Learning Doubly Sparse Transforms for Images

Saiprasad Ravishankar, *Student Member, IEEE,* and Yoram Bresler, *Fellow, IEEE*

***Abstract*—The sparsity of images in a transform domain or dictionary has been exploited in many applications in image processing. For example, analytical sparsifying transforms such as Wavelets and DCT have been extensively used in compression standards. Recently, synthesis sparsifying dictionaries that are directly adapted to the data have become popular especially in applications such as image denoising. Following up on our recent work, where we introduced the idea of learning square sparsifying transforms, we propose here novel problem formulations for learning doubly sparse transforms for signals or image patches. These transforms are a product of a fixed, fast analytic transform such as the DCT, and an adaptive matrix constrained to be sparse. Such transforms can be learnt, stored, and implemented efficiently. We show the superior promise of our learnt transforms as compared to analytical sparsifying transforms such as the DCT for image representation. We also show promising performance in image denoising, which compares favorably with approaches involving learnt synthesis dictionaries such as the K-SVD algorithm. The proposed approach is also much faster than K-SVD denoising.**

***Index Terms*—Sparsifying transforms, Structured transforms, Image representation, Image denoising, Dictionary learning, Sparse representation, Compressed sensing.**

I. Introduction

A. Sparse Modeling: Synthesis, Analysis, and Transform

Sparse representation of signals and images has become widely popular in recent years. Two well-known models for sparse representation are the synthesis model and the analysis model [1]. The *synthesis model* suggests that a signal $y \in \mathbb{R}^n$ may be represented as a linear combination of a small number of columns/atoms from a synthesis dictionary $D \in \mathbb{R}^{n \times K}$. Here, the assumption is $y = Dx$ with $x \in \mathbb{R}^K$ being sparse, i.e., $\|x\|_0 \ll K$. The l_0 quasi norm counts the number of non-zero entries in x. Real-world signals more generally satisfy $y = Dx + e$, where e is an approximation term in the signal domain [2]. When $K = n$ and D is full rank, we have a basis representation. When $K > n$, the dictionary is said to be overcomplete.

Given the signal y and synthesis dictionary D, the problem of finding the sparse representation x is the *synthesis sparse coding* problem [3]. The problem is to find x that minimizes $\|y - Dx\|_2^2$ subject to $\|x\|_0 \leq s$, where s is the required sparsity level. This synthesis sparse coding problem is known to be NP-hard (Non-deterministic Polynomial-time hard) [4], [5], and is therefore, in general, computationally intractable. In practice, the problem can be solved approximately by greedy [6]–[8], or relaxation algorithms [9]–[11]. Under certain conditions, these algorithms can be guaranteed to provide the correct solution, or provide it with high probability. However, all these various algorithms are computationally expensive in practice, particularly for large-scale problems.

On the other hand, the *analysis model* [1], [12], [13] suggests that a signal $y \in \mathbb{R}^n$ is sparse in an "analysis" domain, i.e., given an analysis dictionary $\Omega \in \mathbb{R}^{m \times n}$, we assume $\Omega y \in \mathbb{R}^m$ to be sparse ($\|\Omega y\|_0 \ll m$). When the signal y is noisy, it is more generally assumed to satisfy a *noisy signal analysis model*, which states that $y = q + e$ with Ωq being sparse, and e is the noise in the signal domain.

Given the noisy signal y and analysis dictionary Ω, the problem of finding the noiseless vector q is known as the *analysis sparse coding* problem [13]. This problem is to find q by minimizing $\|y - q\|_2^2$ subject to $\|\Omega q\|_0 \leq m - l$, where l is referred to as the co-sparsity level (number of zeros) [13]. This problem too is NP-hard just like sparse coding in the synthesis model. Similarly to sparse coding in the synthesis model, approximate algorithms exist for analysis sparse coding [12]–[16], which however, tend to be computationally expensive.

In this paper, we focus on a third alternative - a generalization of the analysis model, which we call the transform model [17]. It suggests that a signal y is approximately sparsifiable using a transform $W \in \mathbb{R}^{m \times n}$. Here, the assumption is that $Wy = x + \eta$, where $x \in \mathbb{R}^m$ is sparse, i.e., $\|x\|_0 \ll m$, and η is a small residual error in the transform domain. This is one of the distinguishing features of the transform model from the synthesis and the noisy analysis models, in both of which the error term is in the signal domain. Natural signals and images are well-known to be approximately sparse in analytical transform domains such as Wavelets [18], discrete cosine transform (DCT), Ridgelets [19], Contourlets [20], Curvelets [21], etc.

The transform model is a generalization of the analysis model with Ωy exactly sparse. The generalization allows the transform model to include a wider class of signals within its ambit than the analysis model. Moreover, while the analysis model enforces the sparse code (Ωy) to lie in the range space of Ω, the sparse representation x in the transform model is not forced to lie in the range space of W. This makes the transform model more general than even the noisy signal analysis model (cf. [17]). The reason we have chosen the name "transform model" is because the assumption $Wy \approx x$ has been traditionally used in transform coding (with orthonormal transforms), and the concept of transform coding is older [22] and pre-dates the terms analysis and synthesis [23].

When a suitable sparsifying transform W is known for the signal y, the process of obtaining a sparse code x of given sparsity s involves minimizing $\|Wy - x\|_2^2$ subject to

This work was supported in part by the National Science Foundation (NSF) under grant CCF 10-18660.

S. Ravishankar and Y. Bresler are with the Department of Electrical and Computer Engineering and the Coordinated Science Laboratory, University of Illinois, Urbana-Champaign, IL, 61801 USA e-mail: (ravisha3, ybresler)@illinois.edu.

$\|x\|_0 \leq s$. For conciseness, we call this the *transform sparse coding problem* [1]. This problem is easy and its solution is obtained exactly by zeroing out all but the s coefficients of largest magnitude in Wy. In contrast, sparse coding with a given synthesis or analysis dictionary involves solving an NP-hard problem approximately. Moreover, given W and sparse code x, we can also recover a least squares estimate of the true signal y by minimizing $\|Wy - x\|_2^2$ over all $y \in \mathbb{R}^n$. The recovered signal is $W^\dagger x$, with $W^\dagger$ denoting the pseudo-inverse of W. Thus, unlike the previous models, the transform model allows for exact and fast computations, a property that has been exploited heavily in the context of analytical sparsifying transforms.

B. Learning Sparse Representations

1) Synthesis Dictionary learning: The idea of adapting synthesis dictionaries to a particular class of signals, also known as synthesis dictionary learning, has been studied in recent papers [25]–[28]. Given a matrix $Y \in \mathbb{R}^{n \times N}$ whose columns represent training signals, the problem of learning a synthesis dictionary D that gives a sparse representation for the signals in Y is formulated as follows [27]

$$\text{(P0s)} \quad \min_{D,X} \|Y - DX\|_F^2 \;\; s.t. \;\; \|X_i\|_0 \leq s \;\; \forall \; i$$

Here, the columns of the matrix $X \in \mathbb{R}^{K \times N}$ denote the sparse codes of the signals in Y. The subscript i indexes the i^{th} column, and the sparsity level allowed for each training signal is s. The columns of the learnt dictionary in (P0s) are typically additionally constrained to be of unit norm in order to avoid the scaling ambiguity [29]. While Problem (P0s) uses the l_0 quasi norm for sparsity, one could also alternatively use its convex relaxation, the l_1 norm [30].

Although Problem (P0s) is highly non-convex and NP-hard, effective heuristics for the solution of (P0s) (or for its variations) have been proposed in recent years [26]–[28], [30], [31]. The algorithms for dictionary learning typically alternate between finding the dictionary D (dictionary update step), and the sparse representations X (sparse coding step). The dictionary update step has been addressed in numerous ways [26], [27], [30], [31]. Of the various algorithms, the K-SVD algorithm [27] has been particularly popular in applications.

Since Problem (P0s) is non-convex, methods such as K-SVD can get easily caught in local minima, or even saddle points [32]. Moreover, the sparse coding step in such methods is computationally expensive. Recent theoretical analysis of dictionary identification using an l_1 relaxation of the sparsity penalty on X provides conditions under which the desired solution is a local minimum of the cost function [29] [33], and under restrictive assumptions (square, non-singular dictionary, Bernoulli-Gaussian X, and noiseless data) an algorithm recovering the correct dictionary with high probability has been proposed [34]. However, no proof exists to date for the global convergence of K-SVD or other popular synthesis dictionary learning algorithms.

[1]The transform sparse coding solution can also be alternatively written as $E(Wy)$, where $E(\cdot)$ is the Moreau-Yosida regularization [24] of the indicator function of the ℓ_0 ball $\left\{x \in \mathbb{R}^m : \|x\|_0 \leq s\right\}$. This interpretation is not surprising given that the transform model generalizes the analysis model.

Adaptive synthesis dictionaries have been demonstrated to be useful in numerous image processing applications such as denoising, inpainting, deblurring, and demosaicing [35]–[39]. Synthesis dictionary learning has also been applied to inverse problems such as those in Tomography [40], and magnetic resonance imaging (MRI) involving highly undersampled k-space data [41], [42].

2) Analysis Dictionary Learning: While there has been considerable research on learning synthesis dictionaries, the idea of learning analysis dictionaries has received only recent attention [43]–[45]. However, the global convergence of these algorithms is unclear, even in the case when l_1 relaxation is used for sparsity.

Some authors alternatively consider analysis dictionary learning employing the noisy signal analysis model [13], [14], [16]. The following formulation is by Rubinstein et al. [13], [14].

$$\text{(P0a)} \quad \min_{\Omega,Z} \|Y - Z\|_F^2 \;\; s.t. \;\; \|\Omega Z_i\|_0 \leq m - l \;\; \forall \; i$$

Here, the columns of the matrix $\Omega Z \in \mathbb{R}^{m \times N}$ have a co-sparsity level $\geq l$. The rows of Ω are also constrained to be of unit norm in (P0a). Yaghoobi et al. [16], [46] relax the l_0 quasi norm in (P0a) to an l_1 norm and add it as a penalty in the cost. They also additionally restrict Ω to be a tight frame ($\Omega^T\Omega = I_n$, where $(.)^T$ denotes the matrix transpose operation, and I_n is the $n \times n$ identity matrix).

The analysis dictionary learning problem (P0a) is non-convex and NP-hard. Heuristics such as the analysis K-SVD [13], [14] alternate between updating Ω and Z, and employ a backward-greedy method for updating Z. However, no convergence guarantees exist for these algorithms. Indeed, because these algorithms are similar to those proposed in the synthesis case, we expect that they can get similarly stuck in bad local minima.

The usefulness of learnt analysis dictionaries in image processing applications (see Section V-E for a discussion of the work of Yaghoobi et al. [46]) has not been sufficiently explored.

3) Transform Learning: Analytical sparsifying transforms such as Wavelets and DCT have been extensively used in image processing including in various compression algorithms and compressions standards such as JPEG2000 [47], and more recently in compressed sensing MRI reconstruction [48], [49]. However, adapting the sparsifying transform to the data could improve its performance significantly in these and other applications. In this adaptive setting, given a matrix $Y \in \mathbb{R}^{n \times N}$ whose columns represent training signals, we recently proposed learning a transform W such that the "sparsification error" denoted by $\|WY - X\|_F^2$ is minimized [17]. Here, X is an unknown matrix containing the sparse codes of the training signals as columns. In contrast to the prior work on synthesis and analysis dictionary learning, the transform model allows WY to be approximated by a sparse matrix X, which makes the learning of this model easier. Specifically, with the transform model, sparse coding is cheap and exact. Furthermore, unlike synthesis or analysis dictionary learning, the transform learning formulation [17] does not include a

highly non-convex function involving the product of two unknown matrices.

