LEARNING CIRCULANT SENSING KERNELS

Yangyang Xu

Department of Computational and Applied Mathematics, Rice University
Houston, TX 77005, USA

Wotao Yin and Stanley Osher

Department of Mathematics, University of California
Los Angeles, CA 90095-1555, USA

Abstract. In signal acquisition, Toeplitz and circulant matrices are widely used as sensing operators. They correspond to discrete convolutions and are easily or even naturally realized in various applications. For compressive sensing, recent work has used random Toeplitz and circulant sensing matrices and proved their efficiency in theory, by computer simulations, as well as through physical optical experiments. Motivated by recent work [8], we propose models to learn a circulant sensing matrix/operator for one and higher dimensional signals. Given the dictionary of the signal(s) to be sensed, the learned circulant sensing matrix/operator is more effective than a randomly generated circulant sensing matrix/operator, and even slightly so than a (non-circulant) Gaussian random sensing matrix. In addition, by exploiting the circulant structure, we improve the learning from the patch scale in [8] to the much large image scale. Furthermore, we test learning the circulant sensing matrix/operator and the nonparametric dictionary altogether and obtain even better performance. We demonstrate these results using both synthetic sparse signals and real images.

1. **Introduction.** Compressive sensing (CS) ([7, 5]) acquires a compressible signal from a small number of linear projections. Let $\bar{x}$ denote an n-dimensional real or complex vector that is sparse under a certain basis Ψ, i.e., one can write $\bar{x} = \Psi\bar{\theta}$ with a sparse $\bar{\theta}$. Let $b := \Phi\bar{x}$ represent a set of m linear projections of $\bar{x}$. The basis pursuit problem

$$(1) \qquad \text{BP:} \quad \min_{\theta} \|\theta\|_1 \quad \text{s.t.} \quad \Phi\Psi\theta = b,$$

as well as several other methods, has been known to return a sparse vector θ and thus recover $x = \Psi\theta$ under certain conditions on the sensing matrix Φ and basis Ψ.

Gaussian random matrix Φ is in the first studied CS matrices and has nice properties. Given enough underdetermined measurements, (1) with Gaussian random matrix Φ is guaranteed to exactly recover a sparse $\bar{x}$ with high probability. However, in many CS applications, the acquisition of the linear projections $\Phi\bar{x}$ requires a physical implementation. In most cases, the use of an i.i.d. Gaussian random matrix Φ is either impossible or overly expensive. This motivates the study of easily implementable CS matrices. Two types of such matrices are the Toeplitz and circulant matrices, which have been shown to be almost *as effective as* the Gaussian

random matrix for CS encoding/decoding. A Toeplitz matrix T has the same element on each diagonal, and a circulant matrix C is a special Toeplitz matrix with each row obtained by shifting the preceeding row one element to the right, namely,

$$(2) \quad T = \begin{bmatrix} t_n & t_{n-1} & \cdots & \cdots & t_1 \\ t_{n+1} & t_n & t_{n-1} & \ddots & \vdots \\ t_{n+2} & t_{n+1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & t_n & t_{n-1} \\ t_{2n-1} & \cdots & t_{n+2} & t_{n+1} & t_n \end{bmatrix}, C = \begin{bmatrix} t_n & t_{n-1} & \cdots & \cdots & t_1 \\ t_1 & t_n & t_{n-1} & \ddots & \vdots \\ t_2 & t_1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & t_n & t_{n-1} \\ t_{n-1} & \cdots & t_2 & t_1 & t_n \end{bmatrix}.$$

When matrix T satisfies the additional property that $t_i = t_{n+i}, \forall i$, it becomes a circulant matrix C. For more about Toeplitz and circulant matrices, please refer to the review paper [11]. Since a (partial[1]) Toeplitz matrix has very similar theoretical and computational properties to a (partial) circulant matrix of the same size, our discussions below are based exclusively on the circulant matrix. Using Toeplitz, rather than circulant, matrices will incur some insignificant computation overhead to the methods proposed in this paper.

1.1. **Circulant Compressive Sensing.** In various physical domains, it is easy to realize $C\bar{x}$ since it is equivalent to the discrete convolution $c * \bar{x}$ for a certain vector c; if P is a row-selection (downsampling) operator, $PC\bar{x}$ becomes circulant CS measurements of $\bar{x}$. Using either Toeplitz or circulant matrices, Tropp *et al.* [27] describes a *random filter* for acquiring a signal $\bar{x}$; Haupt *et al.* [12] describes a channel estimation problem to identify a vector $\bar{x}$ (called impulse responses) that characterizes a discrete linear time–invariant (LTI) system; Meng *et al.* [19, 20] applies it to high-resolution OFDM channel estimation. Random convolutions can also be applied in some imaging systems in which convolutions either naturally arise or can be physically realized [3, 13, 17, 16, 24]. Furthermore, random convolutions can be realized by an optical correlator [24]. Since any circulant matrix C in (2) can be diagonalized by a Fourier transform, i.e., obeying

$$(3) \qquad\qquad\qquad C = FDF^*,$$

where F is the discrete Fourier matrix of the same size as C and D is a diagonal matrix, implementing $C\bar{x}$ is equivalent to implementing $FDF^*\bar{x}$, which can be realized through optical lens and a static spatial light modulator (SLM) [29]. Recently, the use of Toeplitz and circulant matrices has been proposed for compressive MR imaging by Liang *et al.* [14].

There are rich theoretical results on circulant matrices for CS. In [2], Toeplitz measurement matrices are constructed with i.i.d. random entries of ± 1 or $\{-1, 0, 1\}$; their downsampling effectively selects the first m rows; and the number of measurements for stable ℓ_1 recovery is shown to be $m \geq O(k^3 \cdot \log n/k)$, where k is the signal sparsity. The work [12] selects the first m rows of a Toeplitz matrix with i.i.d. Bernoulli or Gaussian entries for sparse channel estimation. Their scheme requires $m \geq O(k^2 \cdot \log n)$ for stable ℓ_1 recovery. The work [20] establishes stable recovery under the condition $m \geq O(k^2 \log(n/k))$. In [24], random convolution with either random downsampling or random demodulation is proposed and studied. It is shown that the resulting measurement matrix is incoherent with any given sparse

---

[1] By "a partial Toeplitz matrix", we refer to a submatrix formed by selecting m out of the n rows of the original Toeplitz matrix, where $m < n$. Given a row selection operator P and a Toeplitz matrix T, PT is a partial Toeplitz matrix.

basis with high probability and ℓ_1 recovery is stable given $m \geq O(k \cdot \log n + \log^3 n)$. Recent results in [23] show that several random circulant matrices satisfy the restricted isometry property (RIP) in expectation and with high probability given $m \geq O(\max\{k^{3/2} \log^{3/2} n, k \log^2 k \log^2 n\})$ with arbitrary downsampling, and thus guarantee exact and stable CS recovery. We note that all these results are based on random circulant matrices, so they do not apply to optimized circulant matrices in this paper. We demonstrate that learned circulant matrices achieve even better performance.

The use of circulant sensing matrices also allows faster signal and image recovery. Practical algorithms (e.g., [28, 30, 31, 34, 32]) for CS are based on performing operations including multiplications involving with Φ and Φ^*. For partial circulant matrix $\Phi = PC$, Φx and $\Phi^* y$ each can be quickly computed by two fast Fourier transforms (FFTs) and simple component-wise operations. This is much cheaper than multiplying a general matrix with a vector. For image recovery, a splitting algorithm taking advantages of the circulant structure has been proposed in [33] and shows both satisfactory recovery quality and speed. These fast algorithms apply to the learned circulant matrices in this paper.

1.2. **Learning dictionaries and sensing matrices.** In CS, signal and image reconstructions are based on how they are sparsely represented. The sparse representation involves a choice of dictionary, a set of elementary signals (or atoms) used to sparsely decompose the underlying signal or image. There are *analytic* dictionaries and *learned* dictionaries. Examples of analytic dictionaries include the discrete cosine basis, various wavelets bases, as well as tight frames. Some of them are orthogonal while others are over-complete. Their analytic properties have been studied, and they feature fast implementations; hence, they have found wide applications. Properly *learned* (as opposed to *analytic*) bases can give rise to even sparser representations of signals and, in particular, images, so they can give better encoding and decoding performance than the analytic dictionaries; see [22, 10, 21] for examples and explanations. Although most theoretical results of CS recovery do not apply to learned dictionaries and optimized sensing matrices, one useful tool is the so-called *mutual coherence* between a dictionary Ψ and a sensing matrix Φ: with $D := \Phi \Psi = [d_1, \ldots, d_K]$, it is defined as [15]

$$(4) \qquad \mu(D) := \max_{i \neq j, 1 \leq i,j \leq K} \frac{|d_i^* d_j|}{\|d_i\|_2 \|d_j\|_2}.$$

A smaller $\mu(D)$ tends to allow more Ψ-sparse signals $\bar{x}$ to be successfully recovered from measurements $\Phi \bar{x}$ via various CS algorithms. Hence, Elad [9] seeks means to reduce $\mu(D)$. His work demonstrates improved recovery quality with learned (non-circulant) sensing matrices Φ, and it has motivated the subsequent work [8], which is not based on minimizing $\mu(D)$ but instead pursuing $(\Phi \Psi)^*(\Phi \Psi) \approx I$, where I denotes the identity matrix. Note that $(\Phi \Psi)^*(\Phi \Psi) \approx I$ directly targets the RIP of $\Phi \Psi$, because in that case, any $k \times k$ leading minor of $(\Phi \Psi)^*(\Phi \Psi)$ also approximates the identity matrix. The results of [8] are even better than those in [9] since $\mu(D)$ is a worst-case characteristic whereas $(\Phi \Psi)^*(\Phi \Psi) \approx I$ tends to reduce the average of off-diagonal elements of $(\Phi \Psi)^*(\Phi \Psi)$ and improve the recovery performance in average. Also, the latter is easier to compute.

1.3. **Contributions.** We are motivated by [8] and propose numerical methods to minimize $\|(\Phi \Psi)^*(\Phi \Psi) - I\|_F$, where Φ is either a full or partial circulant matrix.

