathbf{X}) = \text{row}(\mathbf{U})$, when combined with the condition $\text{row}(\mathbf{X}) = \text{null}(\mathbf{A})$, we have $\text{row}(\mathbf{U}) = \text{row}(\mathbf{X}) = \text{row}(\mathbf{N}^T)$. $\mathbf{N}^T$ is therefore a feasible solution of problem (2), and so is $\mathbf{U}^T$ of problem (1). It follows that $\|\mathbf{U}\|_0 = \|\mathbf{N}^T\|_0$, and hence $\mathbf{N}^T$ is also a minimizer of (2) and $\mathbf{U}^T$ is a minimizer of (1).

This essentially tells us that we can solve the ADL problem, should we have access to an effective method of solving the corresponding SNS problem. Specifically, given a data matrix $\mathbf{X}$, the analysis dictionary for $\mathbf{X}$ may be constructed in the following three steps:

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

4. An iterative sparse null space pursuit. We have discussed the relation of SNS and ADL in section 3, and have shown that they may be cast in one unified framework of SNS 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 that it is still possible to approximate the SNS basis in polynomial time. In this section, we propose an l_1-based iterative optimization method for an SNS pursuit.

4.1. A greedy algorithm for the sparse null space problem. Previous works on the SNS problem have shed some light on a polynomial time solution. In [10], the authors proposed a greedy algorithm for the SNS problem. For the sake of clarity and of further discussion, we refer to it as Algorithm 1. Additionally, it has been proved in [10] that Algorithm 1 can be used to construct an SNS basis, as stated in our Theorem 2 (see [10]).

Algorithm 1 A greedy algorithm for the SNS problem.

Initialize: matrix $\mathbf{A} \in R^{m\times d}$, $\mathbf{N} = \emptyset$
for i = 1,. . . ,q **do**
  Find a 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 amounts 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 an NP-hard problem [10]. We therefore next focus on finding a method to solve this subproblem in each iteration of Algorithm 1.

4.2. l_1-based search for sparse null space. We first reformulate the subproblem of finding a sparsest null vector in Algorithm 1 as follows:

$$\min_{\mathbf{n}_i} \|\mathbf{n}_i\|_0 \quad \text{s.t. } \mathbf{A}\mathbf{n}_i = 0, P_{\mathbf{N}_{i-1}^{\perp}}\mathbf{n}_i \neq 0, \tag{5}$$

where $\mathbf{N}_{i-1}$ is the subspace spanned by the thus constructed null space vectors at the previous $(i-1)$th iteration. The condition $P_{\mathbf{N}_{i-1}^{\perp}}\mathbf{n}_i \neq 0$ 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, and hence is the reason for the NP-hardness of the problem. Second, the constraint in (5),

$$P_{\mathbf{N}_i^{\perp}} \mathbf{n}_i \neq 0, \tag{6}$$

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 [7], [8]. To address the second problem, and to hence obtain a convex and compact feasible region, we propose to restate the condition $P_{\mathbf{N}_i^{\perp}} \mathbf{n}_i \neq 0$ as follows:

$$\exists j \in \{1, \ldots, d\}, (P_{\mathbf{N}_{i-1}^{\perp}} \mathbf{n}_i)_j = c, \tag{7}$$

where c can be any positive constant.

Additionally, we establish the following lemma to justify the variation on the constraint from (6) to (7).

Lemma 3. *The solution of problem* (5) *remains invariant if the constraint* (6) *is substituted by constraint* (7).

The proof of Lemma 3 is presented in Appendix A.

The meaning of Lemma 3 is that we may then separate the region defined by (7) into compact and convex regions based on j, i.e., the location of the forced nonzero element. Since the optimal solution must reside in one of these regions, we may search for the sparsest null vector in each region from $j = 1$ to d. We subsequently have a convex formulation of l_1 minimization for each j. Algorithm 1 may then be specifically realized as Algorithm 2.

Algorithm 2 SNS 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:

$$\begin{aligned} &\min_{\mathbf{n}} \|\mathbf{n}\|_1 \\ &\text{s.t. } \mathbf{A}\mathbf{n} = 0, (P_{\mathbf{N}^{\perp}} \mathbf{n})_j = c. \end{aligned} \tag{8}$$

It is worth noting that the exact recovery of each $\mathbf{n}$ via (8) is determined by the incoherence of the linear operator defined by the two constraints and the sparsity of each $\mathbf{n}$, which we will

discuss in more detail in section 4.3. To solve (8), we adopt the framework of the augmented Lagrange method (ALM) on account of its good performance in matrix-norm minimization problems [23], [24]. Specifically, we have the augmented Lagrange function of (8) as

$$L(\mathbf{n}, \mathbf{Y}_1, \mathbf{Y}_2, \mu) = \|\mathbf{n}\|_1 + \langle \mathbf{Y}_1, \mathbf{An}\rangle + \langle \mathbf{Y}_2, (P_{\mathbf{N}^\perp}\mathbf{n})_j - c\rangle + \frac{\mu}{2}\|\mathbf{An}\|^2 + \frac{\mu}{2}\|(P_{\mathbf{N}^\perp}\mathbf{n})_j - c\|^2. \tag{9}$$

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

$$\mathbf{n}_{k+1} = \mathcal{T}_{\frac{1}{\mu_k \eta}}\left(\mathbf{n}_k - \frac{\mathbf{n}_k^1 + \mathbf{n}_k^2}{\eta}\right), \tag{10}$$

where $\mathcal{T}$ is the soft-thresholding operator, and $\|\eta\|^2 \geq \|\mathbf{A}\|^2 + \|P_{\mathbf{N}^\perp}\|^2$, [1] and

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

$$\mathbf{n}_k^2 = P_{\mathbf{N}^\perp}\left((P_{\mathbf{N}^\perp}\mathbf{n})_j - c + \frac{\mathbf{Y}_2^k}{\mu_k}\right)^*_j. \tag{12}$$

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

$$\mathbf{Y}_1^{k+1} = \mathbf{Y}_1^k + \mu_k\left(\mathbf{An}_{k+1}\right), \tag{13}$$

$$\mathbf{Y}_1^{k+1} = \mathbf{Y}_1^k + \mu_k\left((P_{\mathbf{N}^\perp}\mathbf{n})_j - c\right), \tag{14}$$

$$\mu_{k+1} = \min\{\rho\mu_k, \mu_{max}\}. \tag{15}$$

The strategy of the linearized ALM method provides a fast convergence rate [24]. This effectively provides us a method (Algorithm 2), named sparse null space basis pursuit (SNS-BP) in this paper, to efficiently solve the SNS problem.