C. Contributions

*1) **Doubly Sparse Transform Model**:* Imposing additional structure on the learnt transform leads to computational advantages over the 'unstructured' transforms introduced in our previous work [17]. We propose to learn a square ($n \times n$) sparsifying transform W that is a product of two different transforms, i.e., $W = B\Phi$, where $\Phi \in \mathbb{R}^{n \times n}$ is an analytical transform with an efficient implementation, and $B \in \mathbb{R}^{n \times n}$ is a transform that is constrained to be sparse (i.e., has few significant non-zero elements). Such a transform W is said to be 'doubly sparse', since it provides a sparse representation for data and has matrix B that is sparse.

The proposed doubly sparse transform structure combines the advantages of trained and analytic transforms: adapting to the data, it provides better representations and denoising than analytical transforms, yet owing to the sparsity of B, it can be stored and applied efficiently. As we show, it can also be learnt more efficiently than unstructured transforms [17].

The structure $W = B\Phi$ is expected to be an effective sparsifying transform because Φ matrices such as the DCT when applied to natural images (or image patches) produce a result that is already approximately sparse. Therefore, by further modifying the result using only the limited number of degrees of freedom in a sparse transform B, one may be able to produce a highly sparse result.

Learning a *synthesis* dictionary composed of a product of an analytical dictionary and a sparse matrix has been proposed with a different motivation by Rubinstein et al. [50]. Their formulation is still non-convex and NP-hard, and their learning algorithm is a variation of K-SVD and thus suffers from similar drawbacks as K-SVD (i.e., lack of any convergence guarantees). Another recent work [51] learns synthesis dictionaries for the different bands in the wavelet domain of images. However, the atoms of those dictionaries are not constrained to be sparse, and it is unclear if the method can outperform previous work [35], [36], [38], [50] in applications.

*2) **Main Highlights**:* In this work, we focus on square transforms. We propose novel problem formulations for learning doubly sparse transforms that are well-conditioned. The main results of our work are enumerated as follows.

(i) When the sparsity of B is measured using a convex differentiable penalty, we guarantee the convergence of the objective in our proposed alternating algorithm. When the non-convex ℓ_0 quasi norm is used to measure sparsity of B, we show the convergence of the objective and iterates in our algorithm empirically. These desirable convergence properties contrast with those of popular synthesis or analysis learning algorithms such as K-SVD, for which no such results, theoretical or empirical, are available.

(ii) The adapted doubly sparse transform $B\Phi$ provides a better representation of images than the analytical transform Φ.

(iii) Doubly sparse transforms can be learnt for natural images with highly sparse B matrices, with only a marginal loss in image representation quality compared to unstructured (or non-sparse) transforms. In fact, imposing the doubly sparse property leads to much faster convergence of learning, and faster computations with the sparse transform.

(iv) The learnt doubly sparse transforms have reduced storage requirement and generalize better than the unstructured transforms.

(v) The proposed doubly sparse transform learning enables the use of "cheap" analytical transforms Φ such as the Hadamard transform, with essentially the same image representation quality as with better Φ, but at a substantially lower computational cost.

(vi) Doubly sparse learning is robust to the size of the training set, and requires far fewer training signals than unstructured transform learning.

(vii) We present a novel adaptive sparsifying-transform-based image denoising formulation and algorithm in this work. The denoising performance of the learnt doubly sparse transforms is comparable to, or better than overcomplete K-SVD. Most importantly, doubly sparse denoising is much faster. The doubly sparse transforms also denoise faster than the unstructured transforms of our previous work [17], with little loss in denoising quality. In fact, at high noise, the doubly sparse transforms mitigate overfitting to noise, and denoise somewhat better than unstructured transforms, which have more degrees of freedom.

The rest of this paper is organized as follows. Our prior work on 'unstructured' transform learning and our proposed problem formulations for doubly sparse transform learning are described in Section II. Section III details the proposed algorithms for learning doubly sparse transforms and discusses their relevant properties such as convergence and computational cost. In Section IV, we introduce the image denoising formulation and algorithm incorporating adaptive sparsifying transforms. Section V demonstrates the performance of our algorithms in image representation and image denoising. In Section VI, we conclude with proposals for future work.

II. Transform Learning Formulations

Given the training matrix $Y \in \mathbb{R}^{n \times N}$, we propose to minimize the sparsification error given by $\|WY - X\|_F^2$ where $W \in \mathbb{R}^{m \times n}$ and X has sparse columns, subject to constraints on W. The notion of sparsification error is justified by the fact that natural signals and image patches are reasonably approximated in a transform domain such as Wavelets using few (say s) significant non-zero transform coefficients.

A. Unstructured Transform Learning

We recently proposed the following problem formulation for learning a square 'unstructured' transform W ($m = n$) [17]. (We use the terminology 'unstructured' to distinguish this transform from the 'doubly sparse' transform of the present paper.)

$$\text{(P1)} \quad \min_{W,X} \ \|WY - X\|_F^2 - \lambda \log |\det W| + \mu \|W\|_F^2 \quad s.t. \ \ \|X_i\|_0 \leq s \ \ \forall \ i$$

Here, the $\log |\det W|$ penalty[2] helps enforce full rank on the transform W and eliminates degenerate solutions (such as those with zero, or repeated rows). Many well-known (square) sparsifying transforms such as the DCT, Wavelets, and Hadamard are non-singular. Even the one-dimensional finite difference transform, which typically has more columns than rows, can be appended with a linearly independent row(s), thereby making it non-singular. For the two-dimensional case, the finite difference transform can be written as a Kronecker product of two non-singular one-dimensional finite difference matrices [17].

The $\|W\|_F^2$ penalty in (P1) helps remove a 'scale ambiguity' [17] in the solution (the scale ambiguity occurs when the data admits an exactly sparse representation), and together with the $-\log |\det W|$ penalty additionally helps control the condition number of the learnt transform. Badly conditioned transforms typically convey little information and may degrade performance in applications. As the parameter λ is increased in (P1) with fixed ratio μ/λ, the optimal/minimizing transform(s) become well-conditioned. In the limit $\lambda \to \infty$, their condition number tends to 1 [17], and their singular values tend to $\sqrt{\lambda/2\mu}$; thus, the ratio μ/λ only controls the scaling of the optimal W. We set $\lambda = \mu$ which results, in the limit $\lambda \to \infty$, in the optimizing transform having a spectral norm of $1/\sqrt{2}$. This setting was also observed to work well in practice.

We have shown [17] that the cost function in Problem (P1) is lower bounded. Moreover, the lower bound is achieved for the problem if there exists a pair $(\hat{W}, \hat{X})$ with $\hat{X}$ sparse (which is the constraint) such that $\hat{W}Y = \hat{X}$, and the singular values of $\hat{W}$ are all equal to $\sqrt{\lambda/2\mu}$, which equals $1/\sqrt{2}$ for our choice of $\mu = \lambda$. Thus, the lower bound favours both a good sparsification error and good transform conditioning.

Another interesting property relates the norms of the rows (or, equivalently the columns) of the transform to its condition number. Denoting the i$^{\text{th}}$ row of W by w_i, we have that

$$\max_{i,j} \frac{\|w_i\|_2 - \|w_j\|_2}{\|w_j\|_2} \leq \kappa(W) - 1 \tag{1}$$

The bound follows from the fact that the largest and smallest singular values of W obey $\sigma_1 \geq \max_i \|w_i\|_2$ and $\sigma_n \leq \min_i \|w_i\|_2$, respectively. The bound is tight when $\kappa(W) = 1$ (in which case all rows of W have identical norms), and it indicates that the norms of the rows of W can be controlled by controlling the condition number. This eliminates the need to have separate constraints on row or column norms of W like in the synthesis and analysis cases, where such a constraint is needed to eliminate the scaling ambiguity.

We note that penalty terms similar to $-\log |\det W|$ and $\|W\|_F^2$ can also be used to regularize synthesis and analysis dictionary learning in order to enforce full-rank, well-conditioning, overcome scale ambiguity, etc.

A cost function similar to that in (P1) but lacking the $\|W\|_F^2$ penalty has been derived under certain assumptions in a very different setting of blind source separation [52]. However, the transform learning Problem (P1) performs poorly in image processing applications in the absence of the crucial $\|W\|_F^2$ penalty [17].

As already discussed, our transform learning Problem formulation (P1) offers explicit advantages over the synthesis (P0s), and analysis (P0a) dictionary learning formulations. The sparse coding problem (i.e., Problem (P1) with fixed W) admits an easy and exact solution, and (P1) also does not include a highly non-convex function of the product of two unknown matrices.

Problem (P1) has been shown [17] to give rise to transforms that provide better representations than analytical transforms such as the DCT. Importantly, transform learning has a low computational cost, and is typically much faster than dictionary learning [17]. Empirical results have demonstrated convergence of the iterates for our algorithms regardless of initial conditions [17].

Since the transform model suggests $WY = X + E$, where E is the sparsification error term in the transform domain, we have $Y = W^{-1}X + W^{-1}E$ when W is square and non-singular. The error of signal recovery from the transform sparse code X is thus $W^{-1}E$, which depends on both the sparsification error and conditioning of W. It can thus be controlled by controlling the sparsification error and condition number, as we do in (P1).

B. Doubly Sparse Transform Learning

The proposed doubly sparse structure $W = B\Phi$ combines adaptivity with the efficiency of the analytical Φ. For example, the cost of applying the transform Φ to an n-dimensional signal scales as $O(n \log n)$ for the Hadamard, DCT, Discrete Fourier Transform (DFT), etc. We now formulate the learning of such a doubly sparse transform by replacing W with $B\Phi$ in Problem formulation (P1) with $\lambda = \mu$.

$$\text{(P2)} \quad \min_{B,X} \|B\Phi Y - X\|_F^2 - \lambda \log |\det (B\Phi)| + \lambda \|B\Phi\|_F^2 \\ s.t. \quad \|B\|_0 \leq r, \ \|X_i\|_0 \leq s \ \forall \ i$$

where $\|B\|_0 \triangleq \sum_{i,j} 1_{\{B_{ij} \neq 0\}}$, with B_{ij} the entry of B from row i and column j, and $1_{\{B_{ij} \neq 0\}}$ is the indicator function of $B_{ij} \neq 0$. The l_0 quasi norm constraint on matrix B enforces sparsity of the entries of B. We assume r non-zeros in B. As the sparsity level $r \to n^2$, Problem (P2) tends to Problem (P1), but instead involves learning a transform W that is decomposable as $B\Phi$. For invertible Φ, the two problems are exactly equivalent in the limit $r = n^2$.

Now, since the determinant is multiplicative, we obtain that

$$\log |\det (B\Phi)| = \log |\det B| + \log |\det \Phi| = \log |\det B| + C$$

where the constant $C = \log |\det \Phi|$. Moreover, if matrix Φ is orthonormal (for example, Wavelets, or DCT), we get that $\|B\Phi\|_F^2 = \|B\|_F^2$.

With these simplifications, the problem formulation (P2) becomes

$$\text{(P2)} \quad \min_{B,X} \left\|B\hat{Y} - X\right\|_F^2 - \lambda \log |\det B| + \lambda \|B\|_F^2 \\ s.t. \quad \|B\|_0 \leq r, \ \|X_i\|_0 \leq s \ \forall \ i$$

[2] In [17], we imposed the restriction $\det W > 0$, which can be made without loss of generality in (P1), since we can switch from a W with $\det W < 0$ to one with $\det W > 0$ trivially by swapping two rows of W.

where $\hat{Y} = \Phi Y$. Thus, the doubly sparse problem formulation (P2) is similar to (P1), but with the additional, yet crucial sparsity constraint. Note that for non-orthonormal Φ, we would still have the $\|B\Phi\|_F^2$ penalty in the cost.