Specifically, we determine the spectrum vector $d = \text{diag}(D)$ of $\Phi = PC = P(FDF^*)$ by numerically optimizing the magnitudes of the entries in d and then assigning them uniformly random phases. Note that if a circulant matrix C is generated by either assigning a random vector as its first row or following Romberg's strategy [24], then the corresponding vector d has uniformly random phases. The only yet key difference among the different approaches lies on the magnitudes of d. Romberg [24] assigns a unit magnitude uniformly whereas we choose to optimize the magnitudes in order to adapt to the training data.

Like [8], we also learn Φ and Ψ together, performing the so-called *coupled learning* of Φ and Ψ. While results of [8] are limited to one dimensional case[2], we take advantages of the circulant structure and deal with more than one dimension, enabling the learned circulant sensing for signals such as images and videos. Furthermore, we address the patch-scale limitation of [8], namely, the sensing matrix size n must equal the dictionary atom size; namely, if the dictionary for a 512×512 image is formed by 8×8 patches, the sensing matrix generated in [8] has only $64 = 8 \times 8$ columns and applies to vectors of length 64, instead of 512×512. Hence, the learned sensing matrix cannot be applied to the entire image. We remove this limitation and perform image-scale learning by generating circulant sensing operators applicable to the entire signals or images.

Our approaches are tested on synthetic 1D signals, as well as the images in the Berkeley segmentation dataset [18]. The learned circulant sensing matrix gives better recoverability over both random circulant and Gaussian random matrices. For real image tests, the coupled learning approach achieves even better recovery performance.

1.4. **Notation.** We let $(\cdot)^\top$ and $(\cdot)^*$ denote *transpose* and *conjugate transpose*, respectively. conj($\cdot$) stands for the *conjugate* operator. Define $A \bullet B = \sum_{i,j} A_{ij} B_{ij}$ and $\langle A, B \rangle = \sum_{i,j} \text{conj}(A_{ij}) B_{ij}$ for any two matrices A, B of the same dimension. vec(A) is a vector formed by stacking all the columns of A, and diag($\cdot$) is defined in the same way as MATLAB function `diag`, which either extracts the diagonal entries of a given matrix to form a vector or, given a vector, forms a diagonal matrix with the vector's entries. In addition, $\odot$ and $\oslash$ denote *component-wise* multiplication and division, respectively.

The rest of this paper is organized as follows. Section 2 overviews one and two dimensional circulant correlations. A two-step procedure to optimize a partial circulant sensing matrix/operator is described in section 3. Section 4 discusses how to form an image-scale dictionary from a patch-scale one, and section 5 describes how the involved optimization problems are solved. Numerical results are presented in section 6.

2. **Overview of circulant correlations.** Given a *kernel* $v \in \mathbb{C}^n$, the circulant correlation $x \star v$ for a vector $x \in \mathbb{C}^n$ is defined as

$$(5) \qquad (x \star v)_k = \sum_{i=1}^{n} x_i v_{k_i}, \text{ for } k = 1, \ldots, n,$$

---

[2]Strictly speaking, the work [8] also applies to two dimensional images. However, it needs to reshape each two dimensional image patch into a one dimensional vector.

where $k_i = \mod(n+i-k, n)+1$. Let $c = [v_n, v_{n-1}, \ldots, v_1]^\top$. Then $x \star v$ is equivalent to the convolution $x * c$, which is defined as

$$(x * c)_k = \sum_{i=1}^n x_i c_{k_i}, \text{ for } k = 1, \ldots, n,$$

where $k_i = \mod(n + k - i - 1, n) + 1$.

We next introduce three methods to compute $x \star v$. First, introducing the $n \times n$ *cyclic permutation* matrix

$$(6) \qquad P_n = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & 0 \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix},$$

we can write formula (5) as

$$(x \star v)_k = x^\top P_n^{k-1} v, \text{ for } k = 1, \ldots, n.$$

Multiplying P_n from the left circularly down-shifts by a row and multiplying $P_n^\top$ from the right circularly right-shifts a column. For example, given $v = [1, 2, 3]^\top$ and $x = [-1, 0, 1]^\top$, $y = x \star v$ equals

$$y_1 = x^\top P_3^0 v = -1 \cdot \mathbf{1} + 0 \cdot \mathbf{2} + 1 \cdot \mathbf{3} = 2,$$
$$y_2 = x^\top P_3 v = -1 \cdot \mathbf{3} + 0 \cdot \mathbf{1} + 1 \cdot \mathbf{2} = -1,$$
$$y_3 = x^\top P_3^2 v = -1 \cdot \mathbf{2} + 0 \cdot \mathbf{3} + 1 \cdot \mathbf{1} = -1.$$

Secondly, we can compute $x \star v$ via the matrix-vector multiplication Cx with the *circulant* matrix

$$(7) \qquad C = \begin{bmatrix} v_1 & v_2 & \cdots & v_n \\ v_n & v_1 & \cdots & v_{n-1} \\ \ddots & \ddots & \ddots & \\ v_2 & v_3 & \cdots & v_1 \end{bmatrix}.$$

Thirdly, $x \star v$ can be quickly computed by two fast Fourier transforms and some component-wise multiplications as described in the following lemma, which is a restatement of the convolution theorem.

Lemma 2.1. *Any circulant matrix C in (7) can be written as $C = FDF^*$, where F is the $n \times n$ unitary discrete Fourier transform matrix, i.e.,*

$$F_{ij} = \frac{1}{\sqrt{n}} \exp(-\frac{2\pi\sqrt{-1}ij}{n}), \text{ for } i = 1, \ldots, n; \ j = 1, \ldots, n,$$

and $D = \diag(d)$ where $d = \sqrt{n} F v$.

Remark 1. The matrix C is *real-valued* if d is *conjugate symmetric*, namely, $d_i = \conj(d_{i'})$ for every i and $i' = \mod(n - i + 1, n) + 1$. Imposing conjugate symmetry leads to real-valued C and reduces the freedom of d to nearly half.

2D circulant correlation inherits the nice properties of 1D circulant correlation. Given a *kernel* $M \in \mathbb{C}^{n_1 \times n_2}$, the 2D circulant correlation $Y = X \star M$ for a given matrix $X \in \mathbb{C}^{n_1 \times n_2}$ is defined by

$$(8) \qquad (X \star M)_{ts} = \sum_{i,j} X_{ij} M_{t_i s_j}, \text{ for } 1 \le t \le n_1, 1 \le s \le n_2,$$

where $t_i = \mathrm{mod}(n_1 + i - t, n_1) + 1$ and $s_j = \mathrm{mod}(n_2 + j - s, n_2) + 1$.

Similar to 1D circulant, $X \star M$ can be computed in three ways. First, with cyclic permutation matrices P_{n_1} and P_{n_2}, formula (8) can be written as

$$(X \star M)_{ts} = X \bullet (P_{n_1}^{t-1} M (P_{n_2}^\top)^{s-1}), \text{ for } 1 \le t \le n_1, 1 \le s \le n_2.$$

For instance, if

$$X = \begin{bmatrix} 1 & -1 & 0 \\ -2 & 0 & 1 \end{bmatrix} \qquad \text{and} \qquad M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix},$$

then $Y = X \star M$ has components

$$Y_{11} = \begin{bmatrix} 1 & -1 & 0 \\ -2 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} = -3, \qquad Y_{21} = \begin{bmatrix} 1 & -1 & 0 \\ -2 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 4 & 5 & 6 \\ 1 & 2 & 3 \end{bmatrix} = 0,$$

$$Y_{12} = \begin{bmatrix} 1 & -1 & 0 \\ -2 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 3 & 1 & 2 \\ 6 & 4 & 5 \end{bmatrix} = -5, \qquad Y_{22} = \begin{bmatrix} 1 & -1 & 0 \\ -2 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 6 & 4 & 5 \\ 3 & 1 & 2 \end{bmatrix} = -2,$$

$$Y_{13} = \begin{bmatrix} 1 & -1 & 0 \\ -2 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 2 & 3 & 1 \\ 5 & 6 & 4 \end{bmatrix} = -7, \qquad Y_{23} = \begin{bmatrix} 1 & -1 & 0 \\ -2 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} 5 & 6 & 4 \\ 2 & 3 & 1 \end{bmatrix} = -4.$$

Secondly, $Y = X \star M$ can be computed by matrix-vector multiplications via $\mathrm{vec}(Y) = C_2 \mathrm{vec}(X)$, where

$$(9) \qquad \begin{aligned} C_2 = \big[&\mathrm{vec}(M^{1,1}), \mathrm{vec}(M^{2,1}), \ldots, \mathrm{vec}(M^{n_1,1}), \ldots, \mathrm{vec}(M^{1,s}), \\ &\ldots, \mathrm{vec}(M^{n_1,s}), \ldots, \mathrm{vec}(M^{n_1,n_2}) \big]^\top, \end{aligned}$$

with the notation $M^{t,s} := P_{n_1}^{t-1} M (P_{n_2}^\top)^{s-1}$ for $1 \le t \le n_1$ and $1 \le s \le n_2$. It is not difficult to verify that C_2 is a block-circulant matrix. For the above example, we have

$$C_2 = \begin{bmatrix} 1 & 4 & 2 & 5 & 3 & 6 \\ 4 & 1 & 5 & 2 & 6 & 3 \\ 3 & 6 & 1 & 4 & 2 & 5 \\ 6 & 3 & 4 & 1 & 5 & 2 \\ 2 & 5 & 3 & 6 & 1 & 4 \\ 5 & 2 & 6 & 3 & 4 & 1 \end{bmatrix} \quad \text{and} \quad C_2 \mathrm{vec}(X) = \begin{bmatrix} 1 & 4 & 2 & 5 & 3 & 6 \\ 4 & 1 & 5 & 2 & 6 & 3 \\ 3 & 6 & 1 & 4 & 2 & 5 \\ 6 & 3 & 4 & 1 & 5 & 2 \\ 2 & 5 & 3 & 6 & 1 & 4 \\ 5 & 2 & 6 & 3 & 4 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ -2 \\ -1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -3 \\ 0 \\ -5 \\ -2 \\ -7 \\ -4 \end{bmatrix}.$$

Thirdly, we can quickly compute $X \star M$ through two fast 2D Fourier transforms and some component-wise multiplications according to the following lemma, which is a restatement of the 2D convolution theorem.