4.3. Exact recovery of the sparse null space using sparse null space basis pursuit. In (8), we consider the l_1 minimization problem as a convex relaxation of the l_0 norm to find sparse null vectors of $\mathbf{A}$. For the sake of simplicity, we rewrite (8) as follows:

$$\min_{\mathbf{n}} \|\mathbf{n}\|_1 \quad \text{s.t. } [\mathbf{A}^T\ \mathbf{p}]^T\mathbf{n} = [\mathbf{0}^T\ c]^T, \tag{16}$$

where $\mathbf{p}^T\mathbf{n} = (P_{\mathbf{N}^\perp}\mathbf{n})_j = c$. We further denote $\mathbf{B} = [\mathbf{A}^T\ \mathbf{p}]^T$.

The exact recovery of a sparse solution by solving an l_1 minimization problem has been well studied in the literature. In particular, one sufficient condition of the exact recovery requires the linear constraint $\mathbf{B}$ of the l_1 minimization problem to satisfy the restricted isometry

[1] The value of η is selected in this way to ensure the convergence of the algorithm. The details are discussed in [24].

property of order $2k$ with $\delta_{2k} < \sqrt{2}-1$ (Theorem 1.2 in [6]). The restricted isometry property (RIP) of order k for a matrix $\mathbf{B}$ [16] is defined as

$$(1-\delta_k)\|\mathbf{n}\|_2^2 \leq \|\mathbf{B}\mathbf{n}\|_2^2 \leq (1+\delta_k)\|\mathbf{n}\|_2^2 \tag{17}$$

for every k-sparse vector $\mathbf{n}$.

A first look at the constraint of (16) tells us that, if $\mathbf{n}$ is k-sparse, $\mathbf{B}$ does not satisfy RIP of order $2k$ with $\delta_{2k} < \sqrt{2}-1$. Specifically, assume that the sparse null vector found in the previous iteration of Algorithm 2 is $\mathbf{n}_{i-1}$; then we have $\mathbf{B}\mathbf{n}_{i-1} = 0$ since $\mathbf{n}_{i-1}$ is in the null space of $\mathbf{A}$ and $\mathbf{n}_{i-1}$ is orthogonal to $\mathbf{N}^{\perp}$. Given the fact that $\mathbf{n}_{i-1}$ is k-sparse, $\mathbf{B}$ fails to satisfy the RIP of order k as $0 = \|\mathbf{B}\mathbf{n}_{i-1}\|_2^2 < (1-\delta)\|\mathbf{n}_{i-1}\|_2^2$ for any $\delta < 1$.

However, if we further check the matrix $\mathbf{B} = [\mathbf{A}^T\ \mathbf{p}]^T$, its specific structure shows that not every k-sparse vector is a feasible solution. In particular, at iteration i, only k-sparse vectors $\mathbf{n}$ satisfying $\mathbf{n} \in \text{null}(\mathbf{A}) \cap \mathbf{N}_i^{\perp}$ are feasible solutions. At each iteration, we therefore need only to satisfy with $\delta < \sqrt{2}-1$ the constrained version of RIP defined next, and using similar arguments in [8] for recovery.

Definition 4. *$\mathbf{B}$ satisfies the constrained RIP of order k with δ if for any $\mathbf{n}, \mathbf{n}' \in \text{null}(\mathbf{A}) \cap \mathbf{N}_i^{\perp} \cap \Sigma_k$,*

$$(1-\delta)\|\mathbf{n}-\mathbf{n}'\|_2^2 \leq \|\mathbf{B}(\mathbf{n}-\mathbf{n}')\|_2^2 \leq (1+\delta)\|\mathbf{n}-\mathbf{n}'\|_2^2. \tag{18}$$

4.4. Solving the analysis dictionary learning problem via sparse null space basis pursuit.

In section 3, we discussed the equivalence of the ADL problem and the SNS problem, and hence further describe the details of solving the ADL problem (as in (2)) via SNS-BP.

For a typical ADL problem as in (2), the first step, as discussed in section 3, is to construct a matrix $\mathbf{A}$ whose transpose is the null space of $\mathbf{X}$. Concretely, we have the following problem.

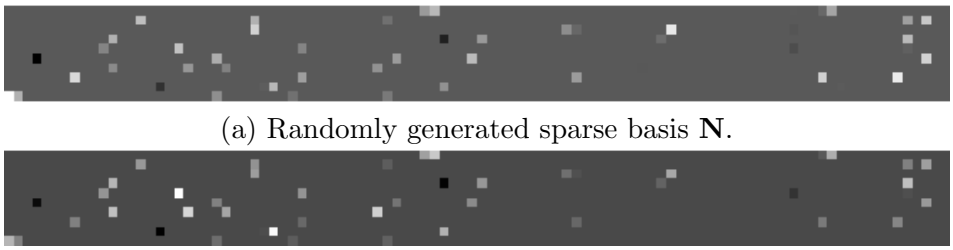
Problem 1. *Find $\mathbf{A}$ such that $\mathbf{X}\mathbf{A}^T = 0$.*

A simple way would be to consider a singular value decomposition of $\mathbf{X}$, and keep the right singular vectors with zero singular values coinciding with the rows of $\mathbf{A}$. Note that at the training stage, we need the training data to be clean and admit the union of subspaces model. For a common scenario where the data matrix $\mathbf{X}$ is contaminated by Gaussian noise, we can set $\mathbf{A}$ to the right singular vectors with small singular values, instead of exactly zero. This in fact offers an additional advantage of filtering out dense Gaussian noise from the data matrix $\mathbf{X}$.