Now, the set of matrices with unit condition number also includes scaled identity matrices, which are the sparsest possible matrices[3]. Therefore, as $\lambda \to \infty$ in (P2), the condition number of the optimal/minimizing transform(s) tends to 1, just like in the unstructured case [17].

As an alternative to (P2), we consider the following formulation, which replaces the l_0 constraint on B with a convex penalty $h(B)$ in the cost with weighting ζ. This formulation will be shown to lead to an algorithm with a convergence guarantee.

$$\text{(P3)} \min_{B,X} \left\|B\hat{Y} - X\right\|_F^2 - \lambda \log |\det B| + \lambda \|B\|_F^2 + \zeta h(B) \quad s.t. \ \ \|X_i\|_0 \leq s \ \forall \ i$$

For example, $h(B)$ can be the l_1 penalty $\|B\|_1 = \sum_{i,j} |B_{ij}|$. Alternatively, we consider a slightly smoothed l_1 penalty $h(B) = \sum_{i,j} \sqrt{B_{ij}^2 + \epsilon}$, where $\epsilon > 0$ is chosen sufficiently small. Such a penalty is amenable to optimization schemes that exploit the gradient. The relaxed formulation (P3) was empirically observed to perform similarly to Problem (P2). However, Problem (P2) with the exact sparsity constraint on B, while not enjoying the same convergence guarantee as (P3), enables faster learning (shown in Section III).

Both Problems (P2) and (P3) admit an equivalence class of solutions [17]. Given a minimizer $(\hat{B}, \hat{X})$ with $\hat{X}$ sparse, we can form trivially equivalent minimizers by simultaneously permuting the rows of $\hat{B}$ and $\hat{X}$, or by pre-multiplying them with a diagonal ± 1 matrix.

One could constrain B in (P3) to be positive definite, i.e., $B \succ 0$. If in addition the l_0 quasi norm for X were relaxed to an l_1 norm constraint or l_1 penalty, the resulting problem would be jointly convex in B and X. However, upon further investigation, we found that enforcing B to be positive definite is too restrictive, and provides almost no improvement over the analytical Φ. In fact, experimental results (see for example, Figure 2 and Section V) show that B learnt via (P2) has an approximately skew-symmetric off-diagonal, thus implying $B \not\succ 0$. We could rely on the observed properties of the learnt B for natural images to design reasonable convex learning formulations. However, a discussion of such formulations is beyond the scope of this work and will be presented elsewhere.

C. Alternative Doubly Sparse Formulations

We have also studied the learning of an alternative doubly sparse structure $W = \Phi B$, which was however, empirically observed to perform worse than the proposed $B\Phi$ structure in applications. Moreover, the structure $B\Phi$ has computational advantages over the alternative ΦB. For the former, the product $\hat{Y} = \Phi Y$ can be pre-computed once, before an optimization algorithm begins, whereas the product $\Phi(BY)$ would have to be computed repeatedly in iterative algorithms, when employing the latter ΦB structure.

Finally, while many more useful properties can be enforced on sparsifying transforms, these are beyond the scope of the current work.

[3]Since no non-singular matrix B exists for $r < n$, such sparsity levels are inadmissible for (P2).

III. Algorithms and Properties

A. Algorithms

Here, we outline algorithms for solving the doubly sparse transform learning problems (P2) and (P3). In earlier work on unstructured square transforms [17], we proposed an alternating algorithm for solving Problem (P1). Our algorithms for solving Problem (P2) or (P3) also alternate between updating X and B.

1) Sparse Coding Step: In this step, we solve Problem (P2) (or (P3)) with fixed B.

$$\min_X \left\|B\hat{Y} - X\right\|_F^2 \ \ s.t. \ \ \|X_i\|_0 \leq s \ \forall \ i \tag{2}$$

The solution X can be computed exactly by zeroing out all but the s coefficients of largest magnitude in each column of $B\hat{Y}$. We refer to this operation as *projection onto the ℓ_0 ball*.

2) Transform Update Step: In this step, we solve Problem (P2), or (P3) with fixed X. For Problem (P3), the transform update step involves the following problem, with $h(B) = \sum_{i,j} \sqrt{B_{ij}^2 + \epsilon}$.

$$\min_B \left\|B\hat{Y} - X\right\|_F^2 - \lambda \ \log |\det B| + \lambda \|B\|_F^2 + \zeta h(B) \tag{3}$$

This unconstrained minimization problem is non-convex but smooth. One could solve it using iterative procedures such as gradient descent, or the conjugate gradient (CG) algorithm [53] (which typically converges faster). The CG method can be employed with backtracking line search [53] (Armijo step size rule), which guarantees decreasing cost (and thus prevents the algorithm from entering the barrier region). Fixed (sufficiently small) step size rules were also empirically observed to work well and faster. The CG method converges quickly, and can be executed for a fixed number of iterations in practice, to save run time.

In the case of Problem (P2), the transform update step solves the following optimization problem.

$$\min_B \left\|B\hat{Y} - X\right\|_F^2 - \lambda \ \log |\det B| + \lambda \|B\|_F^2 \quad s.t. \ \ \|B\|_0 \leq r \tag{4}$$

This constrained problem does not have have an analytical solution. Moreover, the constraint set is non-convex. One could perform the transform update using the iterative projected gradient algorithm, or projected CG [53]. However, we found that the alternative heuristic strategy of employing a few iterations of the standard CG algorithm followed by post-thresholding led to better empirical performance in terms of the metrics defined in Section V-A. Hence, we choose this alternative strategy. Matrix B is essentially thresholded after some iterations of CG, and we retain only the r elements of largest magnitude.

The gradient expressions for the various terms in the objective function of (P2) or (P3) (used for CG) are listed below for completeness.

$$\nabla_B \log |\det B| = B^{-T} \tag{5}$$

$$\nabla_B \|B\|_F^2 = 2B \tag{6}$$

$$\nabla_B \|BY - X\|_F^2 = 2BYY^T - 2XY^T \tag{7}$$

$$\nabla_{B_{ij}} \left(\sqrt{B_{ij}^2 + \epsilon} \right) = \frac{B_{ij}}{\sqrt{B_{ij}^2 + \epsilon}} \tag{8}$$

When the transform update step of (P2) is solved using the proposed heuristic strategy, then the determinant of B needs to be monitored after the post-thresholding, to ensure that the algorithm does not enter the log barrier. If the thresholding produces a B with $\det(B) = 0$, we perturb B away from $\det(B) = 0$ by adding a random matrix of small l_2 norm. In all our experiments (Section V), we observed that with reasonable initializations for B, such as the identity matrix, the aforementioned extra step was never invoked by the algorithm for (P2), as it did not reach the degenerate state.

B. Convergence

Here, we discuss the convergence of the proposed doubly sparse transform learning algorithms. The cost functions in our formulations are all lower bounded [17].

The algorithms alternate between sparse coding and transform update steps. The solution for the sparse coding step of (P2) and (P3) is exact. Therefore, the respective cost functions decrease in this step.

For the transform update step of (P3), the solution is obtained by conjugate gradients (for instance with Armijo step size rule). Thus, in this step, the cost function can again only decrease for (P3). The cost function being monotone decreasing and lower bounded, it must converge for the alternating algorithm for (P3).

On the other hand for (P2), the transform update step involving CG followed by post-thresholding is a heuristic approach. Hence, we cannot guarantee convergence of the cost in this case. However, as illustrated empirically in Section V, both the cost and the iterate converge for the alternating algorithm for (P2). It will be also demonstrated empirically in Section V that for reasonably sparse B (small r), the algorithm for Problem (P2) converges much faster (i.e., in fewer iterations) than that for (P1). We hypothesize that this is because, for small r, doubly sparse transforms have far fewer free parameters than unstructured transforms.

C. Computational Cost

We now demonstrate the low computational cost of the proposed algorithms. We estimate the cost of each step of our algorithms. The sparse coding step in Problem (P2) requires rN multiply-add operations to compute $B\hat{Y}$, when B has r non-zeros. Since $r = \beta n^2$, where typically the non-zero fraction $\beta \ll 1$, the preceding cost becomes $\beta n^2 N$. The projection of $B\hat{Y}$ onto the ℓ_0 ball (2), if done by full sorting [54] would involve $O(nN \log n)$ computations. Thus, the sparse coding step of Problem (P2) has a cost of $Nn(\beta n + C_2 \log n)$, where C_2 is a constant. For $\beta < 1$, this is lower (better) than the cost of the sparse coding step for the 'unstructured' Problem (P1) [17]. Thus, the doubly sparse problem formulation can provide speed-ups over the unstructured (P1) in learning. In any case, the cost of sparse coding is dominated by $\beta n^2 N$.

When the doubly sparse learning involves the convex sparsity penalty for B (Problem (P3)), the matrix B in each iteration may not be exactly sparse (may only be compressible). Hence, the computational speed-ups in the sparse coding step for this case are typically lower than with the l_0 quasi norm.

Other than the extra post-thresholding in (P2), the algorithms for both (P1) and (P2) use conjugate gradients in the transform update step. Hence, since the objective functions of (P2) and (P1) are the same, so are the costs of the respective conjugate gradient steps, which are roughly $\alpha N n^2 + (1 + C_3) J n^3$ [17]. Here, $\alpha = s/n$, J is the number of conjugate gradient steps, and C_3 is a constant. The $\alpha N n^2$ cost arises from the computation of XY^T (7) (computed once at the beginning of the transform update step), while the n^3 costs arise from computing B^{-T} (5) and BYY^T (YY^T assumed pre-computed once at a total cost of Nn^2 for the entire algorithm). The post-thresholding of B in the transform update step of (P2) if done by sorting would require $O(n^2 \log n)$ operations. Since $\log n \ll N$ typically, and assuming that $(1 + C_3) J n < \alpha N$, the cost per transform update step of (P2) is dominated by $\alpha N n^2$. The cost per transform update step of (P3) scales similarly. (Monitoring the determinant of B after the post-thresholding step of (P2) would take $O(n^3)$ operations, but this is typically unnecessary for reasonable initializations for B.)

The total cost per iteration (of sparse coding and transform update) of the proposed doubly sparse transform learning algorithms is thus roughly $(\alpha + \beta) N n^2$. (To account for the non-exact sparsity of B in (P3), the non-zero fraction β needs to be increased in this expression in the case of (P3).) In contrast, for the unstructured transform learning case [17], the cost per iteration is roughly $(\alpha + 1) N n^2$. A direct comparison of the costs indicates the computational efficacies of the proposed doubly sparse transform learning. The smaller the β, the greater the speedup that can be expected for doubly sparse learning. Nonetheless, the cost per algorithm iteration scales in order as $O(n^2 N)$ for all the transform learning algorithms.

Compared to the computational cost of synthesis or analysis learning, the costs of the proposed doubly sparse transform learning algorithms are significantly lower. We have shown [17] that the computational cost per-iteration of our algorithm for (P1) is much lower than the per-iteration cost of synthesis dictionary learning algorithms such as K-SVD (which scales as $O(n^3 N)$ for square dictionaries). A similar advantage exists over the per-iteration cost of analysis K-SVD [13]. The actual per-iteration run times for (P1) were also shown [17] to be orders of magnitude smaller than for synthesis K-SVD. Since, as argued above, the algorithm for Problem (P2) is cheaper than for (P1), we can expect even shorter per-iteration run times for doubly sparse learning compared to K-SVD.