Lemma 2.2. *Given* kernel $M \in \mathbb{C}^{n_1 \times n_2}$, *define 2D circulant operator* $\mathcal{C}_2$ *on* $\mathbb{C}^{n_1 \times n_2}$ *by* $\mathcal{C}_2(X) = X \star M$ *for* $X \in \mathbb{C}^{n_1 \times n_2}$. *Then* $\mathcal{C}_2$ *can be represented as* $\mathcal{C}_2 = \mathcal{F}_2 \mathcal{V} \mathcal{F}_2^*$, *where* $\mathcal{F}_2$ *is the orthogonal 2D discrete Fourier transformation operator on* $\mathbb{C}^{n_1 \times n_2}$, $\mathcal{F}_2^*$ *is the adjoint operator of* $\mathcal{F}_2$ *as well as its inverse operator, and* $\mathcal{V}$ *is an operator on* $\mathbb{C}^{n_1 \times n_2}$ *defined by* $\mathcal{V}(X) = V \odot X$ *for any* $X \in \mathbb{C}^{n_1 \times n_2}$ *with* $V = \sqrt{n_1 n_2} \mathcal{F}_2(M)$.

Remark 2. The block-circulant matrix C_2 corresponding to $\mathcal{C}_2$ is real valued if V is *conjugate symmetric*, namely, $V_{ij} = \overline{V_{i'j'}}$ for every pair of i, j and $i' = \mathrm{mod}(n_1 - i + 1, n_1) + 1, j' = \mathrm{mod}(n_2 - j + 1, n_2) + 1$.

3. **Learning circulant sensing matrix/operator.** In this section, we illustrate how to learn the 1D kernel v in formula (5) and the 2D kernel M in formula (8), as well as their corresponding downsampling operators P (row selection) and $\mathcal{P}_\Omega$ (sample selection); hence, we form a partial circulant sensing matrix $PC \in \mathbb{C}^{m \times n}$ and a partial 2D circulant sensing operator $\mathcal{P}_\Omega \mathcal{C}_2$, respectively. Here, P selects m out of the n rows from C, and $\mathcal{P}_\Omega$ collects the entries in $\Omega \subset \{1, \ldots, n_1\} \times \{1, \ldots, n_2\}$ and forms a column vector. For example,

$$X = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \qquad \text{and} \qquad \Omega = \{(1,2), (2,1), (3,1), (2,3)\},$$

lead to $\mathcal{P}_\Omega(X) = [4, 7, 2, 6]^\top$.

Remark 3. In our experiments, we will normalize each column of the sensing matrix Φ, for otherwise the recovery can be numerically instable in particular when Φ is a Gaussian random matrix. Note that the normalization to PC (or $\mathcal{P}_\Omega \mathcal{C}_2$) is equivalent to multiplying a diagonal matrix (or having a componentwise operator) to the right, and thus it only slightly increases the computation.

3.1. **Learning 1D circulant kernel and downsampler.** Let $\Psi \in \mathbb{C}^{n \times K}$ be a given dictionary such that the underlying signal $\bar{x} = \Psi \bar{\theta}$, where $\bar{\theta}$ is a (nearly) sparse vector. Following [8], we would like to design a partial circulant sensing matrix $\Phi = PC$ such that $\Psi^* \Phi^* \Phi \Psi \approx I$, where C is an $n \times n$ circulant matrix and P is an $m \times n$ downsampling matrix. To construct a partial circulant matrix Φ, we shall determine P and C. We first learn the circulant matrix C and then the downsampler P.

According to Lemma 2.1, $C = FDF^*$ where the diagonal matrix $D = \text{diag}(d)$ with entries $d = \sqrt{n} F v$ and v is the *kernel* of C. Therefore, learning C is equivalent to choosing the best kernel v or vector d. Our approach is based on solving

$$(10) \qquad \min_C \| \Psi^* C^* C \Psi - I \|_F,$$

so that $\Psi^* C^* C \Psi \approx I$. From $F^* F = I$, we have

$$\Psi^* C^* C \Psi = \Psi^* F D^* F^* F D F^* \Psi = \Psi^* F \text{diag}(d^*) \text{diag}(d) F^* \Psi.$$

Introduce $B := F^* \Psi \Psi^* F$, which satisfies $B^* = B$. Let $\bar{B} := [|B_{ij}|^2]$. Given Ψ, matrices B and $\bar{B}$ are constant matrices. Let

$$(11) \qquad z := (|d_1|^2, |d_2|^2, \ldots, |d_n|^2)^\top \geq 0,$$

which are unknowns to be determined. We have $\text{diag}(d^*)\text{diag}(d) = \text{diag}(z)$ and

$$\frac{1}{2} \| \Psi^* C^* C \Psi - I \|_F^2$$
$$= \frac{1}{2} \| \Psi^* F \text{diag}(d^*) \text{diag}(d) F^* \Psi - I \|_F^2$$
$$= \frac{1}{2} \text{tr}(\Psi^* F \text{diag}(z) B \text{diag}(z) F^* \Psi) - \text{tr}(\Psi^* F \text{diag}(z) F^* \Psi) + \frac{n}{2}$$
$$= \frac{1}{2} \text{tr}(\text{diag}(z) B \text{diag}(z) B) - \text{tr}(\text{diag}(z) B) + \frac{n}{2}$$
$$= \frac{1}{2} z^\top \text{diag}(B \text{diag}(z) B) - z^\top \text{diag}(B) + \frac{n}{2}$$
$$= \frac{1}{2} z^\top \bar{B} z - z^\top \text{diag}(B) + \frac{n}{2}.$$

Hence, we reduce (10) to

$$(12) \qquad \min_{z \geq 0} \frac{1}{2} z^\top \bar{B} z - z^\top \text{diag}(B).$$

Problem (12) is a convex quadratic program and can be reformulated as a non-negative least-squares problem.

Given a solution $z = z_{\text{opt}}$ to (12), any d obeying (11) gives $C = F\text{diag}(d)F^*$ as a solution to (10). Since only $|d_i|$, $i = 1, \ldots, n$, are specified, the phases of d_i are still subject to determine. Generally, phases encode the location of the information in a signal. Since such location is unknown at the time of sensing, we choose to generate the phase of every d_i uniformly at random, which is suggested by [24]. We want to mention that if Ψ is an orthogonal basis or tight frame, i.e., $\Psi\Psi^* = I$, the solution of (12) is simply a vector with all ones, and in this case our generated d is the same as that in [24]. However, as $\Psi\Psi^* \neq I$, e.g., all the learned dictionaries in our experiments, the solution of (12) is generally not the all-one vector, and the optimized d is different from that in [24]. Models (10) and (12) lead to significant improvement in sensing efficiency as we shall demonstrate by numerical examples in section 6.

Remark 4. The above procedure generates a complex-valued C. To obtain a real-valued C, one shall add constraints $z_i = z_{n-i+2}$ for $i = 2, \ldots, n$ in the problem (12) and then generate a *conjugate symmetric* d from the solution z_{opt}.

As Ψ is square, i.e., $n = K$, we also tested an alternative model as follows. Although it does not work as well as (10), we believe it is worth conveying the information to the readers. The model is

$$(13) \qquad \min_{C} \ \frac{1}{2}\|C\Psi - I\|_F^2,$$

which is equivalent to

$$(14) \qquad \min_{d \in \mathbb{C}^n} \ \frac{1}{2}d^*Ad - \frac{1}{2}d^\top \text{diag}(F^*\Psi F) - \frac{1}{2}\text{conj}(d^\top \text{diag}(F^*\Psi F)),$$

where A is a diagonal matrix with diagonal entries given by vector $\text{diag}(F^*\Psi\Psi^*F)$. Since (14) is component-wise separable in d_i, it is easy to derive its closed-form solution

$$(15) \qquad d_{\text{opt}} = \text{conj}(\text{diag}(F^*\Psi F)) \oslash \text{diag}(F^*\Psi\Psi^*F).$$

Our numerical experience suggests that this solution (and thus model (13)) is not as effective as that of (10). On the other hand, Ψ is usually overcomplete, for which case (13) is inapplicable.

After the circulant matrix C is determined, we now optimize P, and thus fully determine Φ. A simple yet effective solution is the random selection, that is, let P select m out of the n rows of C uniformly at random. This tends to work well on signals without dominating frequencies.

We can also choose to minimize $\|\Psi^*\Phi^*\Phi\Psi - I\|_F$ so that $\Psi^*\Phi^*\Phi\Psi \approx I$. Let p_i be the binary variable indicating the selection of element i, i.e.,

$$(16) \qquad p_i = \begin{cases} 1, & \text{matrix } P \text{ selects row } i, \\ 0, & \text{otherwise.} \end{cases}$$

Since $\Phi = PC$, we shall solve

$$(17) \qquad \min_P \frac{1}{2}\|\Psi^*C^*(P^*P)C\Psi - I\|_F^2 = \left\|\sum_{i=1}^n p_i(q_i q_i^*) - I\right\|_F^2,$$

where q_i is the i-th column of Ψ^*C^*. Model (17) is equivalent to the binary quadratic program

$$(18) \qquad \begin{cases} \min_p \|[\mathrm{vec}(q_1 q_1^*), \cdots, \mathrm{vec}(q_n q_n^*)]p - \mathrm{vec}(I)\|_2^2, \\ \text{subject to } \sum_i p_i = m \text{ and } p_i \in \{0, 1\}, \ \forall i. \end{cases}$$

Here, $[\mathrm{vec}(q_1 q_1^*), \cdots, \mathrm{vec}(q_n q_n^*)]$ is a matrix of n^2 rows and n columns. By simple calculation, we can write (18) into the following equivalent problem

$$(19) \qquad \min_p p^\top H p - 2f^\top p, \text{ subject to } \sum_i p_i = m \text{ and } p_i \in \{0, 1\}, \ \forall i,$$

where $H = [|G_{ij}|^2]$, $f = \mathrm{diag}(G)$ and $G = C\Psi\Psi^*C^*$. Problem (19) is NP-hard in general and can be solved to a moderate size by solvers such as Gurobi [4].

3.2. **Learning 2D circulant kernel and downsampler.** A 2D circulant operator $\mathcal{C}_2$ is often used in imaging [25, 29]. It is not equivalent to a circulant matrix even if the image is vectorized but can be reduced to a block circulant matrix with circulant blocks [6].