Upon constructing $\mathbf{A}$, we proceed to obtain the sparse coefficient matrix $\mathbf{U}^T = SNS(\mathbf{A})$. Note that $\mathbf{U}$ is computed independently of the analysis operator $\mathbf{D}$. In case $\mathbf{D}$ is required for further processing of incoming data, using $\mathbf{D}\mathbf{X} = \mathbf{U}$, we may easily obtain $\mathbf{D} = \mathbf{U}\mathbf{X}^{\dagger}$, where $\mathbf{X}^{\dagger}$ is the pseudo-inverse of $\mathbf{X}$. In particular, if all entries of the dataset are independent, i.e., $\mathbf{X}$ is full row rank, then $\mathbf{X}^{\dagger} = \mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}$.

While we formulated the ADL problem with the constraint $\text{row}(\mathbf{U}) = \text{row}(\mathbf{X})$ in section 3, if a more compact representation of $\mathbf{X}$ is preferred, we may opt for $\text{row}(\mathbf{U}) \subset \text{row}(\mathbf{X})$, hence further reducing the dimension of the original data space.

It is worth noting that in the above approach, we choose not to further normalize the learned dictionary $\mathbf{D}$, i.e., to have each atom as unit-length. Instead, we normalize each data



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

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

Figure 1. *An example of the result of SNS-BP.*

sample to unit-length and the l_∞ norm of each column of the sparse coefficient matrix $\mathbf{U}$. The norm of $\mathbf{D}$ is automatically constrained by $\mathbf{D} = \mathbf{U}\mathbf{X}^\dagger$. Since the sparse coefficients are the features needed for representing the original data, we directly apply the norm constraint on sparse coefficients instead of the learned dictionary.

The computational complexity of SNS-BP is determined by the inner loop of Algorithm 2 as $\mathcal{O}(pd^2)$, where p is the dimension of the null space and d is the dimension of data. In particular, all d locations of the possible nonzero elements need to be calculated, and each of these subproblems is of $\mathcal{O}(d)$.

5. Numerical experiments on sparse null space and analysis dictionary learning. For a quantitative evaluation of our algorithm, we synthesize data that are 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 and the sparse coefficients of the ADL problem.

First, we synthesize a $d \times p$ sparse matrix $\mathbf{N}$ as the SNS basis of some matrix $\mathbf{A}$, where $\mathbf{A}$ may be constructed by exploiting the SVDS of $\mathbf{N}$, and by selecting its left singular vectors corresponding to the zero singular values as the rows of $\mathbf{A}$, i.e., $\mathbf{A}\mathbf{N} = \mathbf{0}$.

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 Figure 1, we show one example of exact recovery of a sparse null space basis up to permutation and scale.

In Figure 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 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 Figure 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 $\rho(\mathbf{N})$ varying from 0.01 to 0.2. Additionally, we define the relative error of $\hat{\mathbf{N}}$ as

$$err(\hat{\mathbf{N}}) = \frac{\|\hat{\mathbf{N}}\mathbf{P}\mathbf{\Gamma} - \mathbf{N}\|_F}{\|\mathbf{N}\|_F}, \tag{19}$$

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 $\mathbf{N}$ varying from 0.01 to 0.2 is 1.69%.

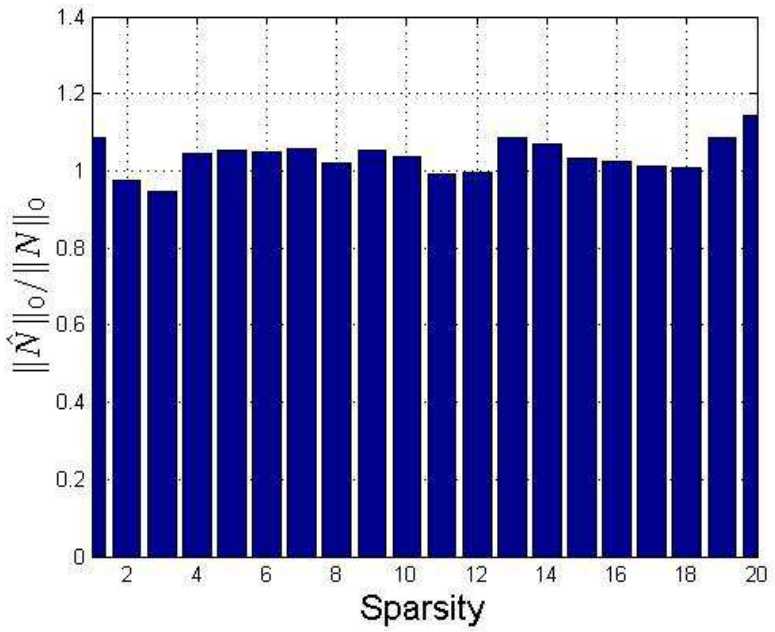
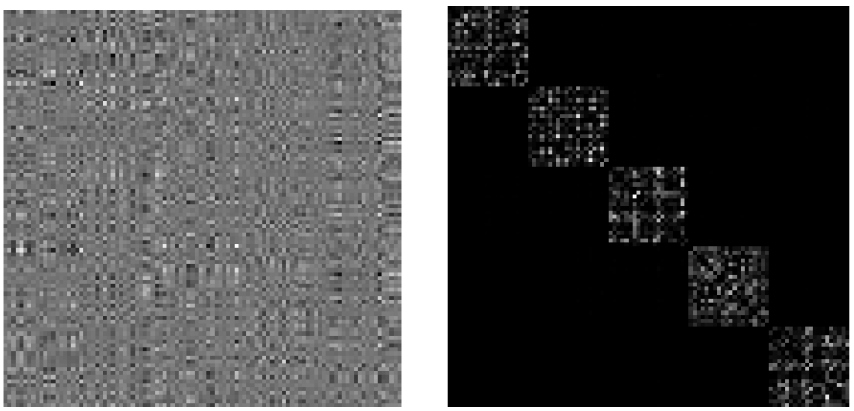