IV. Application to Image Denoising

A. Problem Formulation

Image denoising is a widely studied image processing problem of estimating an image $x \in \mathbb{R}^P$ (2D image represented as a vector) from its measurement $y = x + h$ corrupted by noise h. We focus in this work on adaptive sparsifying-transform-based image denoising. Our goal here is an initial exploration of the potential of the proposed framework and algorithms for learning a sparsifying transform in this classical and prototypical application.

Similar to previous dictionary-based denoising approaches [16], [35], [38], [46], we work with overlapping image patches, and learn a transform adapted to them. We propose a simple problem formulation (similar to the one proposed for signal denoising [17]) for denoising image patches using sparsifying transforms, as follows.

$$\min_{W,\{x_i\},\{\alpha_i\}} \sum_{i=1}^{M} \|Wx_i - \alpha_i\|_2^2 + \lambda Q(W) + \tau \sum_{i=1}^{M} \|R_i\, y - x_i\|_2^2 \quad s.t. \quad \|\alpha_i\|_0 \leq s_i \ \forall\, i \tag{P4}$$

Here, $Q(W)$ represents the portion of the cost depending on only W. For the doubly sparse case, $Q(W) = Q(B)$, since $W = B\Phi$. For example, the $Q(W)$ corresponding to (P2) includes the log-determinant and frobenius norm penalties, along with the indicator function of the set of r-sparse matrices. Vector $R_i\, y$ in (P4) denotes the i^{th} patch of image y (M overlapping patches are assumed), with $R_i \in \mathbb{R}^{n \times P}$ being the operator that extracts it as a vector from the image. We assume that the noisy patch $R_i\, y$ can be approximated by a noiseless version x_i that is approximately sparsifiable by an adaptive transform (the *noisy signal transform model* [17], see also [55]). Vector $\alpha_i \in \mathbb{R}^n$ denotes the transform sparse code of x_i with an apriori unknown number s_i of non-zeros.

The denoising model here is different from the model assumed in Problem (P2), since the latter is aimed at sparse image representation (without requiring explicit denoising). The parameter τ in (P4) is typically inversely proportional to the noise level σ [35]. When $\sigma = 0$, the optimal $x_i = R_i\, y$, and (P4) reduces to a transform learning problem. Once the denoised patches x_i are found, the denoised image x is obtained by averaging the x_i's at their respective locations in the image (cf. [41] for a similar technique).

B. Algorithm

Problem formulation (P4) denoises the patches using an adaptive W, and is non-convex. The algorithm that we propose to solve Problem (P4) iterates over a transform learning step and a variable sparsity update step. Once the iterations complete, there is a denoised image update step.

1) Transform Learning: In this step, we fix $x_i = R_i\, y$ and $s_i = s$ (fixed s initially) for all i in (P4), and solve for W and α_i ($i = 1, 2, ..., M$) using our proposed transform learning algorithms.

2) Variable Sparsity Update: In this step, we update the sparsity levels s_i for all i.

Image Denoising Algorithm

Input : y - Noisy Image, s - fixed sparsity, L - number of iterations

Output : x - Denoised image

Initialization : Patches $x_i = R_i y$, $s_i = s$ for $i = 1, 2, ..., M$

For k = 1:L Repeat

1) Learn W and α_i for patches $x_i = R_i y$ with known sparsities s_i, for all $i = 1, 2, ..., M$.
2) Update s_i for all $i = 1, 2, ..., M$: Increase s_i in $\alpha_i = H_{s_i}(WR_i y)$ in (9), until the error condition $\|R_i\, y - x_i\|_2^2 \leq nC^2\sigma^2$ is reached.

End

Update x **:** Average the denoised patches x_i at respective image locations.

Fig. 1. Algorithm to denoise images using (P4).

For fixed W and α_i ($i = 1, 2, ..., M$), (P4) reduces to a least squares problem in the x_i's. Each x_i can then be independently updated as follows, where † denotes the pseudo-inverse.

$$x_i = G \begin{bmatrix} \sqrt{\tau} R_i y \\ \alpha_i \end{bmatrix}, \text{where } G = \begin{bmatrix} \sqrt{\tau} I \\ W \end{bmatrix}^{\dagger} \tag{9}$$

However, we do not fix α_i in (9), and instead only let it be a thresholded version of $WR_i\, y$, and determine the sparsity level s_i. Let $H_{s_i}(b)$ denote the operator that zeroes out all but the s_i elements of largest magnitude of $b \in \mathbb{R}^n$. Then, our assumption is that $\alpha_i = H_{s_i}(WR_i y)$ for some s_i that is unknown apriori.

We choose the sparsity s_i for the i^{th} patch such that the error term $\|R_i\, y - x_i\|_2^2$ computed after updating x_i by (9) (with α_i held at $H_{s_i}(WR_i y)$) is below $nC^2\sigma^2$ [35] (the error term decreases to zero, as $s_i \nearrow n$), where C is a fixed parameter. This requires repeating the least squares update (9) of x_i for each i at various sparsity levels incrementally, to determine the level at which the error term falls below the required threshold.

We propose an efficient way to do this: add one non-zero element at a time from $WR_i\, y$ (elements chosen in descending magnitude ordering) to α_i in (9), until the error measure $\|R_i\, y - x_i\|_2^2$ with the newly updated x_i falls below the required threshold. The matrix G in (9) can be pre-computed, and the addition of a new non-zero element to α_i leads to x_i being updated by adding a column of G scaled by the new non-zero element.

Once the variable sparsity levels s_i are chosen for all i, we use the new s_i's back in the transform learning step, and iterate over the learning and variable sparsity update steps, which leads to a better denoising performance compared to one iteration. In the final iteration, the x_i's that are computed (satisfying the $\|R_i\, y - x_i\|_2^2 \leq nC^2\sigma^2$ condition) represent the denoised patches.

3) Denoised Image Update: Once the denoised patches x_i are found using the proposed scheme, they are restricted to their range (e.g., 0-255), and averaged at their respective image locations to produce the denoised image x. The image denoising algorithm is summarized in Figure 1.

We work with mean subtracted patches during optimization, and the means are added back to the final denoised patches. We also typically learn on a subset of all patches selected uniformly at random [38]. In this case, the update of s_i's is only performed on a subset of patches, except in the final denoising iteration when all the s_i's are updated.

In Problem (P4), we denoise the image patches without directly enforcing the constraint $R_i x = x_i$. In the end, we obtain the least-square solution for x in the set of equations $R_i x = x_i$. This results in the averaging of patches that produces x. This technique, although sub-optimal, results in a highly efficient algorithm. As will be shown in Section V, the proposed algorithm indeed provides promising denoising performance at a low computational cost.

V. Numerical Experiments

A. Framework

We present results demonstrating the promise of our doubly sparse transform learning framework for image representation and image denoising. We first illustrate the convergence of the cost function and iterates for our alternating algorithms. Next, we demonstrate the effect of different choices of the analytical transform Φ. We then show that highly sparse transforms B can be learnt for natural images with only a marginal loss in image representation quality as compared to unstructured transforms, and in fact, with dramatic gains in speed of learning. The doubly sparse transforms will be shown to be substantially better than analytical ones such as the DCT for image representation. The representation performance of adaptive transforms will be shown to improve with increasing patch size. Moreover, the doubly sparse transforms will be shown to require fewer training signals for learning than unstructured transforms. We also discuss the generalizability of learnt doubly sparse transforms and their promise for image compression[4]. Finally, we demonstrate the promise of learnt doubly sparse transforms in image denoising, where they perform comparably, or better than learnt overcomplete synthesis dictionaries, while being much faster.

We work with the l_0 quasi norm for sparsity of X and B in the experiments (i.e., Problem (P2)), but this can be easily substituted by alternative sparsity penalties. We use a fixed step size in the transform update step of our algorithms. In our experiments, we initialize learning with the identity matrix for B, which we found to work best [5]. All implementations were coded in Matlab v7.8 (R2009a).

The data is generated as patches of natural images. We employ our doubly sparse transform learning Problem formulations for learning adaptive sparse representations of these patches. The means (or DC values) of the patches are removed and we only sparsify the mean-subtracted patches which are stacked as columns of Y (patches represented as vectors). The means are added back for image display. Mean removal is typically adopted in image processing applications such as K-SVD-based image denoising.

Performance Metrics: We introduce several metrics to evaluate the quality of learnt transforms. We define the *normalized sparsification error* (NSE) as $\|B\hat{Y} - X\|_F^2 / \|B\hat{Y}\|_F^2$. This measures the fraction of energy lost in sparse fitting. In other words, it indicates the degree of energy compaction achieved in the transform domain, or how well the doubly sparse transform model holds for the signals. This is an interesting property to observe for the adaptive transforms. We also define the *normalized recovery error* as $\left\|Y - W^{-1}X\right\|_F^2 / \|Y\|_F^2$. This measures the normalized error in recovering the data Y as $W^{-1}X$ from their sparse codes X obtained by projecting $WY = B\Phi Y$ onto the ℓ_0 ball (2). This metric serves as a good surrogate for the performance of the learnt transform in a compression application. A closely related metric is the recovery peak signal to noise ratio (or *recovery PSNR* (RP)) defined as $255\sqrt{P} / \left\|Y - W^{-1}X\right\|_F$ in dB, where P is the number of image pixels. The *condition number* (CN) of $W = B\Phi$ and the *sparsity* of B are the other measures of usefulness and efficiency of the doubly sparse transform. An important parameter in our results is the transform sparsity fraction $\beta = \frac{r}{n^2}$, which we express in this section as a percentage.

B. Convergence and Learning

For the first experiment, we extract the non-overlapping patches of size $\sqrt{n} \times \sqrt{n} = 8 \times 8$ ($n = 64$) from the 512×512 Barbara image [27]. The data matrix Y in this case has 4096 training signals (patches represented as vectors), and we work with $s = 11$. We fix the transform Φ in the experiment to be the 2D DCT (i.e., $\Phi = \Phi_0 \otimes \Phi_0$, where Φ_0 is the $\sqrt{n} \times \sqrt{n}$, or the 8×8 1D DCT matrix, and "$\otimes$" denotes the Kronecker product). Problem (P2) is solved to learn a sparse transform B that is adapted to the image patches. The algorithm parameters are $\lambda = 4 \times 10^5$, and $r = 0.25 \times n^2$ ($\beta = 25\%$). The conjugate gradient algorithm for transform update was run for 30 iterations with a fixed step size of 2×10^{-9}.

Figure 2 shows the progress of the algorithm over iterations. The objective function, sparsification error, and condition number are plotted over the iterations in Figures 2(a), 2(b), and 2(c) respectively. They all converge quickly. The condition number of B, which is also the condition number of $W = B\Phi$ for orthonormal Φ, converges to a low value of 1.41, indicating well-conditioning of the learnt transform. The horizontal line in the sparsification error plot corresponds to the sparsification error of the patch-based 2D DCT (i.e., $W = \Phi = DCT$) at $s = 11$. Our algorithm improves/decreases the sparsification error by 5.8 dB compared to the analytical DCT.

The normalized sparsification error and normalized recovery error for the learnt transform $W = B\Phi$ (with X obtained by projecting WY onto the ℓ_0 ball (2)) are 0.0450 (or 4.5%) and 0.0474, respectively. The two errors are similar for well-conditioned transforms. On the other hand, the normalized errors (both sparsification and recovery) corresponding to the 2D DCT for the zero mean Y are worse, at 0.0676. The results

[4]The study of a complete compression scheme involving learnt transforms is beyond the scope of this work. Instead, we use the quality of sparse image representation provided by the learnt transforms as a surrogate to indicate their potential for compression.