Given a dictionary $\Psi = [\psi_1, \psi_2, \ldots, \psi_K]$ where each $\psi_i \in \mathbb{C}^{n_1 \times n_2}$, we define a linear operator $\mathcal{Q}$ on $\mathbb{C}^K$ by $\mathcal{Q}(\theta) = \sum_{i=1}^K \theta_i \psi_i$ for $\theta \in \mathbb{C}^K$. An array $X \in \mathbb{C}^{n_1 \times n_2}$ is sparse with respect to the dictionary Ψ if it can be represented as $X = \mathcal{Q}(\theta)$ with a sparse $\theta \in \mathbb{C}^K$. Let $\mathcal{P}_\Omega$ be the downsampling operator on $\mathbb{C}^{n_1 \times n_2}$ defined at the beginning of this section. Then the basis pursuit problem to recover X can be written as

$$(20) \qquad \min_\theta \|\theta\|_1, \text{ subject to } \mathcal{P}_\Omega \mathcal{C}_2 \mathcal{Q}(\theta) = b,$$

where $b = \mathcal{P}_\Omega \mathcal{C}_2(X)$ contains the measurements of X. To improve the recoverability of the ℓ_1 optimization problem (20), we shall try to make $(\mathcal{C}_2\mathcal{Q})^*\mathcal{C}_2\mathcal{Q}$ close to the identity operator $\mathcal{I}$ on $\mathbb{C}^K$. The adjoint operator of $\mathcal{Q}$ is defined as

$$\mathcal{Q}^*(X) = [\langle \psi_1, X \rangle, \langle \psi_2, X \rangle, \ldots, \langle \psi_K, X \rangle]^\top, \text{ for any } X \in \mathbb{C}^{n_1 \times n_2}.$$

Hence, for any $\theta \in \mathbb{C}^K$,

$$(\mathcal{C}_2\mathcal{Q})^*\mathcal{C}_2\mathcal{Q}(\theta) = \left[\langle \mathcal{C}_2(\psi_1), \sum_{i=1}^K \theta_i \mathcal{C}_2(\psi_i)\rangle, \ldots, \langle \mathcal{C}_2(\psi_K), \sum_{i=1}^K \theta_i \mathcal{C}_2(\psi_i)\rangle\right]^\top,$$

and making $(\mathcal{C}_2\mathcal{Q})^*\mathcal{C}_2\mathcal{Q} \approx \mathcal{I}$ is equivalent to making $\langle \mathcal{C}_2(\psi_j), \sum_{i=1}^K \theta_i \mathcal{C}_2(\psi_i)\rangle \approx \theta_j$ for $1 \le j \le K$. Toward this goal, we solve

$$(21) \qquad \min_G \|G - I\|_F^2, \text{ subject to } G_{ts} = \langle \mathcal{C}_2(\psi_t), \mathcal{C}_2(\psi_s)\rangle,$$

which can be simplified as follows. According to Lemma 2.2, we have

$$\langle \mathcal{C}_2(\psi_t), \mathcal{C}_2(\psi_s)\rangle = \langle \mathcal{F}_2 \mathcal{V} \mathcal{F}_2^*(\psi_t), \mathcal{F}_2 \mathcal{V} \mathcal{F}_2^*(\psi_s)\rangle$$
$$= \langle \mathcal{V} \mathcal{F}_2^*(\psi_t), \mathcal{V} \mathcal{F}_2^*(\psi_s)\rangle = \langle \mathcal{V}^* \mathcal{V} \mathcal{F}_2^*(\psi_t), \mathcal{F}_2^*(\psi_s)\rangle.$$

Let $U = \text{conj}(V) \odot V$ and define $\mathcal{U} := \mathcal{V}^*\mathcal{V}$, where $\mathcal{V}^*$ is the adjoint operator of $\mathcal{V}$. We have $\mathcal{U}(X) = U \odot X$ for any $X \in \mathbb{C}^{n_1 \times n_2}$. Let $u = \text{vec}(U)$ and $Y = [y^1, y^2, \ldots, y^K]$ with $y^s = \text{vec}(\mathcal{F}_2^*(\psi_s))$ for $s = 1, 2, \ldots, K$. Then

$$G_{ts} = \langle \mathcal{V}^*\mathcal{V}\mathcal{F}_2^*(\psi_t), \mathcal{F}_2^*(\psi_s)\rangle = (u \odot y^t)^* y^s = u^*(\text{conj}(y^t) \odot y^s).$$

Hence,

$$\begin{aligned}
\|G\|_F^2 &= \sum_{t,s} \text{conj}\,(G_{ts}) \cdot G_{ts} \\
&= \sum_{t,s} \text{conj}\,\big(u^*(\text{conj}(y^t) \odot y^s)\big) \cdot \big(u^*(\text{conj}(y^t) \odot y^s)\big) \\
&= \sum_{t,s} u^*(\text{conj}(y^t) \odot y^s)\big((\text{conj}(y^t) \odot y^s)\big)^* u \\
&= \sum_{t,s} u^* \big(\text{conj}(y^t(y^t)^*) \odot (y^s(y^s)^*)\big) u = u^* A u,
\end{aligned}$$

where $A := \sum_{t,s} \text{conj}(y^t(y^t)^*) \odot (y^s(y^s)^*) = \text{conj}(YY^*) \odot (YY^*)$. In addition,

$$\text{tr}(G) = \text{tr}(G^*) = \sum_t u^*(\text{conj}(y^t) \odot y^t) = (\text{diag}(YY^*))^* u = f^* u,$$

with $f := \text{diag}(YY^*)$. Therefore, problem (21) is equivalent to the convex quadratic program

$$(22) \qquad \min_{u \geq 0} \frac{1}{2} u^\top A u - f^\top u$$

in the same form of (12), where we have used real transpose instead of conjugate transpose since f, u and A are all real.

Given a solution $u = u_{\text{opt}}$ of (22), we can obtain matrix U via $u = \text{vec}(U)$. Any V satisfying $U = \text{conj}(V) \odot V$ defines $\mathcal{C}_2 = \mathcal{F}_2 \mathcal{V} \mathcal{F}_2^*$ as a solution to (21). Like done in Section 3.1 for 1D circulant, we choose to generate the phase of each entry V_{ts} uniformly at random.

Remark 5. The above procedure gives a complex-valued $\mathcal{C}_2$. To obtain a real-valued $\mathcal{C}_2$, we can add constraints $u_{(n_1-1)j+i} = u_{(n_1-1)j'+i'}$ for every pair of i, j and $i' = \text{mod}(n_1 - i + 1, n_1) + 1, j' = \text{mod}(n_2 - j + 1, n_2) + 1$ to (22), and then generate a conjugate symmetric V from the solution u_{opt}.

Given $\mathcal{C}_2$, we can choose Ω uniformly at random or optimize $\mathcal{P}_\Omega$ such that $(\mathcal{P}_\Omega \mathcal{C}_2 \mathcal{Q})^*(\mathcal{P}_\Omega \mathcal{C}_2 \mathcal{Q}) \approx \mathcal{I}$, which can be achieved by solving

$$(23) \qquad \min_G \|G - I\|_F^2, \text{ subject to } G_{ts} = \langle \mathcal{P}_\Omega \mathcal{C}_2(\psi_t), \mathcal{P}_\Omega \mathcal{C}_2(\psi_s)\rangle.$$

Let $n = n_1 n_2$ and $D_\Omega \in \mathbb{C}^{n \times n}$ be a diagonal matrix with diagonal entries defined by

$$(24) \qquad (D_\Omega)_{k_{ij}k_{ij}} = \begin{cases} 1, & \text{if } (i,j) \in \Omega, \\ 0, & \text{otherwise,} \end{cases} \qquad \text{for } 1 \leq i \leq n_1, 1 \leq j \leq n_2,$$

where $k_{ij} = i + (j-1)n_1$. Then $\langle D_\Omega \text{vec}(X), D_\Omega \text{vec}(Y)\rangle = \langle \mathcal{P}_\Omega X, \mathcal{P}_\Omega Y\rangle$ for any $X, Y \in \mathbb{C}^{n_1 \times n_2}$. Letting $y^t = \text{vec}(\mathcal{C}_2(\psi_t))$ for $t = 1, \ldots, K$, we have

$$G_{ts} = \langle \mathcal{P}_\Omega \mathcal{C}_2(\psi_t), \mathcal{P}_\Omega \mathcal{C}_2(\psi_s)\rangle = \langle D_\Omega y^t, D_\Omega y^s\rangle = (y^t)^* D_\Omega y^s.$$

Let $Y = \sum_s y^s (y^s)^*$. Then

$$\|G\|_F^2 = \sum_{t,s} \mathrm{conj}(G_{ts}) \cdot G_{ts} = \sum_{t,s} \left((y^t)^* D_\Omega y^s \right) \cdot \left((y^s)^* D_\Omega y^t \right)$$

$$= \sum_t (y^t)^* (D_\Omega Y D_\Omega) y^t = \sum_t \mathrm{tr} \left(D_\Omega Y D_\Omega y^t (y^t)^* \right) = \mathrm{tr}(D_\Omega Y D_\Omega Y)$$

$$= \sum_i (D_\Omega Y D_\Omega Y)_{ii} = \sum_{i,j} (D_\Omega Y)_{ij} \cdot (D_\Omega Y)_{ji} = \sum_{i,j} (D_\Omega)_{ii} \cdot Y_{ij} \cdot (D_\Omega)_{jj} \cdot Y_{ji}$$

$$= \sum_{i,j} (D_\Omega)_{ii} \cdot Y_{ij} \cdot \mathrm{conj}(Y_{ij}) \cdot (D_\Omega)_{jj} = p^\top H p,$$

where $H = [|Y_{ij}|^2]$ and $p = \mathrm{diag}(D_\Omega)$. In addition,

$$\mathrm{tr}(G) = \sum_t G_{tt} = \sum_t (y^t)^* D_\Omega y^t = \sum_t \mathrm{tr}(D_\Omega y^t (y^t)^*) = \mathrm{tr}(D_\Omega Y) = f^\top p,$$

where $f = \mathrm{diag}(Y)$. Hence, (23) is equivalent to

$$(25) \qquad \min_p \frac{1}{2} p^\top H p - f^\top p, \text{ subject to } \sum_k p_k = |\Omega| \text{ and } p_k \in \{0, 1\}, \ \forall k.$$

4. Image-scale dictionary.
For natural images, current dictionary learning methods, e.g., KSVD [1], train a dictionary Ψ^0 from a set of image patches. Hence, the dictionary atoms have the same size as patches. In practice, we often take measurements directly from an entire image $\hat{X}$ instead of its small patches, so the learned dictionary Ψ^0 does not match the image $\hat{X}$. How can we use Ψ^0 to recover $\hat{X}$ from the directly taken measurements? One approach is to make an image-scale dictionary under which $\hat{X}$ is sparse.