Figure 2. $\|\hat{\mathbf{N}}\|_0/\|\mathbf{N}\|_0$ *versus sparsity.*



(a) Synthetic data matrix $\mathbf{X}$. (b) Sparse coefficient matrix $\mathbf{W}$.

Figure 3. *Sample synthetic data matrix and its underlying structure.*

We next test the ADL via SNS-BP by exploring data samples with hidden underlying sparse structures. In particular, data samples are randomly selected from a union of low-dimensional subspaces $S = S_1 \cup S_2 \cdots$, in which each subspace is also randomly constructed by using the orthogonal basis of a set of uniformly distributed vectors. Under this setting, each sample can be represented as a linear combination of other samples in the same subspace. The dataset, written as a matrix $\mathbf{X} = [\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}]$ as shown in Figure 3(a), has a sparse intrinsic structure W such that $\mathbf{X} = \mathbf{XW}$, where W is a block-diagonal matrix as Figure 3(b), and each block represents one subspace. In our experiment, we have data points distributed

(a) The sparse coefficient matrix $\mathbf{U}$ by ADL using SNS-BP.

(b) Permute rows of $\mathbf{U}$ to show the structure of $\mathbf{X}$.

Figure 4. *Sparse coefficients using ADL.*

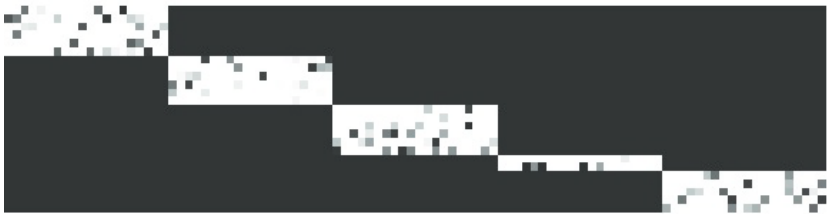
in five three-dimensional subspaces within the ambient space R^{100}. It hence implies that the null space of $\mathbf{A}$ constructed from $\mathbf{X}$ is of dimension 15. An analysis dictionary $\mathbf{D}$ is then trained using SNS-BP, and the associated sparse coefficient matrix $\mathbf{U}$ is obtained as shown in Figure 4(a). Specifically, we can see that all nonzero entries of each sparse vector are clustered together, corresponding to data samples from the same subspace. In other words, for an atom d_i in the analysis dictionary $\mathbf{D}$, all the data points that have a significant response to it are from one subspace, and the rest of the data have zero response.

Next, we cluster the rows of $\mathbf{U}$ and permute them accordingly, as presented in Figure 4(b). It is interesting to see that a more compact block-diagonal structure emerges again. Note that in this example we find in totality 15 atoms, with each of three atom sets supporting the data samples (have largely nonzero inner product) in a subspace. This number corresponds to the intrinsic dimension of each subspace. Having trained $\mathbf{D}$, it is subsequently simple to figure out which subspace a data point belongs to: we may simply separate $\mathbf{D}$ into $\mathbf{D} = [\mathbf{D}_1, \mathbf{D}_2, \ldots]$, with $\mathbf{D}_i$ being the set of atoms supporting the ith subspace, and ultimately determine the maximal $\|\mathbf{D}_i\mathbf{x}\|$. It is therefore more efficient to recover the underlying structure of a given dataset and represent this structure in a more compact way.

5.1. The relation to analysis k-SVD. Analysis k-SVD arguably provides the state-of-the-art solution for the ADL problem and has achieved a much improved performance in signal denoising by discovering the underlying UoS structure of a wide class of signals [33]. We next comparatively evaluate the performance of SNS-BP and analysis k-SVD in the recovery problem of UoS of synthesized data.

The synthesis of data from a UoS is similar to the setting in section 5. In particular, data samples are randomly chosen from a union of low-dimensional subspaces $S = S_1 \cup S_2 \cdots$, in which each subspace is also randomly constructed by using the orthogonal basis of a set of uniformly distributed vectors. In generating our data, we pick an ambient space of dimension 100, and collectively five subspaces. We evaluate the performance of SNS-BP and analysis k-SVD by varying the intrinsic dimensions of each subspace. Interestingly, SNS-BP shows consistent performance with the increase of the intrinsic dimension of each subspace, while the performance of analysis k-SVD deteriorates when the intrinsic dimension of each subspace increases.

(a) The sparse coefficient matrix $\mathbf{U}$ by ADL using SNS-BP.