[5]Initializing the sparse matrix B with identity is equivalent to initializing the doubly sparse matrix W with the analytical transform Φ, which already provides reasonable sparsification.

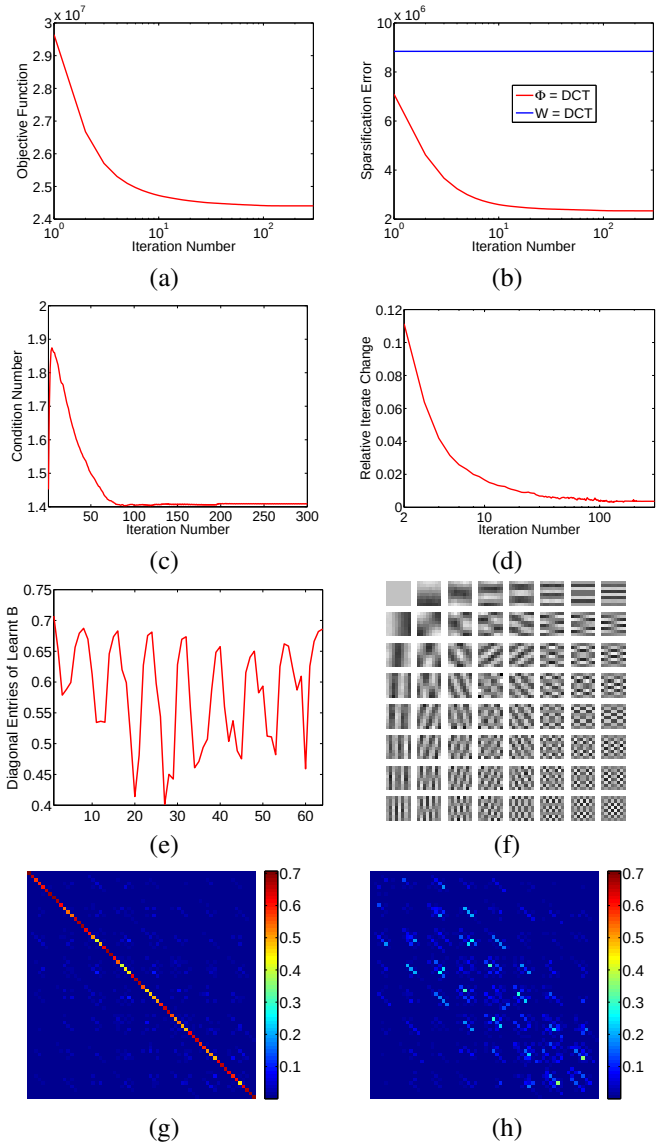


Fig. 2. Behavior of the doubly Sparse Learning Algorithm for (P2): (a) Objective function, (b) Sparsification error when $\Phi = DCT$, along with the sparsification error (horizontal line) of the DCT by itself, (c) Condition number of B, (d) Relative iterate change, (e) Diagonal entries of the learnt sparse B, (f) Rows of the learnt transform $W = B\Phi$ shown as patches, (g) Magnitude of the symmetric part ($\frac{B+B^T}{2}$) of the learnt B, (h) Magnitude of skew-symmetric part ($\frac{B-B^T}{2}$) of the learnt B.

here indicate that image patches admit reasonably good doubly sparse transform representations.

The sparse approximation image recovered as $W^{-1}X$ using the learnt W, has a recovery PSNR of 34.39 dB. In contrast, the image recovered using the fixed, patch-based 2D DCT (i.e., $W = \Phi = DCT$) has a worse PSNR of 32.85 dB.

The learnt sparse matrix B has the structure of a positive diagonal matrix (Figure 2(e) shows the diagonal elements of B) with added off-diagonal perturbation. The off-diagonal perturbation represents the modification to the analytical Φ, and is approximately skew-symmetric (Figures 2(g) and 2(h) show most of the off-diagonal energy in the skew-symmetric part of the learnt B).

Figure 2(f) shows the learnt transform $W = B\Phi$, with each row shown as an 8×8 patch called the 'transform atom'.

Φ	CN-I	NSE-I	RP-I	NSE-F	RP-F
DCT	1.40	0.0456	34.37	0.0676	32.85
Hadamard	1.50	0.0467	34.24	0.1156	30.52
Wavelets	1.65	0.0574	33.48	0.1692	28.86
Identity	2.92	0.1193	31.12	0.5145	24.03

TABLE I
CONDITION NUMBERS (CN-I), NORMALIZED SPARSIFICATION ERRORS (NSE-I), AND RECOVERY PSNRS (RP-I) FOR THE (BARBARA) IMAGE-SPECIFIC TRANSFORMS FOR VARIOUS Φ MATRICES AT $\beta = 25\%$, ALONG WITH THE NORMALIZED SPARSIFICATION ERRORS (NSE-F) AND RECOVERY PSNRS (RP-F) FOR THE FIXED TRANSFORMS Φ.

The learnt doubly sparse transform exhibits geometric and frequency like structures, that sparsify Barbara.

Finally, we plot in Figure 2(d), the evolution of $\|B_i - B_{i-1}\|_F / \|B_{i-1}\|_F$, where i denotes the algorithm iteration number. This quantity measures the relative change between successive iterates/transforms. It quickly decreases to a low value indicating convergence of the iterates.

Stopping Condition and Initialization. The relative iterate change can be used as a stopping condition for the algorithm for (P2). Hence, in the following experiments, the algorithm terminates when the relative change of successive iterates (B) is less than a small threshold empirically chosen as 0.1%. We have observed that a smaller threshold increases the iteration count while providing only marginal improvement in the results.

Additionally, in the following experiments, the initial (40) iterations of the algorithm are executed with no post-thresholding of the transform in the transform update step. This usually results in better initialization, and thus convergence to better solutions.

C. Performance For Different Φ Matrices

We now test the performance of our algorithm for (P2) on the data of Figure 2 for different choices of the matrix Φ. All algorithm parameters are the same as for the experiment of Figure 2.

Table I shows the normalized sparsification errors (NSE-I), recovery PSNRs (RP-I for Barbara), and condition numbers (CN-I) for the learnt transforms (also referred to as *image-specific transforms* since they are adapted to a specific image) with 2D DCT, 2D Hadamard, 2D Haar Wavelets[6], and identity matrix as Φ matrices. The table also lists the performance metrics (abbreviated as NSE-F and RP-F) for the various Φ matrices themselves at the same sparsity level s as in the experiment of Figure 2.

All the learnt doubly sparse transforms $W = B\Phi$ provide better normalized sparsification errors and recovery PSNRs than the corresponding analytical transforms Φ. The learnt doubly sparse transforms with the 2D DCT and 2D Hadamard Φ matrices differ only slightly in their sparsification, recovery

[6]Here, the 2D Haar transform matrix Φ is obtained as the Kronecker product of two 1D 3-level (since $\log_2(\sqrt{n}) = 3$) Haar matrices. Alternatively, one could perform the 3-level 2D Haar transformation as a sequence of three 1-level transformations, each obtained by using the Kronecker product of two 1D 1-level Haar matrices. In this case, the 1-level transformation in each step of the sequence is applied only on the approximation coefficients (in the first step, this is the entire image patch) from the previous step. However, this alternative construction of the Wavelet Φ led to inferior performance metrics, and is hence not included in Table I.

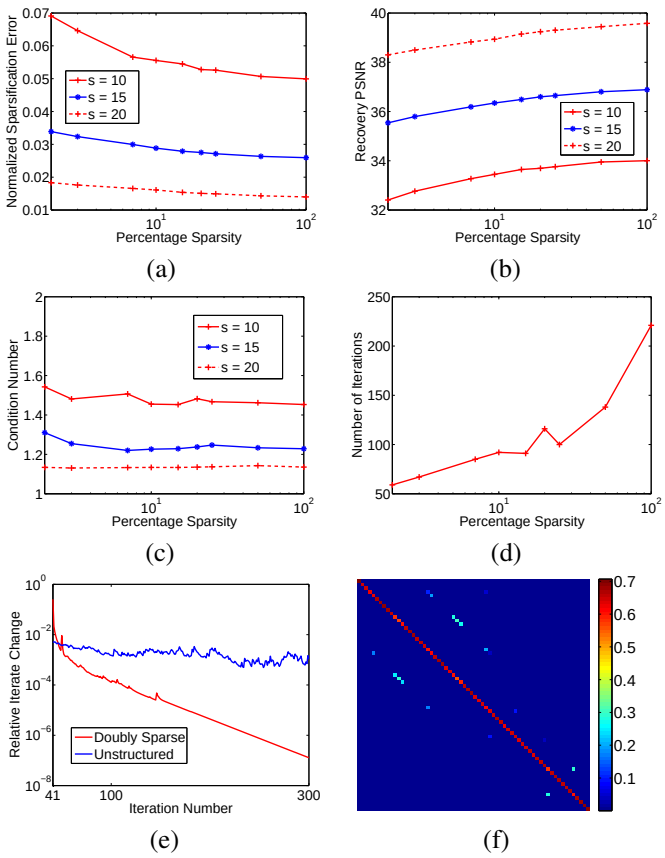


Fig. 3. Behavior of algorithm as function of β for different values of s: (a) Normalized Sparsification error, (b) Recovery PSNR, (c) Condition number, (d) Algorithm iteration count for $s = 15$, (e) Relative iterate change vs. iteration count for $s = 15$ at transform sparsity fractions $\beta = 2\%$ and $\beta = 100\%$ (i.e., unstructured), (f) Magnitude of learnt B at $\beta = 2\%$.

performance, although the 2D DCT matrix itself performs significantly better than Hadamard. Importantly, the operations involving the Hadamard matrix are faster since its entries are ± 1. Therefore, the doubly sparse learning allows us to exploit the inexpensive but poor Hadamard transform without loss of performance.

The results with $\Phi = \text{Wavelets}$ are quite inferior to both the DCT and Hadamard for the Barbara image. Furthermore, among the various learnt doubly sparse transforms, the one learnt with identity as Φ matrix performs the worst. Since this case essentially corresponds to a 'self-sparse' transform (i.e., the transform $W = B\Phi = B$ is sparse), the results indicate that doubly sparse transforms can perform significantly better than a self-sparse transform. On the other hand, the learnt self-sparse transform is significantly better than the identity matrix, which by itself provides an unacceptable normalized error and recovery PSNR of 51% and 24 dB, respectively.

D. Algorithm Behavior as Function of Parameters

*1) **Performance as Function of** s **and** β:* We now work with the same data as for Figure 2, and test the performance of the algorithm for (P2) as a function of the data sparsity level s and transform sparsity fraction β. All other parameters are fixed as in the experiment of Figure 2 (except that a larger step size of 10^{-8} is used for large values of β, such as 50%, to ensure faster algorithm convergence).

	$s = 10$	$s = 15$	$s = 20$
NSE	0.0782	0.0393	0.0211
RP	32.22	35.21	37.91

TABLE II
NORMALIZED SPARSIFICATION ERRORS AND RECOVERY PSNRS FOR THE BARBARA IMAGE WITH THE PATCH-BASED 2D DCT.