Given a patch-scale atom, we construct a list of image-scale atoms that have no common non-zero parts by placing the patch-scale atom at each possible place in the image. Special cares are given near the boundary where only part of the patch-scale atom is used. The precise treatment is specified below.

Let $\hat{X} \in \mathbb{C}^{N_1 \times N_2}$ be an image, and assume that each of its small patches $x \in \mathbb{C}^{n_1 \times n_2}$, where $n_1 < N_1$ and $n_2 < N_2$, has a sparse representation under dictionary $\Psi^0 = [\psi_1, \ldots, \psi_K]$. We want to construct an image-scale dictionary $\boldsymbol{\Psi}$ from Ψ^0 such that $\hat{X}$ is sparse under $\boldsymbol{\Psi}$. Let $n_r = \lfloor \frac{N_1}{n_1} \rfloor, n_c = \lfloor \frac{N_2}{n_2} \rfloor, s_r = N_1 - n_r n_1, s_c = N_2 - n_c n_2$, and $N = n_r n_c + \mathrm{sign}(s_r) n_c + \mathrm{sign}(s_c) n_r + \mathrm{sign}(s_r s_c)$. For each atom ψ_k, we form N image-scale atoms $\Psi_1^k, \Psi_2^k, \ldots, \Psi_N^k \in \mathbb{C}^{N_1 \times N_2}$ as follows.

- For $j = 1, \ldots, n_c$ and $i = 1, \ldots, n_r$, let $t = (j-1)n_r + i$ and Ψ_t^k be the matrix with all elements being *zero* except having ψ_k on the submatrix indexed by the $(n_r(i-1)+1)$th through $(n_r i)$th row and the $(n_c(j-1)+1)$th through $(n_c j)$th column.
- If $s_r \neq 0$, let $t = n_r n_c + j$ and Ψ_t^k be the matrix with all elements being *zero* except having the last s_r rows of ψ_k on the submatrix indexed by the $(n_r n_1 + 1)$th through N_1th row and the $(n_c(j-1)+1)$th through $(n_c j)$th column for $j = 1, \ldots, n_c$.
- If $s_c \neq 0$, n_r more atoms are formed in a similar way using the last s_c columns of ψ_k.
- If $s_c s_r \neq 0$, we form the last atom Ψ_N^k with all elements being *zero* except having the right-bottom $s_r \times s_c$ submatrix of ψ_k on its right-bottom $s_r \times s_c$ submatrix.
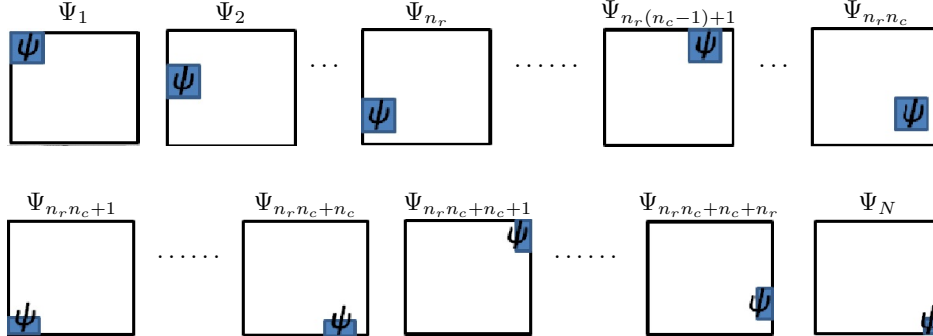
FIGURE 1. From a small patch ψ, we form N image-scale atoms $\Psi_1, \Psi_2, \ldots, \Psi_N$ that have no common non-zero parts by placing ψ at each possible place in the image-size frame. Special cares are given near the boundary where only part of the patch-scale atom is used.

Note that any two atoms generated from ψ_k have no common non-zero parts. The image $\hat{X}$ has a sparse representation under $\mathbf{\Psi}$ since each patch of $\hat{X}$ is sparse under Ψ^0. Figure 1 illustrates how we form N image-scale atoms $\Psi_1, \Psi_2, \ldots, \Psi_N$ from a small atom ψ. In section 6, we will show that this generated image-scale dictionary works well in recovering an entire image from its linear measurements.

5. **Algorithm and implementation.** With a given dictionary Ψ, Algorithm 1 outlines our approach for optimizing a partial circulant matrix Φ.

Algorithm 1

1: Solve (12) for z, generate randomly-phased d from z via (11), and then form $C = F\text{diag}(d)F^*$.
2: Solve (19) for p or randomly generate p and then form P from p via (16).
3: Generate $\Phi = PC$.

For 2D, an optimized partial circulant operator $\mathcal{P}_\Omega \mathcal{C}_2$ can be obtained in a similar way. At Step 1 of Algorithm 1, we use the MATLAB function `quadprog` to solve (12) with its default settings. At Step 2, we use the commercial code Gurobi [4] with MATLAB interface [35] to solve the binary program (19). In our test, the maximum number of iterations was set to 2000. Usually, Gurobi terminated at the maximum number of iterations with the best solution obtained. Hence, (19) was only approximately solved.

We used both synthetic and real data for test. For synthetic data, Ψ was Gaussian randomly generated basis or discrete Fourier basis, and Φ was learned by Algorithm 1. For real data, Ψ was learned in either an uncoupled way or a coupled way. In the uncoupled test, we first learned Ψ from a set of training data $X = [X_1, \ldots, X_L] \in \mathbb{C}^{n \times L}$ by applying KSVD [1] to

$$(26) \qquad \min_{\Psi, \Theta} \|X - \Psi\Theta\|_F^2, \text{ subject to } \|\Theta_i\|_0 \leq S, \forall i$$

for a given sparsity level S, where $\Theta = [\Theta_1, \ldots, \Theta_L]$ and $\|\Theta_i\|_0$ denotes the number of non-zeros of Θ_i. Then we learned Φ by Algorithm 1. In the coupled test, we simultaneously learned a pair of Ψ and Φ in a way similar to [8]. Specifically, during each loop of the coupled algorithm, we first compute Φ via Algorithm 1 with a fixed Ψ, then update Θ by applying OMP [26] to

$$(27) \qquad \min_{\Theta} \alpha \|X - \Psi\Theta\|_F^2 + \|\Phi X - \Phi\Psi\Theta\|_F^2, \text{ subject to } \|\Theta_i\|_0 \leq S,$$

where $\alpha > 0$ is a weight parameter, and finally update Ψ and Θ via solving (27) with respect to Ψ and the current nonzero entries of Θ jointly. Algorithm 2 summarizes our coupledly learning method, and in our experiments we terminated the algorithm if the relative changes of $\|X - \Psi\Theta\|_F$ is less than 10^{-6}.

Algorithm 2

1: **Initialization:** randomly generate Ψ or set Ψ as some learned dictionary.
2: **while** Not converged **do**
3:     Compute Φ via Algorithm 1 with Ψ fixed;
4:     Update Θ by solving (27);
5:     Update Ψ and Θ jointly by solving (27) with Φ and the nonzero locations of Θ fixed.
6: **end while**

After obtaining Ψ and Φ, we used YALL1 [31] (version 1.4) to recover the sparse signal $\bar{\theta}$ via solving

$$(28) \qquad \min_{\theta} \|\theta\|_1 + \frac{1}{\rho}\|\Phi\Psi\theta - b\|_2^2,$$

where $b = \Phi(\Psi\bar{\theta} + \eta)$ was the measurement contaminated by Gaussian noise $\eta \sim \mathcal{N}(0, \sigma^2 I)$ and $\rho \geq 0$ was a parameter corresponding to σ. Throughout our tests, σ was known, and the parameter ρ was set to σ. If $\rho = 0$, problem (28) reduces to

$$(29) \qquad \min_{\theta} \|\theta\|_1, \text{ subject to } \Phi\Psi\theta = b.$$

The stopping tolerance for YALL1 was set to 10^{-5} for noiseless case and 10^{-3} for noise case, unless specified otherwise. The maximum number of iterations was set to 10^4, which is sufficiently large to make YALL1 converge within the given tolerance.

6. **Numerical simulations.** We compared the performance of optimized circulant sensing matrices/operators to that of random ones on both synthetic and real-world data. In addition, we tested a pair of circulant sensing matrix Φ and dictionary Ψ learned together on real-world data. The tested sensing matrices and 2D operators are listed in Table 1. To be fair, we either take all the compared sensing matrices and 2D operators to be real-valued or take all of them to be complex-valued in the same test. In addition, except for the coupled learned circulant matrix *coupled-plus-rand-circ*, all sensing matrices or 2D operators were generated and tested with the same set of synthetic or learned dictionaries.

As in [8], throughout our tests, all columns of Ψ and Φ were normalized to have the unit 2-norm. The normalization of Ψ is only for convenience while that of Φ is critical, in particular when Φ is a Gaussian randomly matrix, for otherwise the recovery by YALL1 or other solvers may become instable. Similarly, we normalized all partial 2D circulant operators. We next introduce an efficient way to normalize 1D matrix PC and 2D operator $\mathcal{P}_\Omega\mathcal{C}_2$.