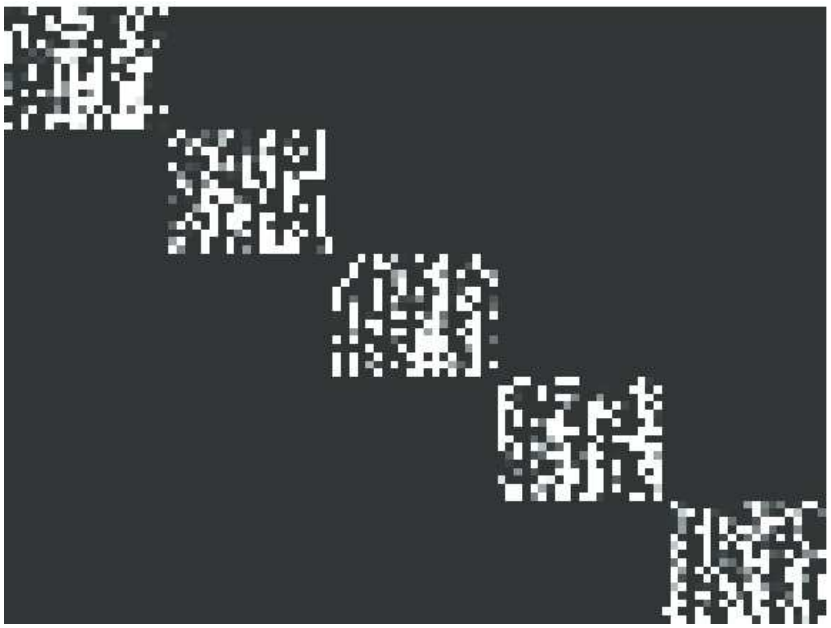


(b) The sparse coefficient matrix $\mathbf{U}$ by ADL using analysis k-SVD.

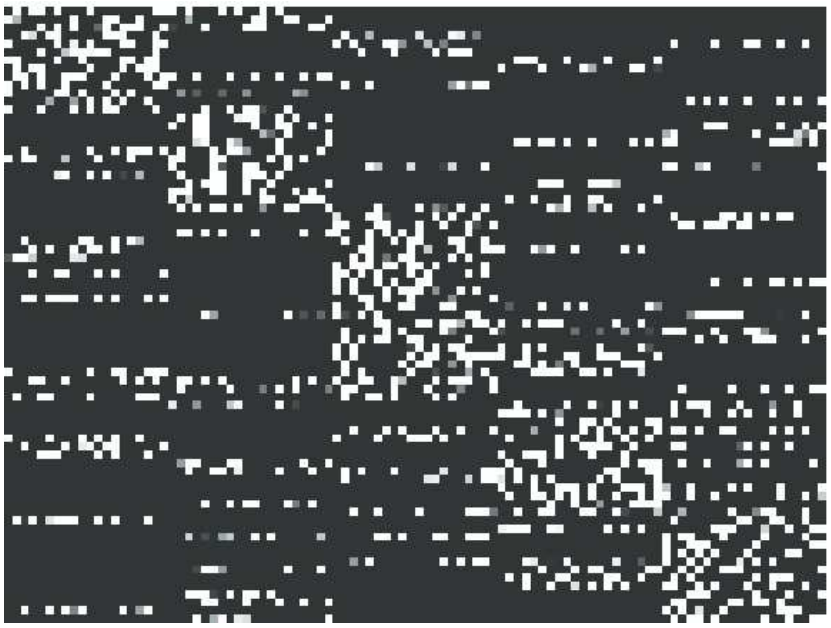
Figure 5. *Sparse coefficients for five subspaces with intrinsic dimension* 5.

As shown in Figure 5, when the intrinsic dimension is low, both SNS-BP and analysis k-SVD can effectively learn atoms from the dataset which yield sparse coefficients to efficiently reflect the underlying subspace of each data point. More precisely, each atom has a strong response on data in only one subspace and is rather absent in other subspaces. Five blocks corresponding to five subspaces appear in the sparse coefficient matrix using both approaches. SNS-BP exhibits a clearly improved performance over analysis k-SVD in recovering the intrinsic dimension of each subspace, as shown in Figure 5. This may be understood by recalling that the algorithm SNS-BP ensures that the sparse vector determined at each iteration is orthogonal to the subspace spanned by the previously found sparse vectors (i.e., truly novel). Equivalently, the rows in Figure 5(a) are linearly independent—a property which provides an accurate estimate of the intrinsic dimension of each subspace. In contrast, since analysis k-SVD can possibly find collinear sparse vectors, it yields more sparse vectors than the dimension of a given subspace, and hence may partially miss another subspace.

As the intrinsic dimension increases, analysis k-SVD begins to learn atoms which span two or more subspaces. Such cases become more prevalent as the intrinsic dimension of each subspace increases. In Figure 6, for instance, the five coefficient blocks representing the five subspaces merge and display significant responses from two or more subspaces. SNS-BP, in contrast, exhibits a more consistent performance even as the intrinsic dimension of each subspace increases, thus resulting in recovering a correct dimension for each subspace. In Figure 7, we show a case of UoS with different randomly generated dimensions. It is worth noting that given the ambient space of R^{100}, we limit the intrinsic dimension of each subspace to between 1 and 19, to avoid the extreme case that when every subspace is of dimension 20, we may run into the situation where all samples are randomly chosen from $R^1 00$ and lose any nontrivial UoS structure. In this case, analysis k-SVD misses one subspace with the lowest dimension entirely after 45 atoms learned, and a nonnegligible portion of the dictionary fails to distinguish data from different subspaces. SNS-BP clearly shows five subspaces of the same dataset.

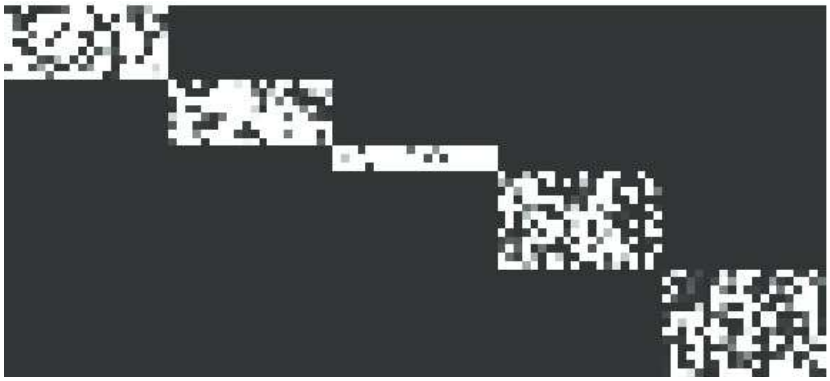


(a) The sparse coefficient matrix $\mathbf{U}$ by ADL using SNS-BP.