Figure 3 plots the performance metrics of the learnt transform as a function of the transform sparsity fraction β, at different data sparsity levels $s = 10, 15, 20$. The condition number (Figure 3(c)) shows little change as a function of β. However, the normalized sparsification error (Figure 3(a)) increases monotonically and the recovery PSNR (Figure 3(b)) decreases as the transform sparsity fraction β is reduced. This behavior is expected because as its sparsity level decreases, the transform has fewer degrees of freedom to adapt (i.e., the learning is more constrained at lower transform sparsity levels). Likewise, all the metrics also degrade when the data sparsity level s is reduced. However, the values of all the metrics (for fixed s) are reasonable even at low transform sparsity levels such as β of 10-15 %, in the sense that the values at 10% are not too different from the corresponding values at 100%. This indicates that B can be constrained to be sparse with only a marginal loss in the performance metrics. A good choice of β would also depend on the transform Φ, since some Φ matrices sparsify the data much better than others.

For comparison, Table II lists the performance metrics obtained with the 2D DCT for the same data. For a particular sparsity level s, the learnt doubly sparse transforms have better normalized sparsification errors and recovery PSNRs (as seen in Figure 3) than the 2D DCT even at very low transform sparsities such as 2%. This indicates the promise of efficient adaptive transforms over analytical ones.

Next, we turn to computational cost. We plot the number of algorithm iterations (Figure 3(d)) required to reach the stopping condition, as a function of β for $s = 15$. (The behavior is similar for other s values.) Depending on the value of β, the number of required algorithm iterations is reduced 2-4 fold compared to the unstructured transform case, i.e., 100% transform sparsity. (This reduction is not an exact monotone function of β, which is possibly an artifact due to the specific choice of the stopping criterion.) In general, the reduction is even greater with careful choice of parameters. In order to better illustrate the accelerated convergence of doubly sparse transform learning, we also plot the relative iterate change itself over the iterations (Figure 3(e)) at $\beta = 100\%$, and another low $\beta = 2\%$. The convergence rate of the iterates is seen to be much better for the latter doubly sparse case. The learnt sparse B at $\beta = 2\%$ is also shown (Figure 3(f) shows the magnitudes of elements of B). Thus, doubly sparse transforms, which have fewer free parameters, can in general be learnt much faster than the unstructured transforms.

At smaller values of β, the per-iteration computational cost is also lower, since the operations involving sparse matrices are faster. Currently, the run time per iteration is up to 2 times lower at smaller values of β compared to $\beta = 100\%$. We expect the speedups to be much greater with more efficient implementation of sparse matrix operations, conversion of the code to C/C++, and code optimization.

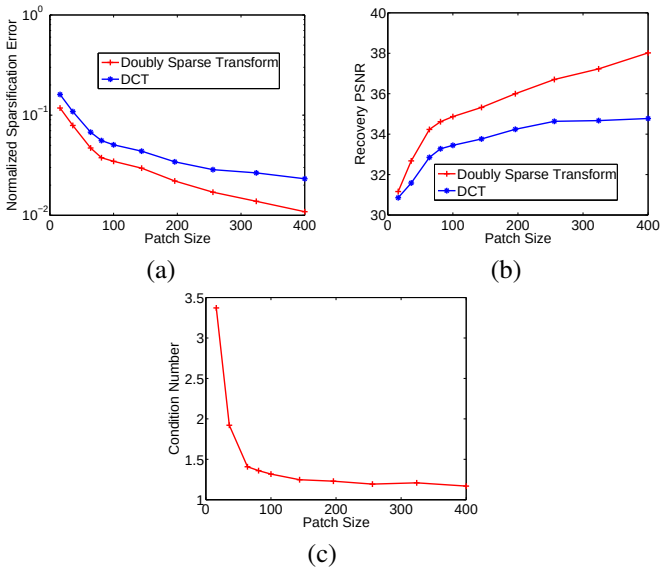


Fig. 4. Behavior of our algorithm as function of patch size n at $\beta = 15\%$: (a) Normalized Sparsification error for the doubly sparse transform and 2D DCT, (b) Recovery PSNR for the doubly sparse transform and 2D DCT, (c) Condition number for the doubly sparse transform.

When $r = n$ ($\beta = 1/n$), we can only learn doubly sparse transforms W whose rows are scaled versions of the rows of the corresponding Φ (rows of Φ that sparsify the data better are likely to get larger scalings than those that sparsify worse), due to the log determinant penalty in (P2), which needs to be finite. Moreover, when the condition number of B is 1, such learnt transforms are merely scalar multiples of Φ. Therefore, as r approaches n with the condition number of B kept close to 1, the normalized sparsification errors and recovery PSNRs of the learnt doubly sparse transforms would approach the corresponding values for the analytical transform Φ. The results of Table II and Figure 3 corroborate this.

*2) **Performance as Function of** λ:* The behaviour of the algorithm for the 'unstructured' Problem (P1) was studied as a function of the parameter λ (with $\lambda = \mu$) in our previous work [17]. It was shown that the normalized sparsification error increases with λ, while the condition number decreases. The doubly sparse transform learning algorithm for (P2) too has a similar behavior with respect to λ. Hence, a separate study of this parameter is omitted in this work. For natural images, the recovery PSNR using the learnt transform is typically better at λ values corresponding to intermediate conditioning or 'well-conditioning', rather than unit conditioning, since unit conditioning is too restrictive [17].

*3) **Performance as Function of Patch Size**:* Next, we test the performance of the algorithm for (P2), as a function of patch size n for the Barbara image. Non-overlapping patches are extracted at each patch size. We work with a transform sparsity level of $r = 0.15 \times n^2$, and data sparsity level of $s = 0.17 \times n$ (values rounded to nearest integers). All other parameters are fixed as in the experiment of Figure 2.

Figure 4 plots the performance metrics for both the learnt doubly sparse transform and the 2D DCT, as a function of patch size. Both normalized sparsification error (Figure 4(a)) and recovery PSNR (Figure 4(b)) improve much more rapidly with increasing patch size for the learnt doubly sparse transform than for the 2D DCT. In fact, the normalized sparsification error and recovery PSNR saturate for the DCT at larger patch sizes. These results indicate that doubly sparse image-specific transforms provide even better representations at larger patch sizes [7]. Furthermore, the condition number of the learnt transforms (Figure 4(c)) is close to 1 at all patch sizes (although the conditioning is slightly worse at low patch sizes such as $n = 16$ for the chosen parameter values). The sparsification and recovery quality improvements of the learnt transform over the 2D DCT can be further increased at each patch size with optimal choice of parameters (e.g., λ).

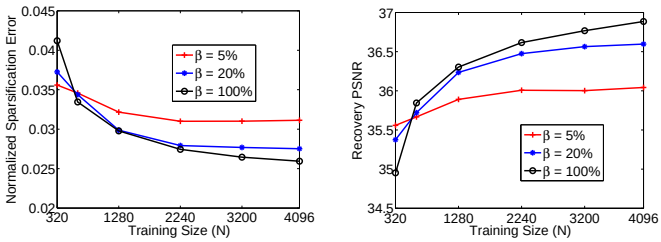


Fig. 5. Behavior of our algorithm as function of training size N at $s = 15$: (a) Normalized Sparsification error for β of 5%, 20%, and 100%, (b) Recovery PSNR for β of 5%, 20%, and 100%.

*4) **Performance as Function of training size** N:* We now investigate the behavior of the algorithm for (P2), as a function of the size of the training set (N), at various transform sparsity fractions β. The data sparsity level is set as $s = 15$, and all other algorithm parameters except λ are fixed as in the experiment of Figure 3. The parameter λ is tuned at each N to enable condition numbers (for the learnt transforms) similar to the experiment of Figure 3 (or, in other words, the $N = 4096$ case), at $s = 15$. We also select a random subset of the non-overlapping patches of the Barbara image at each N and β, for training.

Figure 5 plots the normalized sparsification error and recovery PSNR metrics (computed over *all* non-overlapping patches) for the learnt transform as a function of training size N at various values of β. At smaller values of β, much smaller training sizes suffice to provide performance metrics similar to the $N = 4096$ case (i.e., full set of training patches). We conjecture that this is because doubly sparse transforms have far fewer free parameters, and hence, the learning requires less training signals (than unstructured transforms) to learn the optimal transform for particular data. Thus, doubly sparse transform learning can provide additional speedups in applications by utilizing smaller training sizes compared to unstructured transforms. Moreover, when N is sufficiently small, the plots of Figure 5 indicate that the sparsification and recovery performance of unstructured transform learning degrade much quicker than that of doubly sparse transform learning. Therefore, we conclude that the proposed doubly

[7] We believe this effect is due to the fact that at larger patch sizes, the adapted transforms can capture high-level (or more noticeable) features of the data. On the other hand, at small patch sizes, the learning can only extract 'coarse' or low-level features. The behavior with patch size is also partly owing to the reduction in the total number of training patches with increasing n, which leads to greater adaptivity to smaller training sets.

Fig. 6. The test images used for denoising.

sparse transform learning via (P2) is robust to the size of the training set.

*5) **Generalizability of Doubly Sparse Transforms**:* While we considered adapting a transform to a specific image, a doubly sparse ('global') transform learnt over a class of images (e.g., magnetic resonance images), and tested on unrelated test images, also provides better representations than fixed transforms such as the DCT. Such global doubly sparse transforms in fact, generalize better than global unstructured (or non-sparse) transforms (cf. [56] for details). These properties make the efficient doubly sparse transform particularly attractive for compression.

E. Preliminary Results for Image Denoising

We present preliminary results for our image denoising framework (Problem (P4)). We consider 6 different test images (Figure 6) and simulate i.i.d. gaussian noise at 5 different noise levels ($\sigma = 5, 10, 15, 20, 100$) for each of the images. All images except Cameraman (256×256) and Man (768×768) have 512×512 pixels. We use the proposed adaptive sparsifying-transform-based denoising algorithm (Section IV) to obtain denoised versions of the noisy images. We compare results and run times obtained by our adaptive transform-based algorithm with those obtained by adaptive overcomplete K-SVD denoising [35]. The goal is to study the potential of the proposed transform-based models (both unstructured and doubly sparse) in comparison to the dictionary-based models. The Matlab implementation of K-SVD denoising [35] available from Michael Elad's website [57] was used in our comparisons, and we used the built-in parameter settings of that implementation. Computations were performed with an Intel Core i5 CPU at 1.7GHz and 6GB memory, employing a 64-bit Windows 8 operating system.

We consider adaptive transform denoising with three different doubly sparse transforms of size 64×64, 81×81, and 121×121, each with a transform sparsity fraction $\beta = 20\%$. The corresponding patch sizes are 8×8, 9×9, and 11×11, respectively. The higher patch sizes will be used to demonstrate the scalability of doubly sparse denoising. We also learn a 64×64 unstructured transform for denoising. Since doubly sparse transforms are highly efficient, we will compare the larger doubly sparse transforms (e.g., 121×121) with the 64×64 unstructured transform, and study the trade-off between denoising and run time.

We choose Φ to be the 2D DCT in our experiments. We set $\lambda = 32 \times 10^6$ for doubly sparse transform learning (via (P2)), and the initial (during the first iteration of the denoising algorithm) fixed data sparsity level is $s = 0.15 \times n$ (rounded to nearest integer). In the transform update step within learning, the conjugate gradient method is run for 30 iterations. We also set a maximum iteration count in transform learning, which works together with the previously discussed stopping condition in Section V-B. The maximum number of allowed iterations is lower at high noise levels, to prevent overfitting to noise. The number of initial learning iterations with no post-thresholding of the transform in the transform update step, is also set appropriately. The size of the training set used for doubly sparse learning is 22400. The training patches are chosen uniformly at random from all overlapping patches in each denoising iteration. As the patch size n increases, the fraction of all patches used in training increases. We observed that different random choices of training subsets provide denoising PSNRs that typically differ only by few hundredths of a dB.