TABLE 1. List of tested sensing matrices and 2D operators[*]

name	type	real/complex	kernel	downsampler	dimension
rand-circ	circulant	complex	random	random	1D
real-rand-circ	circulant	real	random	random	1D
Gaussian	Gaussian	complex	——	random	1D
real-Gaussian	Gaussian	real	——	random	1D
opt-circ	circulant	complex	optimized	random	1D
real-opt-circ	circulant	real	optimized	random	1D
opt-circ-and-P	circulant	complex	optimized	optimized	1D
real-opt-circ-and-P	circulant	real	optimized	optimized	1D
opt-plus-rand-circ	circulant	complex	optimized+random[†]	random	1D
coupled-plus-rand-circ	circulant	complex	optimized+random	random	1D
rand-2D-circ	circulant	complex	random	random	2D
opt-plus-rand-2D-circ	circulant	complex	optimized+random	random	2D

[*]The kernel v for real random 1D circulant was generated by MATLAB command `randn(n,1)` and that for complex random 1D circulant by `randn(n,1)+1i*randn(n,1)`; real Gaussian was generated by `randn(m,n)` and complex Gaussian by `randn(m,n)+1i*randn(m,n)`; the kernel M for real random 2D circulant was generated by `randn(n1,n2)` and that for complex random 2D circulant by `randn(n1,n2)+1i*randn(n1,n2)`;

[†]60% of the normalized optimized circulant plus 40% of a normalized real random circulant.

6.1. **Normalization of 1D matrix** PC **and 2D operator** $\mathcal{P}_\Omega \mathcal{C}_2$. Let C be defined in (7) and $c = [v_1, v_n, v_{n-1}, \ldots, v_2]^\top$ be its first column. It is easy to verify that the jth column of C is $C_j = [c_{n-j+2}, c_{n-j+3}, \ldots, c_n, c_1, \ldots, c_{n-j+1}]^\top$. Let $\ell = [\ell_1, \ldots, \ell_n]^\top$ with ℓ_j being the Euclidean norm of the jth column of PC. Recalling (16), we have for $j = 1, \ldots, n$ that

$$\ell_j^2 = \|PC_j\|_2^2 = p_1 c_{n-j+2}^2 + p_2 c_{n-j+3}^2 + \ldots p_{j-1} c_n^2 + p_j c_1^2 + \ldots p_n c_{n-j+1}^2 = \sum_{i=1}^{n} p_i c_{s_{ij}}^2,$$

where $s_{ij} = \mathrm{mod}(n+i-j, n) + 1$. Hence, $\ell_j = \sqrt{\sum_{i=1}^{n} p_i c_{s_{ij}}^2}$, for $j = 1, \ldots, n$, and $PC(\mathrm{diag}(\ell))^{-1}$ has columns with the unit 2-norm.

For $\mathcal{P}_\Omega \mathcal{C}_2$, recall (9) and (24). Then normalizing $\mathcal{P}_\Omega \mathcal{C}_2$ is equivalent to normalizing $D_\Omega \mathcal{C}_2$. Let $n = n_1 n_2$ and $\ell = [\ell_1, \ldots, \ell_n]^\top$ with ℓ_j being the Euclidean norm of the jth column of $D_\Omega \mathcal{C}_2$. Note that for $1 \le t, \tau \le n_1$ and $1 \le s, \kappa \le n_2$,

$$(C_2)_{i_{ts} j_{\tau\kappa}} = M_{\tau\kappa}^{t,s} = M_{\alpha_{t\tau}\beta_{s\kappa}}$$

where $i_{ts} = t + (s-1)n_1$, $j_{\tau\kappa} = \tau + (\kappa-1)n_1$ and $\alpha_{t\tau} = \mathrm{mod}(n_1 + \tau - t, n_1) + 1$, $\beta_{s\kappa} = \mathrm{mod}(n_2 + \kappa - s, n_2) + 1$. Hence, for each pair of (τ, κ), we can compute ℓ_j with $j = \tau + (\kappa-1)n_1$ via

$$\ell_j = \sqrt{\sum_{t,s} (D_\Omega)_{i_{ts} i_{ts}} M_{\alpha_{t\tau}\beta_{s\kappa}}^2}.$$

Therefore, we normalize $D_\Omega C_2$ to $D_\Omega C_2(\mathrm{diag}(\ell))^{-1}$ and $\mathcal{P}_\Omega \mathcal{C}_2$ to $\mathcal{P}_\Omega \mathcal{C}_2 \mathcal{L}$, where $\mathcal{L}(X) = X \oslash L$ for any $X \in \mathbb{C}^{n_1 \times n_2}$ with $L \in \mathbb{C}^{n_1 \times n_2}$ obtained from ℓ via $\ell = \mathrm{vec}(L)$.

6.2. **Synthetic data.** We tested different optimized circulant sensing matrices Φ on synthetic data along with two kinds of bases Ψ: the Gaussian random basis and the discrete Fourier basis. The performance was compared to that of unoptimized random circulant, as well as random Gaussian, sensing matrices. The Gaussian random basis was generated by MATLAB command `randn(n)` and then turned to

an orthogonal matrix by QR decomposition, and the Fourier basis was generated by MATLAB command `dftmtx(n)/sqrt(n)`. Both bases were 512×512, and the sensing matrices had the size 64×512, i.e., an 8x downsample. In this test, $\bar{\theta}$ was generated with k randomly located nonzero entries sampled from the standard Gaussian distribution and then normalized to have the unit 2-norm. The stopping tolerance for YALL1 was set to 5×10^{-8}. Figure 2 depicts the comparison results of sensing matrices: *rand-circ*, *Gaussian*, *opt-circ*, *opt-circ-and-P*, as well as their real-valued counterparts. We call a recovery θ^* *successful* if the relative error $\|\theta^* - \bar{\theta}\|_2 / \|\bar{\theta}\|_2$ was below 10^{-4}. We calculated the success rate out of 50 independent trials at every sparsity level k.

All the four subfigures of Figure 2 reveal that optimized circulant sensing matrices led to equally good performance as random Gaussian matrices. For the random basis, random circulant matrices achieved similar recovery success rate, while they performed extremely bad for the discrete Fourier basis. The reason for the bad performace of random circulant matrices can be found in [33]. On the other hand, optimizing the selection matrix P hardly made further improvement. We believe that unless the underlying signal has dominant frequencies, optimizing the selection matrix P will not lead to consistent improvement. Therefore, in the subsequent tests, we let P be random selection matrices. In addition, the improvement by optimizing circulant sensing matrices over randomly generated ones is similar for real and complex-valued. For convenience, we use complex-valued sensing matrices/operators in the rest tests.

Although the tests above are based on synthetic signals that are exactly sparse, similar performance of different sensing matrices was observed on those that are nearly sparse. Since real images are nearly sparse (under certain dictionaries), we skip presenting the results of nearly sparse synthetic signals and proceed to the real image tests.

6.3. **Image tests with circulant matrices.** In this subsection, we present the performance of sensing matrices *rand-circ*, *Gaussian*, *opt-plus-rand-circ* and *coupled-plus-rand-circ* on real images.[3] The dictionaries for all of them were learned from the same training set and had size 64×256, namely, we set $K = 256$. The first three sensing matrices shared the same set of dictionaries learned by KSVD [1] with sparsity level $S = 6, 8$ in (26), and the last one was learned simultaneously with its corresponding dictionary by the coupled method described in Section 5 with sparsity level $S = 6, 8$ and $\alpha = \frac{1}{32}$ in (27). This choice of α was recommended in [8]. All images used in these tests were scaled to have the unit maximum pixel value. The training data consists of 20,000 8×8 patches, that are 100 randomly extracted patches from each of the 200 images in the training set of the Berkeley segmentation dataset [18]. The 100 images in the testing set were used to measure the recovery performance.

In the first set of tests, we uniformly randomly extracted non-overlapping 600 patches from the 100 test images to recover using their measurements obtained with the four sensing matrices. Figure 3 depicts the mean squared error: MSE $= \sum_{i=1}^{\ell} \|x^i - \Psi\theta^i\|_2^2 / (64\ell)$, for sparsity level $S = 6, 8$ and measurement size $m = 16, 24$. Here, x^i is the vector obtained by reshaping the ith selected patch, $\ell = 600$ is the

---

[3]We also tested *opt-circ* and *coupled-circ*, and found that they performed as well as *opt-plus-rand-circ* and *coupled-plus-rand-circ* on average. However, they tend to cause the loss of small image features that may not be well described by the dictionary.
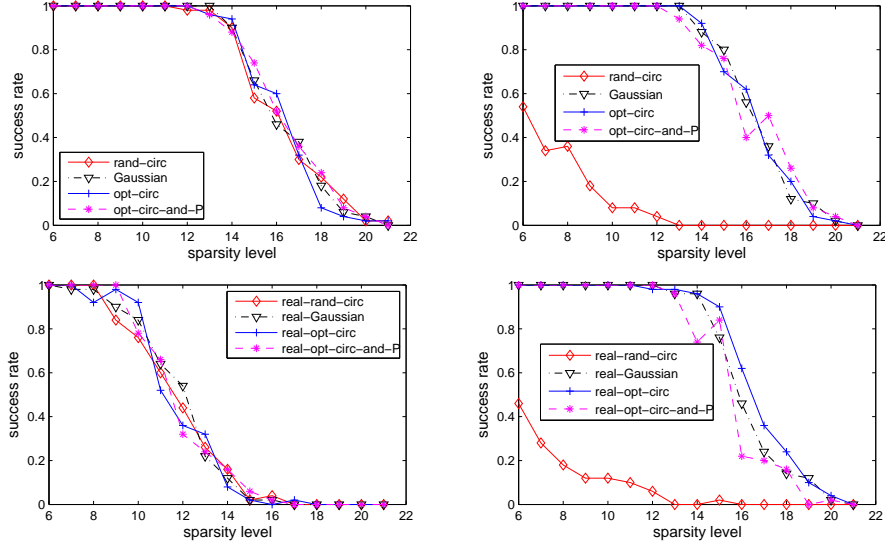
Figure 2. Comparison results of two groups of sensing matrices: *rand-circ*, *Gaussian*, *opt-circ*, *opt-circ-and-P*, and their real valued counterparts with Gaussian random basis (left) and Fourier basis (right).

number of tested patches, and θ^i was the YALL1 solution of

$$(30) \qquad \min_{\theta} \|\theta\|_1 + \frac{1}{\sigma}\|\Phi\Psi\theta - b\|_2^2,$$

where $b = \Phi(x^i + \sigma\eta)$ and $\sigma = \hat{\sigma}\|x^i\|_\infty / \|\eta\|_\infty$ with $\eta \sim \mathcal{N}(0, I)$ and $\hat{\sigma}$ varying among $\{0, 0.01, 0.05, 0.10, 0.15\}$. All results are the averages of 20 independent trials.