(b) The sparse coefficient matrix $\mathbf{U}$ by ADL using analysis k-SVD.

Figure 6. *Sparse coefficients for five subspaces with intrinsic dimension* 15.



(a) The sparse coefficient matrix $\mathbf{U}$ by ADL using SNS-BP.

(b) The sparse coefficient matrix $\mathbf{U}$ by ADL using analysis k-SVD.

Figure 7. *Sparse coefficients for five subspaces with intrinsic dimension randomly picked between* 1 *and* 19 *as* 9, 8, 3, 12, 13.

We further thoroughly test the performance of SNS-BP and analysis k-SVD on a dataset with intrinsic dimension from 5 to 16 using the measure of intra-cluster covariance ratio. For each dimension, we repeat the process of data generation 20 times to obtain the mean value of the intra-cluster correlation ratios for both methods.

We define the intra-cluster covariance ratio as

$$CV(\mathbf{U}) = \frac{\text{intra-cluster covariance of } \mathbf{U}}{\text{total covariance of } \mathbf{U}}. \tag{20}$$

In Figure 8, we can see that the CV of analysis k-SVD deteriorates from 1.00 to around 0.75 with the increase of the intrinsic dimension of each subspace, while SNS-BP maintains a consistently high performance throughout.

We further test the performance of SNS-BP and analysis k-SVD by randomly choosing the intrinsic dimension of each subspace between 1 and 19. In this experiment, the intrinsic dimension of each subspace can be different. We repeat the data generation 100 times to obtain mean CV values for both methods. The results show a higher CV value 1.00 of SNS-BP versus 0.97 of analysis k-SVD.

All these experiments in this part prove a more robust performance of SNS-BP compared

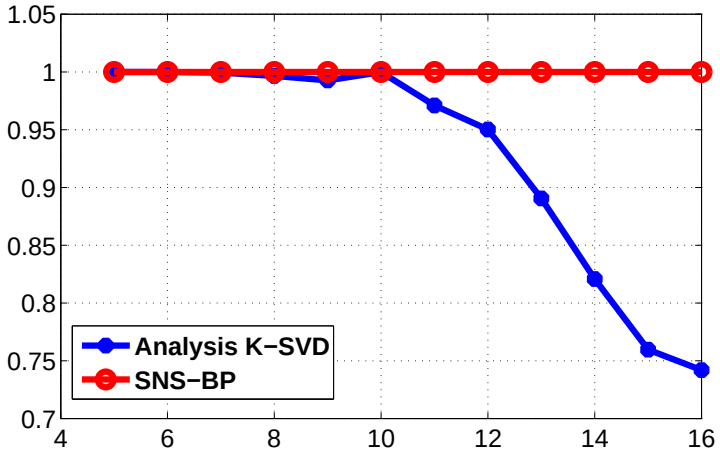


Figure 8. *Intra-cluster covariance ratio for SNS-BP and analysis k-SVD. The x-axis is the intrinsic dimension of each subspace, and the y-axis is the CV value.*

to analysis k-SVD when the intrinsic dimension of each subspace is higher. Moreover, SNS-BP demonstrates the ability to recover the intrinsic dimensionality of each underlying subspace, and automatically avoids redundant atoms.

5.2. Applications on real-world data. In this experimental section, we explore the inference potential of our method on images. The performance of our algorithm is evaluated on texture images from the Brodatz database [30]. Each texture image is partitioned into a set of patches, and their analysis operator is learned from patches of different textures. The latter is subsequently 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. For example, the texture with more randomness may lead to less sparsely structured coefficients and more coherent/correlated textures, i.e., on account of a broadly spread distribution of its patches, discovered upon applying the learned operator to incoming data. Note that we avoid the complexity of processing the order of patches by considering the distribution of the coefficients instead of matching the output vectors.

Specifically, we segment each texture image into 10×10 patches, and randomly pick a subset of patches as the training set from each texture image, and the rest of the patches are used as a testing set. The texture images from the Brodatz database [30], and the corresponding sample patches, are shown in Figure 9. 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 ith class of texture in the training set. In the next testing stage, texture images are used as a set of patches, for which the distribution of the coefficients $\mathbf{U}_j = \mathbf{D}\mathbf{X}_j$ with all P_i are compared. In particular, we assign $\mathbf{X}_j$ to the class with the closest distribution, such as