For the unstructured transform learning case, the training set size is increased 1.43 fold compared to the doubly sparse cases[8]. Likewise, the maximum iteration count in learning is increased 2.5 fold for the unstructured case, since convergence of unstructured transform learning is slower. The parameter $\lambda = 45 \times 10^6$ ($\mu = \lambda$) for unstructured learning.

The denoising algorithm parameter $\tau = 0.01/\sigma$. The parameter $C = 1.08$ at $n = 64$, whereas it is set as 1.07 and 1.04 at $n = 81$ and $n = 121$, respectively. The denoising algorithm is executed for 4 iterations at lower noise levels ($\sigma = 5, 10$), 8 iterations at intermediate noise levels ($\sigma = 15, 20$). At very high noise ($\sigma = 100$), we use only 1 iteration, which saves run time without degrading denoising performance.

Table III lists the denoising PSNRs obtained by the doubly sparse and unstructured schemes, together with the PSNRs obtained by K-SVD. For all images and noise levels considered, the best PSNRs are provided by our transform-learning-based (either doubly sparse or unstructured) schemes. The best improvement over K-SVD averaged over all rows of Table III is 0.134 dB. Moreover, among the transform-based schemes, the best PSNRs are provided in most cases by the 121×121 doubly sparse transform [9]. The 64×64 doubly sparse transform also provides better PSNRs than K-SVD for Cameraman, Couple, and Man. For Barbara, Hill, and Lena, it provides PSNRs comparable to K-SVD in many cases. All the transforms considered have far fewer actual free parameters than the K-SVD dictionary. The 64×64 and 121×121 doubly sparse transforms have 20 fold and 5.6 fold fewer free parameters than the 64×256 K-SVD dictionary, respectively. Thus, the transform-based scheme (P4) provides comparable, or better PSNRs than K-SVD despite the highly constrained transforms.

[8]This training set size is still smaller than that used by K-SVD [35], [57], since the overcomplete K-SVD has more unknown parameters.

[9]We believe the larger transform denoises better, since it captures larger scale image features.

Image	σ	K-SVD	Unstr-uctured	Doubly Sparse		
				$n = 64$	$n = 81$	$n = 121$
1	5	38.08	38.16	38.14	38.20	**38.25**
	10	34.41	34.36	34.37	34.43	**34.53**
	15	32.33	32.09	32.06	32.20	**32.37**
	20	30.83	30.53	30.53	30.71	**30.90**
	100	21.87	21.91	21.91	22.06	**22.26**
2	5	37.81	**38.04**	37.91	37.86	37.82
	10	33.72	**33.91**	33.81	33.81	33.80
	15	31.50	**31.68**	31.58	31.61	31.61
	20	29.82	29.90	29.83	29.86	**29.91**
	100	21.76	21.79	21.82	21.93	**22.06**
3	5	37.28	**37.33**	37.30	37.30	37.30
	10	33.51	33.62	33.59	33.61	**33.64**
	15	31.46	31.51	31.47	31.50	**31.59**
	20	30.02	30.03	30.02	30.04	**30.14**
	100	22.57	22.58	22.59	22.69	**22.78**
4	5	36.47	**36.68**	36.64	36.64	36.64
	10	32.71	**32.97**	32.92	32.92	32.93
	15	30.78	30.98	30.96	30.96	**30.99**
	20	29.40	29.57	29.54	29.55	**29.60**
	100	22.76	22.89	22.90	22.99	**23.01**
5	5	37.08	**37.10**	37.04	37.04	37.08
	10	33.45	33.43	33.40	33.44	**33.49**
	15	31.52	31.49	31.47	31.50	**31.57**
	20	30.17	30.10	30.09	30.13	**30.23**
	100	23.98	23.81	23.79	23.94	**24.00**
6	5	38.61	**38.64**	38.57	38.60	38.63
	10	35.49	35.52	35.47	35.52	**35.56**
	15	33.74	33.69	33.67	33.75	**33.80**
	20	32.41	32.27	32.27	32.35	**32.44**
	100	24.51	24.31	24.29	24.49	**24.60**

TABLE III
PSNR VALUES FOR DENOISING FOR OUR ALGORITHM USING 64×64, 81×81, AND 121×121 DOUBLY SPARSE TRANSFORMS AT $\beta = 20\%$. OUR RESULTS ARE COMPARED WITH THOSE OBTAINED USING A 64×64 UNSTRUCTURED TRANSFORM, AND A 64×256 OVERCOMPLETE K-SVD DICTIONARY [35]. THE BEST PSNRS ARE MARKED IN BOLD. THE IMAGES NUMBERED 1 THROUGH 6 CORRESPOND TO BARBARA, CAMERAMAN, COUPLE, MAN, HILL, AND LENA, RESPECTIVELY.

Next, we compare the denoising performance of the doubly sparse transforms to the unstructured transform. The 64×64 doubly sparse transform denoises almost identically to the 64×64 unstructured transform for most images (barring Cameraman). In fact at $\sigma = 100$, it denoises slightly better than the unstructured version (on an average). Thus, the lower parametrization of the doubly sparse transforms prevents over-fitting to noise. Furthermore, on comparing the best denoising PSNRs provided by doubly sparse transforms of all sizes with those of the 64×64 unstructured transform (for each image and noise level), we obtain an average PSNR improvement of 0.0913 dB for the doubly sparse transforms. The 121×121 doubly sparse transform, although larger than the 64×64 unstructured transform still has fewer free parameters than the latter. Thus, doubly sparse transforms with fewer parameters provide better denoising than unstructured transforms.

Turning to Table IV, we can also ascertain the computational advantages of doubly sparse denoising. Table IV lists the average speedups over K-SVD for the various transform-based denoising schemes. For each image and noise level, the ratio of the run times of K-SVD denoising and transform-based denoising is computed, and the speedups are averaged over the images at each noise level.

σ	Unstructured ($n = 64$)	Doubly Sparse ($n = 64$)	Doubly Sparse ($n = 81$)	Doubly Sparse ($n = 121$)
5	4.8	18.0	11.6	5.6
10	3.3	13.8	9.1	4.3
15	1.7	7.5	5.0	2.3
20	1.2	5.6	3.7	1.7
100	3.0	6.4	4.9	3.0

TABLE IV
THE DENOISING SPEEDUPS OVER K-SVD [35] FOR THE VARIOUS TRANSFORM LEARNING SCHEMES. THE SPEEDUPS ARE AVERAGED OVER THE SIX IMAGES AT EACH NOISE LEVEL.

It is clear that both unstructured and doubly sparse transforms provide speedups over K-SVD. While the 64×64 unstructured transform provides a speedup of upto 4.8x over K-SVD, the speedups for the doubly sparse cases range upto 18x in Table IV. The speedups over K-SVD for specific images can be much higher. For example, for the Cameraman image at $\sigma = 5$, the run times for denoising with K-SVD, the 64×64 unstructured transform, 64×64 doubly sparse transform, 81×81 doubly sparse transform, and 121×121 doubly sparse transform are 15 minutes, 2.5 minutes, 35 seconds, 55 seconds, and 1.9 minutes, respectively, indicating a speedup of 25.7x for the 64×64 doubly sparse transform over K-SVD[10]. The results of Table IV also indicate that the doubly sparse transform model scales well with patch size.

Now, combining the conclusions from Tables III and IV, we see that the 64×64 doubly sparse transform while denoising comparably to the unstructured one, is much faster (3.83x faster on average over all noise levels) than the latter. The 121×121 doubly sparse transform which denoises better than the 64×64 unstructured one, also does so more efficiently.

We show some doubly sparse denoising results in Figure 7, that illustrate the effectiveness of the proposed algorithm to denoise images.

All of our learnt transforms are well-conditioned (condition numbers 1-3). Poorly conditioned transforms were empirically observed to degrade denoising performance. Although we presented denoising results at 20% transform sparsity, the denoising performances are reasonable even at lower transform sparsities such as 10%, where the speedups over previous methods is even greater.

Comparison to Other Methods.

The recent overcomplete analysis-dictionary-based image denoising method of Yaghoobi et al. [46] was shown by the authors to denoise worse than K-SVD. Hence, we have only included the comparison to K-SVD here.

A method closely related to our proposed approach is the synthesis double sparsity method of Rubinstein et al. [50]. However, for 2D image denoising, it was shown by the authors in [50] (also confirmed in our own experiments with the author's code [58]) that the synthesis double sparsity method, although faster than standard K-SVD [35], typically falls behind in denoising performance. Hence, we only compare to the (better) unconstrained K-SVD [35] in this work. Synthesis

[10]We anticipate greater speedups with optimized implementation of sparse matrix operations.

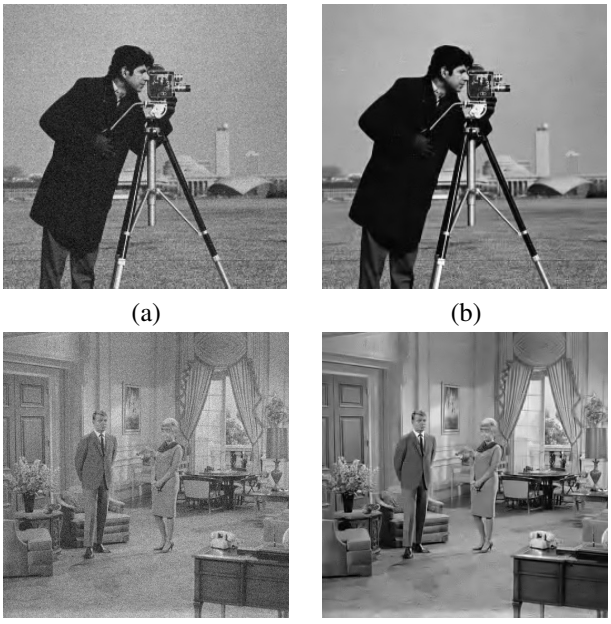


Fig. 7. Denoising: (a) Noisy ($\sigma = 10$) cameraman image (PSNR = 28.14 dB), (b) Denoised cameraman image (PSNR = 33.81 dB) using 64×64 transform at $\beta = 20\%$, (c) Noisy ($\sigma = 20$) couple image (PSNR = 22.11 dB), (d) Denoised couple image (PSNR = 30.14 dB) using 121×121 transform at $\beta = 20\%$.

double sparsity has been shown to be more benefical for 3D applications such as 3D denoising [50]. We plan to study such applications in the near future. Note that the speedups over K-SVD shown in Table IV for the 2D image cases in this paper, can only increase, upon switching to 3D data. This is because the learning via K-SVD is significantly slower than transform learning (Section III-C) at larger problem sizes [17].

The denoising PSNRs presented for our algorithm can be further improved by optimal choice of parameters. Our experiments thus demonstrate the promise of learnt doubly sparse transforms in image denoising. The results with just the adaptive square transforms are comparable to, or better than with overcomplete dictionaries. Doubly sparse transforms denoise as well or better than unstructured transforms, but do so much faster.

We emphasize that the transforms considered in this work are only square transforms. We conjecture that the denoising performance of our algorithms would improve further and become comparable to the state of the art (for example [59]) with overcomplete and multiscale extensions of transform learning (similarly to the synthesis case [38]).

VI. Conclusions and Future Work

In this paper, we presented novel problem formulations for learning square doubly sparse transforms. The proposed formulations give rise to significantly sparse and well-conditioned transforms with better sparsification errors and recovery PSNRs than analytical transforms. Moreover, imposing the doubly sparse property leads to faster learning and faster computations with the sparse transform. The adapted doubly sparse transform has a reduced storage requirement and generalizes better than the 'unstructured' (or non-sparse) transform. We also introduced a novel problem formulation for image denoising and demonstrated the promise of adaptive transforms in this application, with results competitive with overcomplete K-SVD. Most importantly, denoising with doubly sparse transforms is much cheaper. The usefulness of doubly sparse transform learning in image compression and other applications merits further study.