All the four pictures in Figure 3 reveal that both uncoupled and coupled learning approaches achieved significantly better recovery over random circulant matrices, and they did even better than Gaussian random matrices when $m = 16$. The coupled learning approach made slight improvement over the uncoupled one.

In the second set of tests, we chose two out of the 100 test images to recover using their measurements obtained with the four different sensing matrices. The first image had the resolution of 481×321, and the second one had the resolution of 321×481. For convenience, we removed the last row and the last column of both the two images and partitioned each images into 1,200 8×8 patches. Then we used YALL1 to solve (30) for each patch with the dictionary Ψ learned by KSVD with the sparsity level $S = 6$. We used peak-signal-to-noise-ratio (PSNR) and mean squared error (MSE) to measure the performance of recovery. Table 2 lists the average results of 20 independent trials for measurement size $m = 16, 24$ and noise level $\hat{\sigma} = 0, 0.01, 0.05, 0.10, 0.15$. One set of the recovered images are shown in Figure 4. From the results, we can see that both uncoupled and coupled learning approaches made significant improvement over random circulant matrices, especially for $m = 16$. When $\hat{\sigma} \geq 0.10$ or $m = 16$, the coupled learning approach did even better than random Gaussian matrices. Coupled learned circulant matrices tend to do better than Gaussian random ones when m is smaller. Although the best PSNRs were achieved with Gaussian random matrices, the best visual results are

(a) $m = 16$, $S = 6$ (b) $m = 24$, $S = 6$

(c) $m = 16$, $S = 8$ (d) $m = 24$, $S = 8$
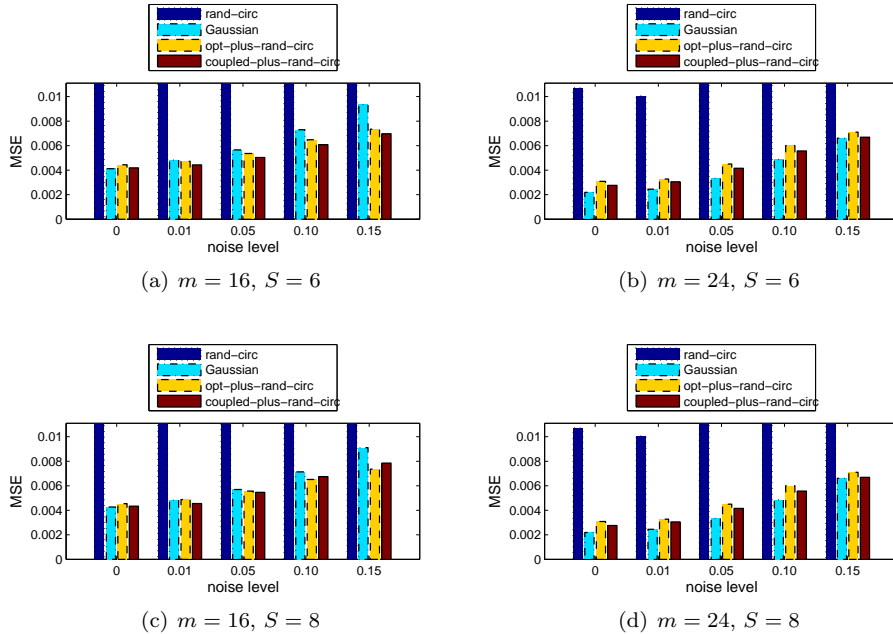
*Some MSEs are out of the display range and they are larger than 0.02.

FIGURE 3. Comparison results* of four different sensing matrices: *rand-circ*, *Gaussian*, *opt-plus-rand-circ* and *coupled-plus-rand-circ* on Berkeley segmentation dataset for different KSVD sparsity levels $S = 6, 8$ and different sampled row numbers $m = 16, 24$.

those with the coupled learned circulant matrices. In addition, the coupled approach did slightly better than the uncoupled one.

6.4. **Image tests with 2D circulant.** For the tests in Section 6.3, each selected patch was reshaped to a vector by stacking its columns. In this subsection, we compare *rand-2D-circ* and *opt-plus-rand-2D-circ* to illustrate how to directly apply 2D circulant operator on the squared patches. The dictionary learned by KSVD in Section 6.3 with the sparsity level $S = 6$ was used in this test. Note that each atom in the dictionary becomes a squared patch as opposed to a vector. We used the same 600 patches and the same two images as in Section 6.3 for this test. Instead of solving (30), now YALL1 was employed to solve

$$(31) \qquad \min_{\theta} \|\theta\|_1 + \frac{1}{\sigma}\|\mathcal{P}_\Omega \mathcal{C}_2 \mathcal{L} \mathcal{Q}(\theta) - b\|_2^2,$$

where $\mathcal{P}_\Omega \mathcal{C}_2 \mathcal{L}$ denotes normalized partial 2D circulant operator, $\mathcal{Q}$ is defined in Section 3.2, $b = \mathcal{P}_\Omega \mathcal{C}_2 \mathcal{L}(X^i + \sigma\eta)$ with the same η and σ to those in Section 6.3, and X^i is the ith tested patch corresponding to the reshaped vector x^i in (30). Figure 5 plots the average MSEs of 20 independent trials with the number of measurements $m = 16, 24$ and noise level $\hat{\sigma} = 0, 0.01, 0.05, 0.10, 0.15$, and Figure 6 plots one set of recovered images Castle and Gulf by YALL1 with the number of measurements $m = 24$ and noise level $\hat{\sigma} = 0.01$. We can see that similar results were obtained as in Section 6.3 where circulant sensing matrices were compared.

rand-circ
PSNR=25.74

Gaussian
PSNR=27.83

opt-plus-rand-circ
PSNR=27.38

coupled-plus-rand-circ
PSNR=27.58

rand-circ, PSNR=25.99

Gaussian, PSNR=28.05

opt-plus-rand-circ, PSNR=27.84
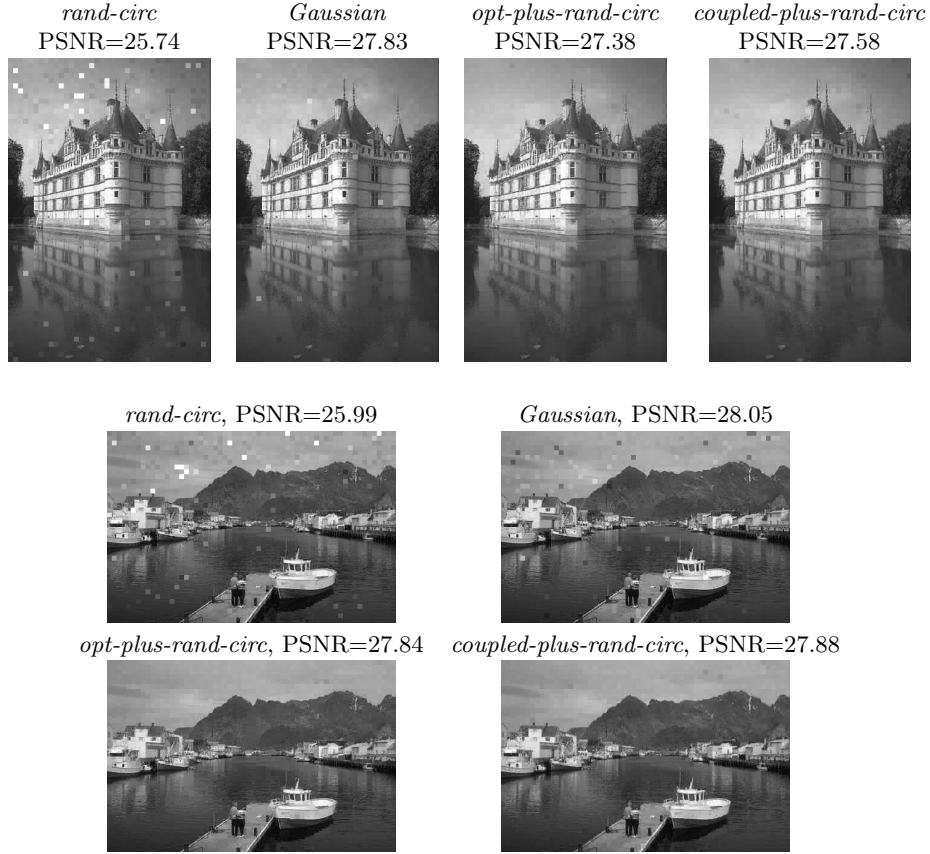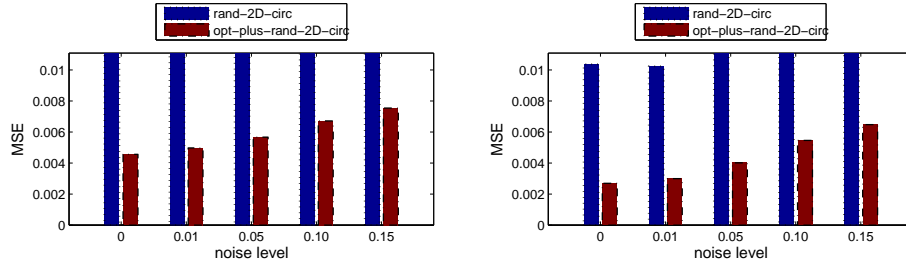
coupled-plus-rand-circ, PSNR=27.88

FIGURE 4. One set of recovered images Castle (upper four) and Gulf (lower four) by YALL1 with four different sensing matrices: *rand-circ*, *Gaussian*, *opt-plus-rand-circ*, *coupled-plus-rand-circ* for noise level $\hat{\sigma} = 0.01$, KSVD sparsity level $S = 6$ and selected row number $m = 24$

*Some MSEs are out of the display range and larger than 0.02 or even 0.04.