$$class(\mathbf{X}_j) = \arg\min_i d(P_i, P_{\mathbf{U}_j}). \tag{21}$$

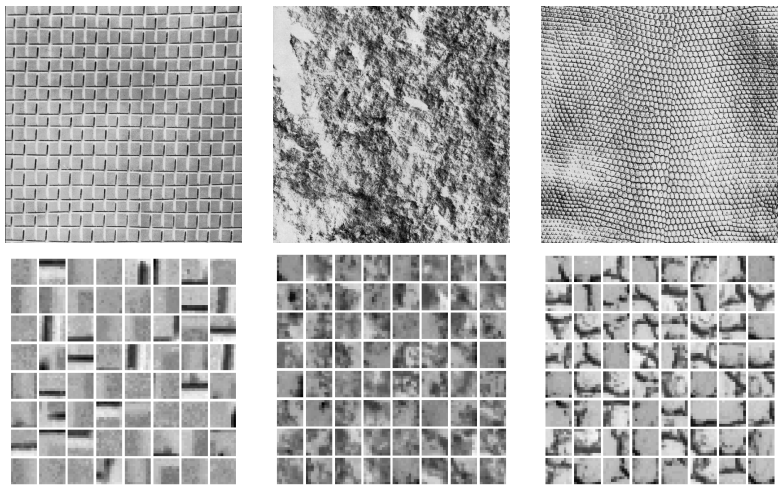


Figure 9. *Textures and the corresponding patches for training.*

We use the total variation distance in (21), as defined in [22],

$$d(p,q) = \|p-q\|_{TV} = \frac{1}{2}\sum_{x\in\Omega} |p(x)-q(x)|. \tag{22}$$

In this experiment, we first segment the entire texture image into 10×10 patches and randomly select 120 patches from each texture for training. The rest of the patches are treated as test data. In the test stage, we randomly select 120 patches from one texture, and then use (21) to determine the label of the test data. The classification rate is 94.78% for the texture images shown in Figure 9. The performance is higher than known state-of-the-art methods based on predesigned features, such as [37] with an 86.63% classification rate, and comparable to the supervised dictionary learning algorithm [27]. It is important to note that the training set in our case 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 its competitive classification performance.

6. Conclusion. We have proposed in this paper a novel approach for the sparse null space (SNS) problem, and have unveiled its equivalence to the analysis dictionary learning (ADL) problem. We have presented the sparse null space basis pursuit (SNS-BP), an iterative algorithm based on l_1 minimization, to pursue the solution of the SNS problem. We have further applied this algorithm to ADL, and showed the efficacy of our approach by experiments on both synthetic datasets and real-world data in texture classification.

Future work may include several aspects related to both SNS and ADL problems. The relation between SNS and nonlinear dimension reduction needs further investigation and may lead to results on graph embedding. Moreover, our future goal is to explore the potential application of ADL on other high-dimensional databases such as image/video classification.

Appendix A. Proof of Lemma 3. Lemma 3 states that the following optimization problems are equivalent:

$$\text{(A)} \qquad \begin{aligned} &\min_{\mathbf{n}} \|\mathbf{n}\|_0 \\ &\text{s.t. } \mathbf{A}\mathbf{n} = 0, P_{\mathbf{N}^\perp}\mathbf{n} \neq 0, \end{aligned}$$

$$\text{(B)} \qquad \begin{aligned} &\min_{\mathbf{n}} \|\mathbf{n}\|_0 \\ &\text{s.t. } \mathbf{A}\mathbf{n} = 0, \exists j \in \{1, \ldots, d\}, (P_{N^\perp}\mathbf{n})_j = c. \end{aligned}$$

Proof. First, we show that if $\mathbf{n}$ is an optimal solution of (A), then for some real number α, $\alpha\mathbf{n}$ is also a minimizer of (B).

Any minimizer $\mathbf{n}'$ of (B) also satisfies the constraints of (A). It hence follows that

$$\|\mathbf{n}'\|_0 \geq \|\mathbf{n}\|_0. \tag{23}$$

Assuming $|(P_{\mathbf{N}^\perp}\mathbf{n})_k| = \|P_{\mathbf{N}^\perp}\mathbf{n}\|_\infty$, and noting that $\|P_{\mathbf{N}^\perp}\mathbf{n}\|_\infty \neq 0$, we construct

$$\hat{\mathbf{n}} = \frac{c}{(P_{\mathbf{N}^\perp}\mathbf{n})_k} \cdot \mathbf{n} = \alpha\mathbf{n}. \tag{24}$$

Since $(P_{\mathbf{N}^\perp}\hat{\mathbf{n}})_k = c$, $\hat{\mathbf{n}}$ is also feasible in (B). Suppose that $\|\hat{\mathbf{n}}\|_0 = \|\mathbf{n}\|_0 \leq \|\mathbf{n}'\|_0$, and in combination with (23), we can conclude that $\hat{\mathbf{n}} = \alpha\mathbf{n}$ and is also a solution of (B), and therefore $\|\hat{\mathbf{n}}\|_0 = \|\mathbf{n}'\|_0$.

Then it is trivial to show that $\mathbf{n}'$ is also a minimizer of (A), given the fact that $\mathbf{n}'$ is a feasible solution of (A) and $\|\mathbf{n}'\|_0 = \|\hat{\mathbf{n}}\|_0 = \|\mathbf{n}\|_0$, thus proving Lemma 3. ∎

REFERENCES

[1] M. Aharon, M. Elad, and A. Bruckstein, *K-svd: An algorithm for designing overcomplete dictionaries for sparse representation*, IEEE Trans. Signal Process., 54 (2006), pp. 4311–4322.

[2] X. Bian and H. Krim, *Optimal operator space pursuit: A framework for video sequence data analysis*, in Computer Vision—ACCV 2012, Springer, Berlin, 2013, pp. 760–769.

[3] X. Bian and H. Krim, *Robust Subspace Recovery via Bi-Sparsity Pursuit*, preprint, arXiv:1403.8067, 2014.

[4] X. Bian, H. Krim, A. Bronstein, and L. Dai, *Sparse null space basis pursuit and analysis dictionary learning for high-dimensional data analysis*, in Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE Press, Piscataway, NJ, 2015, pp. 3781–3785.

[5] O. Bousquet and L. Bottou, *The tradeoffs of large scale learning*, in Advances in Neural Information Processing Systems 20, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, eds., Curran Associates, Inc., Red Hook, NY, 2008, pp. 161–168.

[6] E. J. Candès, *The restricted isometry property and its implications for compressed sensing*, C. R. Math. Acad. Sci. Paris, 346 (2008), pp. 589–592.

[7] E. J. Candès, J. K. Romberg, and T. Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information.*, IEEE Trans. Inform. Theory, 52 (2006), pp. 489–509.

[8] E. J. Candès, J. K. Romberg, and T. Tao, *Stable signal recovery from incomplete and inaccurate measurements*, Comm. Pure Appl. Math., 59 (2006), pp. 1207–1223.