References

[1] M. Elad, P. Milanfar, and R. Rubinstein, "Analysis versus synthesis in signal priors," *Inverse Problems*, vol. 23, no. 3, pp. 947–968, 2007.

[2] A. M. Bruckstein, D. L. Donoho, and M. Elad, "From sparse solutions of systems of equations to sparse modeling of signals and images," *SIAM Review*, vol. 51, no. 1, pp. 34–81, 2009.

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

[4] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM J. Comput.*, vol. 24, no. 2, pp. 227–234, Apr. 1995.

[5] G. Davis, S. Mallat, and M. Avellaneda, "Adaptive greedy approximations," *Journal of Constructive Approximation*, vol. 13, no. 1, pp. 57–98, 1997.

[6] Y. Pati, R. Rezaiifar, and P. Krishnaprasad, "Orthogonal matching pursuit : recursive function approximation with applications to wavelet decomposition," in *Asilomar Conf. on Signals, Systems and Comput.*, 1993, pp. 40–44 vol.1.

[7] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.

[8] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE Trans. Information Theory*, vol. 55, no. 5, pp. 2230–2249, 2009.

[9] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 20, no. 1, pp. 33–61, 1998.

[10] I. F. Gorodnitsky, J. George, and B. D. Rao, "Neuromagnetic source imaging with FOCUSS: A recursive weighted minimum norm algorithm," *Electrocephalography and Clinical Neurophysiology*, vol. 95, pp. 231–251, 1995.

[11] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Annals of Statistics*, vol. 32, pp. 407–499, 2004.

[12] S. Nam, M. E. Davies, M. Elad, and R. Gribonval, "Cosparse analysis modeling - uniqueness and algorithms," in *ICASSP*, 2011, pp. 5804–5807.

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

[14] R. Rubinstein and M. Elad, "K-SVD dictionary-learning for analysis sparse models," in *Proc. SPARS11*, June 2011, p. 73.

[15] A. Chambolle, "An algorithm for total variation minimization and applications," *J. Math. Imaging Vis.*, vol. 20, no. 1-2, pp. 89–97, 2004.

[16] M. Yaghoobi, S. Nam, R. Gribonval, and M. E. Davies, "Noise aware analysis operator learning for approximately cosparse signals," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 5409–5412.

[17] S. Ravishankar and Y. Bresler, "Learning sparsifying transforms," *IEEE Trans. Signal Process.*, vol. 61, no. 5, pp. 1072–1086, 2013.

[18] S. Mallat, *A Wavelet Tour of Signal Processing*. Academic Press, 1999.

[19] E. J. Candès and D. L. Donoho, "Ridgelets: A key to higher-dimensional intermittency?" *Phil. Trans. R. Soc. Lond. A*, vol. 357, no. 1760, pp. 2495–2509, 1999.

[20] M. N. Do and M. Vetterli, "The contourlet transform: an efficient directional multiresolution image representation," *IEEE Trans. Image Process.*, vol. 14, no. 12, pp. 2091–2106, 2005.

[21] E. J. Candès and D. L. Donoho, "Curvelets - a surprisingly effective nonadaptive representation for objects with edges," in *Curves and Surfaces*. Vanderbilt University Press, 1999, pp. 105–120.

[22] W. K. Pratt, J. Kane, and H. C. Andrews, "Hadamard transform image coding," *Proc. IEEE*, vol. 57, no. 1, pp. 58–68, 1969.

[23] J. B. Allen and L. R. Rabiner, "A unified approach to short-time fourier analysis and synthesis," *Proc. IEEE*, vol. 65, no. 11, pp. 1558–1564, 1977.

[24] C. Lemaréchal and C. Sagastizábal, "Practical aspects of the moreau–yosida regularization: Theoretical preliminaries," *SIAM J. on Optimization*, vol. 7, no. 2, pp. 367–385, 1997.

[25] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 6583, pp. 607–609, 1996.

[26] K. Engan, S. Aase, and J. Hakon-Husoy, "Method of optimal directions for frame design," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1999, pp. 2443–2446.

[27] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on signal processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

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

[29] R. Gribonval and K. Schnass, "Dictionary identification–sparse matrix-factorization via l_1 -minimization," *IEEE Trans. Inform. Theory*, vol. 56, no. 7, pp. 3523–3539, 2010.

[30] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *J. Mach. Learn. Res.*, vol. 11, pp. 19–60, 2010.

[31] K. Skretting and K. Engan, "Recursive least squares dictionary learning algorithm," *IEEE Transactions on Signal Processing*, vol. 58, no. 4, pp. 2121–2130, 2010.

[32] 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.

[33] Q. Geng, H. Wang, and J. Wright, "On the local correctness of ℓ^1-minimization for dictionary learning," 2011, preprint: http://arxiv.org/abs/1101.5672.

[34] D. A. Spielman, H. Wang, and J. Wright, "Exact recovery of sparsely-used dictionaries," 2012, preprint: http://arxiv.org/abs/1206.5882.

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

[36] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *IEEE Trans. on Image Processing*, vol. 17, no. 1, pp. 53–69, 2008.

[37] M. Aharon and M. Elad, "Sparse and redundant modeling of image content using an image-signature-dictionary," *SIAM Journal on Imaging Sciences*, vol. 1, no. 3, pp. 228–247, 2008.

[38] J. Mairal, G. Sapiro, and M. Elad, "Learning multiscale sparse representations for image and video restoration," *SIAM Multiscale Modeling and Simulation*, vol. 7, no. 1, pp. 214–241, 2008.

[39] G. Yu, G. Sapiro, and S. Mallat, "Image modeling and enhancement via structured sparse model selection," in *Proc. IEEE International Conference on Image Processing (ICIP)*, 2010, pp. 1641–1644.

[40] H. Y. Liao and G. Sapiro, "Sparse representations for limited data tomography," in *Proc. IEEE International Symposium on Biomedical Imaging (ISBI)*, 2008, pp. 1375–1378.

[41] S. Ravishankar and Y. Bresler, "MR image reconstruction from highly undersampled k-space data by dictionary learning," *IEEE Trans. Med. Imag.*, vol. 30, no. 5, pp. 1028–1041, 2011.

[42] ——, "Multiscale dictionary learning for MRI," in *Proc. ISMRM*, 2011, p. 2830.

[43] G. Peyré and J. Fadili, "Learning analysis sparsity priors," in *Proc. Int. Conf. Sampling Theory Appl. (SampTA)*, Singapore, May 2-6, 2011. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00542016/

[44] M. Yaghoobi, S. Nam, R. Gribonval, and M. Davies, "Analysis operator learning for overcomplete cosparse representations," in *European Signal Processing Conference (EUSIPCO)*, 2011, pp. 1470–1474.

[45] B. Ophir, M. Elad, N. Bertin, and M. Plumbley, "Sequential minimal eigenvalues - an approach to analysis dictionary learning," in *Proc. European Signal Processing Conference (EUSIPCO)*, 2011, pp. 1465–1469.

[46] M. Yaghoobi, S. Nam, R. Gribonval, and M. E. Davies, "Constrained overcomplete analysis operator learning for cosparse signal modelling," *IEEE Trans. Signal Process.*, vol. 61, no. 9, pp. 2341–2355, 2013.

[47] M. W. Marcellin, M. J. Gormish, A. Bilgin, and M. P. Boliek, "An overview of JPEG-2000," in *Proc. Data Compression Conf.*, 2000, pp. 523–541.

[48] M. Lustig, D. Donoho, and J. Pauly, "Sparse MRI: The application of compressed sensing for rapid MR imaging," *Magnetic Resonance in Medicine*, vol. 58, no. 6, pp. 1182–1195, 2007.

[49] M. Lustig, J. M. Santos, D. L. Donoho, and J. M. Pauly, "k-t SPARSE: High frame rate dynamic MRI exploiting spatio-temporal sparsity," in *Proc. ISMRM*, 2006, p. 2420.

[50] R. Rubinstein, M. Zibulevsky, and M. Elad, "Double sparsity: Learning sparse dictionaries for sparse signal approximation," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1553–1564, 2010.

[51] B. Ophir, M. Lustig, and M. Elad, "Multi-scale dictionary learning using wavelets," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 5, pp. 1014–1024, 2011.

[52] M. Zibulevsky and B. A. Pearlmutter, "Blind source separation by sparse decomposition in a signal dictionary," *Neural Computation*, vol. 13, no. 4, pp. 863–882, 2001.

[53] R. Pytlak, *Conjugate Gradient Algorithms in Nonconvex Optimization*. Springer-Verlag, 2009.

[54] J. A. Fill and S. Janson, "Quicksort asymptotics," *Journal of Algorithms*, vol. 44, no. 1, pp. 4–28, 2002.

[55] J. Portilla, "Image restoration through l0 analysis-based sparse optimization in tight frames," in *IEEE International Conference on Image Processing (ICIP)*, 2009, pp. 3909–3912.

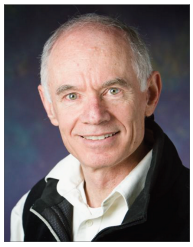
[56] S. Ravishankar and Y. Bresler, "Learning doubly sparse transforms for image representation," in *IEEE Int. Conf. Image Process.*, 2012, pp. 685–688.

[57] M. Elad, "Michael Elad personal page," http://www.cs.technion.ac.il/~elad/Various/KSVD_Matlab_ToolBox.zip, 2009.

[58] R. Rubinstein, http://www.cs.technion.ac.il/~ronrubin/Software/ksvdsbox11.zip, 2009, [Online; accessed 09-October-2012].

[59] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3D transform-domain collaborative filtering," *IEEE Trans. on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

Saiprasad Ravishankar received the B.Tech. degree in Electrical Engineering from the Indian Institute of Technology Madras, in 2008. He received the M.S. degree in Electrical and Computer Engineering, in 2010, from the University of Illinois at Urbana-Champaign, where he is currently a Ph.D candidate. His current research interests are in signal and image processing, and medical imaging.



Yoram Bresler received the B.Sc. (cum laude) and M.Sc. degrees from the Technion, Israel Institute of Technology, in 1974 and 1981 respectively, and the Ph.D degree from Stanford University, in 1986, all in Electrical Engineering. In 1987 he joined the University of Illinois at Urbana-Champaign, where he is currently a Professor at the Departments of Electrical and Computer Engineering and Bioengineering, and at the Coordinated Science Laboratory. Yoram Bresler is also President and Chief Technology Officer at InstaRecon, Inc., a startup he co-founded to commercialize breakthrough technology for tomographic reconstruction developed in his academic research. His current research interests include multi-dimensional and statistical signal processing and their applications to inverse problems in imaging, and in particular compressed sensing, computed tomography, and magnetic resonance imaging.

Dr. Bresler has served on the editorial board of a number of journals, and on various committees of the IEEE. Dr. Bresler is a fellow of the IEEE and of the AIMBE. He received two Senior Paper Awards from the IEEE Signal Processing society, and a paper he coauthored with one of his students received the Young Author Award from the same society in 2002. He is the recipient of a 1991 NSF Presidential Young Investigator Award, the Technion (Israel Inst. of Technology) Fellowship in 1995, and the Xerox Senior Award for Faculty Research in 1998. He was named a University of Illinois Scholar in 1999, appointed as an Associate at the Center for Advanced Study of the University in 2001-02, and Faculty Fellow at NCSA in 2006.