[9] Y. Chen, R. Ranftl, and T. Pock, *Insights into analysis operator learning: From patch-based sparse models to higher order MRFs*, IEEE Trans. Image Process., 23 (2014), pp. 1060–1072.
[10] T. F. Coleman and A. Pothen, *The null space problem* I. *Complexity*, SIAM J. Algebraic Discrete Methods, 7 (1986), pp. 527–537. doi:10.1137/0607059.
[11] T. F. Coleman and A. Pothen, *The null space problem* II. *Algorithms*, SIAM J. Algebraic Discrete Methods, 8 (1987), pp. 544–563. doi:10.1137/0608045.
[12] D. L. Donoho and C. Grimes, *Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data*, Proc. Natl. Acad. Sci. USA, 100 (2003), pp. 5591–5596.
[13] M. Elad, *Sparse and redundant representation modeling: What next?*, IEEE Signal Process. Lett., 19 (2012), pp. 922–928.
[14] M. Elad and M. Aharon, *Image denoising via sparse and redundant representations over learned dictionaries*, IEEE Trans. Image Process., 15 (2006), pp. 3736–3745.
[15] M. Elad, M. A. Figueiredo, and Y. Ma, *On the role of sparse and redundant representations in image processing*, Proc. IEEE, 98 (2010), pp. 972–982.
[16] Y. C. Eldar and G. Kutyniok, *Compressed Sensing: Theory and Applications*, Cambridge University Press, Cambridge, UK, 2012.
[17] E. Elhamifar and R. Vidal, *Sparse subspace clustering*, in Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009), IEEE Press, Piscataway, NJ, 2009, pp. 2790–2797.
[18] J. R. Gilbert and M. T. Heath, *Computing a sparse basis for the null space*, SIAM J. Algebraic Discrete Methods, 8 (1987), pp. 446–459. doi:10.1137/0608037.
[19] S. Hawe, M. Kleinsteuber, and K. Diepold, *Analysis operator learning and its application to image reconstruction*, IEEE Trans. Image Process., 22 (2013), pp. 2138–2150.
[20] K. Huang and S. Aviyente, *Sparse representation for signal classification*, in Advances in Neural Information Processing Systems 19, B. Schölkopf, J. C. Platt, and T. Hoffman, eds., MIT Press, Cambridge, MA, 2006, pp. 609–616.
[21] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T.-W. Lee, and T. J. Sejnowski, *Dictionary learning algorithms for sparse representation*, Neural Comput., 15 (2003), pp. 349–396.
[22] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov chains and mixing times*, AMS, Providence, RI, 2009.
[23] Z. Lin, M. Chen, and Y. Ma, *The Augmented Lagrange Multiplier Method for Exact Recovery of Corrupted Low-Rank Matrices*, preprint, arXiv:1009.5055, 2010.
[24] Z. Lin, R. Liu, and Z. Su, *Linearized alternating direction method with adaptive penalty for low-rank representation*, in Advances in Neural Information Processing Systems 24, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberge, eds., Curran Associates, Inc., Red Hook, NY, 2011, pp. 612–620.
[25] H. Liu, Z. Yang, and Z. Wu, *Locality-concept factorization*, in Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Spain, 2011, pp. 1378–1383.
[26] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, *Online dictionary learning for sparse coding*, in Proceedings of the 26th Annual International Conference on Machine Learning, ACM, New York, 2009, pp. 689–696.
[27] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, *Supervised dictionary learning*, in Neural Information Processing Systems 21, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, eds., Curran Associates, Inc., Red Hook, NY, 2009, pp. 1033–1040.
[28] S. Nam, M. E. Davies, M. Elad, and R. Gribonval, *The cosparse analysis model and algorithms*, Appl. Comput. Harmon. Anal., 34 (2013), pp. 30–56.
[29] A. Pothen, *Sparse null basis computations in structural optimization*, Numer. Math., 55 (1989), pp. 501–519.
[30] T. Randen and J. H. Husoy, *Filtering for texture classification: A comparative study*, IEEE Trans. Pattern Anal. Mach. Intell., 21 (1999), pp. 291–310.
[31] S. T. Roweis and L. K. Saul, *Nonlinear dimensionality reduction by locally linear embedding*, Science, 290 (2000), pp. 2323–2326.

[32] R. Rubinstein, T. Faktor, and M. Elad, *K-SVD dictionary-learning for the analysis sparse model*, in Proceedings of the 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE Press, Piscataway, NJ, 2012, pp. 5405–5408.

[33] R. Rubinstein, T. Peleg, and M. Elad, *Analysis K-SVD: A dictionary-learning algorithm for the analysis sparse model*, IEEE Trans. Signal Process., 61 (2013), pp. 661–677.

[34] B. Wen, S. Ravishankar, and Y. Bresler, *Structured overcomplete sparsifying transform learning with convergence guarantees and applications*, Int. J. Comput. Vision, 114 (2015), pp. 137–167.

[35] M. Yaghoobi, S. Nam, R. Gribonval, and M. E. Davies, *Analysis operator learning for overcomplete cosparse representations*, in Proceedings of the 9th European Conference on Signal Processing, IEEE Press, Piscataway, NJ, 2011, pp. 1470–1474.

[36] M. Yaghoobi, S. Nam, R. Gribonval, and M. E. Davies, *Constrained overcomplete analysis operator learning for cosparse signal modelling*, IEEE Trans. Signal Process., 61 (2013), pp. 2341–2355.

[37] G. Zhao, G. Wu, Y. Liu, and J. Chen, *Texture classification based on completed modeling of local binary pattern*, in Proceedings of the 2011 International Conference on Computational and Information Sciences (ICCIS), IEEE Press, Piscataway, NJ, 2011, pp. 268–271.