FIGURE 5. Comparison results* of solutions by YALL1 with two different 2D circulant operators: *rand-2D-circ* and *opt-plus-rand-2D-circ* from $m = 16$ (left) and $m = 24$ (right) measurements

Table 2. Comparison results of recovered images Castle and Gulf by YALL1 with four different sensing matrices: *rand-circ*, *Gaussian*, *opt-plus-rand-circ*, *coupled-plus-rand-circ*

Castle		rand-circ		Gaussian		opt-plus-rand-circ		coupled-plus-rand-circ	
m	$\hat{\sigma}$	PSNR	MSE	PSNR	MSE	PSNR	MSE	PSNR	MSE
16	0.00	19.41	4.78e-2	**25.58**	**2.96e-3**	25.03	3.16e-3	25.31	**2.96e-3**
16	0.01	18.61	4.36e-2	24.40	3.85e-3	24.41	3.66e-3	**24.72**	**3.39e-3**
16	0.05	17.87	5.06e-2	23.55	4.67e-3	23.56	4.44e-3	**23.95**	**4.05e-3**
16	0.10	16.88	5.54e-2	22.35	6.24e-3	22.70	5.40e-3	**23.05**	**4.98e-3**
16	0.15	16.22	5.87e-2	21.27	8.03e-3	22.15	6.12e-3	**22.42**	**5.76e-3**
24	0.00	25.66	8.57e-3	**28.77**	**1.33e-3**	27.02	2.00e-3	27.43	1.82e-3
24	0.01	24.05	8.31e-3	**27.22**	**1.91e-3**	26.05	2.57e-3	26.50	2.27e-3
24	0.05	22.07	1.82e-2	**25.58**	**2.79e-3**	24.31	3.77e-3	24.76	3.38e-3
24	0.10	20.15	3.17e-2	**23.91**	**4.11e-3**	22.99	5.06e-3	23.36	4.65e-3
24	0.15	18.82	4.08e-2	**22.56**	5.62e-3	22.28	5.95e-3	22.53	**5.61e-3**
Gulf		rand-circ		Gaussian		opt-plus-rand-circ		coupled-plus-rand-circ	
m	$\hat{\sigma}$	PSNR	MSE	PSNR	MSE	PSNR	MSE	PSNR	MSE
16	0.00	19.57	4.29e-2	**25.64**	**2.91e-3**	25.13	3.11e-3	25.36	2.93e-3
16	0.01	18.87	3.94e-2	24.48	3.74e-3	24.59	3.52e-3	**24.87**	**3.28e-3**
16	0.05	18.06	4.51e-2	23.65	4.55e-3	23.69	4.32e-3	**24.06**	**3.96e-3**
16	0.10	17.23	4.91e-2	22.51	5.93e-3	22.76	5.35e-3	**23.15**	**4.89e-3**
16	0.15	16.48	5.26e-2	21.39	7.83e-3	22.16	6.12e-3	**22.47**	**5.70e-3**
24	0.00	25.91	7.79e-3	**29.06**	**1.24e-3**	27.39	1.85e-3	27.58	1.76e-3
24	0.01	24.29	7.82e-3	**27.55**	**1.78e-3**	26.45	2.34e-3	26.77	2.13e-3
24	0.05	22.35	1.65e-2	**25.81**	**2.64e-3**	24.56	3.57e-3	25.01	3.20e-3
24	0.10	20.44	2.84e-2	**24.08**	**3.95e-3**	23.12	4.93e-3	23.55	4.47e-3
24	0.15	19.06	3.64e-2	**22.73**	**5.40e-3**	22.29	5.94e-3	22.68	5.44e-3

rand-2D-circ
PSNR = 25.53

opt-plus-rand-2D-circ
PSNR = 27.47

rand-2D-circ, PSNR = 25.54
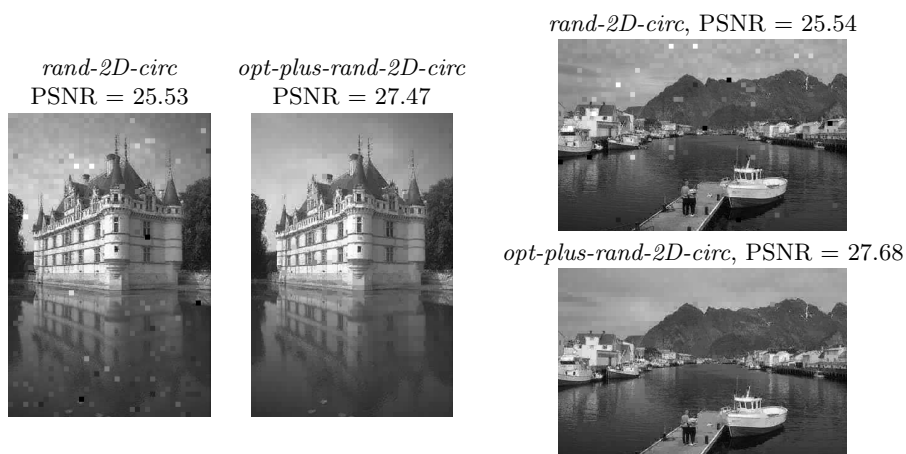
opt-plus-rand-2D-circ, PSNR = 27.68

Figure 6. One set of recovered images Castle (left two) and Gulf (right two) by YALL1 using *rand-2D-circ* and *opt-plus-rand-2D-circ* with $m = 24$ measurements and noise level $\hat{\sigma} = 0.01$

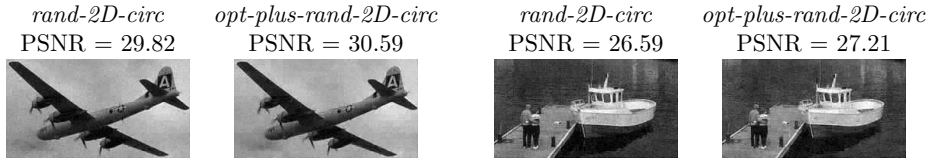| *rand-2D-circ* | *opt-plus-rand-2D-circ* | *rand-2D-circ* | *opt-plus-rand-2D-circ* |
| PSNR = 29.82 | PSNR = 30.59 | PSNR = 26.59 | PSNR = 27.21 |

FIGURE 7. One set of recovered images *Plane* (left two) and *Gulf* (right two) by YALL1 with two different 2D circulant operators: *rand-2D-circ* and *opt-plus-rand-2D-circ* for sample ratio SR = 0.30 and noise level $\hat{\sigma} = 0.01$

6.5. **Image-scale recovery.** In previous tests, we divided an image into a number of small patches and recovered each patch from its linear measurements, since the learned circulant sensing matrices/operators need to match those dictionary atoms in size. In practice, we often take measurements directly from the entire image instead of its small patches, so the previous learned patch-scale dictionaries do not match the image. In this section, we first use the method discussed in section 4 to generate an image-scale dictionary $\mathbf{\Psi}$ from the patch-scale one learned by KSVD in section 6.3 with sparsity level $S = 6$. Then we use $\mathbf{\Psi}$ to recover an image $\hat{X} \in \mathbb{C}^{N_1 \times N_2}$ from its measurements $b = \mathcal{G}(\hat{X} + \sigma \eta)$ by solving

$$(32) \qquad \min_{\theta} \|\theta\|_1 + \frac{1}{\sigma}\|\mathcal{G}\mathcal{Q}(\theta) - b\|_2^2,$$

where $\mathcal{G} = \mathcal{P}_{\Omega}\mathcal{C}_2\mathcal{L}$ is a normalized partial 2D circulant operator defined on $\mathbb{C}^{N_1 \times N_2}$, $\eta \sim \mathcal{N}(0, I)$ is Gaussian noise, $\sigma = \hat{\sigma}\|\hat{X}\|_{\infty}/\|\eta\|_{\infty}$, and $\mathcal{Q}$ is a 2D operator (corresponding to $\mathbf{\Psi}$) defined in the same way as in section 3.2.

At first glance, the recovery by solving (32) may take a lot of time since the dictionary $\mathbf{\Psi}$ has so many atoms. However, the sparsity structure of these atoms enables recoveries to be as fast as those using patch-size dictionaries. Note that fully random sensing operators are out of the game at the image scale since the number of free entries of each is at least ten thousand times of the image resolution, making such an operator impossible to fit in the memory of a typical workstation.

We compared *rand-2D-circ* and *opt-plus-rand-2D-circ* on two images. Both of them had the resolution of 241×129, and they were obtained by cropping[4] their larger originals Plane and Gulf in the testing set of Berkeley segmentation dataset. The noise level $\hat{\sigma}$ was set to $0, 0.01, 0.05, 0.10, 0.15$ and the sample ratio SR $:= \frac{|\Omega|}{N_1 N_2}$ to $0.20, 0.30$. Table 3 lists the average results of 20 independent trials, and Figure 7 plots one set of recovered images Plane (left two) and Gulf (right two) for $\hat{\sigma} = 0.01$ and SR = 0.30. From the table, we can see that the generated image-scale dictionary performed well in recovering the images with sufficiently many measurements, and that the optimized circulant operators did significantly better than the random ones on average.

Acknowledgments. The authors thank Prof. Guillermo Sapiro for his encouragement and very valuable suggestions, and Dr. Ming Yan for his comments on

---

[4]Though the recovery by solving (32) did not take much time, it took us quite a long time to obtain the optimized 2D circulant operator by the method discussed in section 3.2. The learning on larger images would take much more time. To save time, we simply used the cropped images to test our idea.

Table 3. Comparison results of recovered full-size images by
YALL1 with two different 2D circulant operators: *rand-2D-circ*
and *opt-plus-rand-2D-circ*; SR=sample ratio

Plane		*rand-2D-circ*		*opt-plus-rand-2D-circ*		Plane		*rand-2D-circ*		*opt-plus-rand-2D-circ*	
SR	$\hat\sigma$	PSNR	MSE	PSNR	MSE	SR	$\hat\sigma$	PSNR	MSE	PSNR	MSE
0.20	0.00	22.35	3.54e-2	26.87	2.01e-3	0.30	0.00	28.35	3.47e-3	30.01	1.00e-3
0.20	0.01	22.50	3.31e-2	26.78	2.10e-3	0.30	0.01	28.57	3.00e-3	30.05	9.93e-4
0.20	0.05	22.11	3.24e-2	26.23	2.38e-3	0.30	0.05	27.73	3.59e-3	29.20	1.21e-3
0.20	0.10	21.16	3.41e-2	25.04	3.14e-3	0.30	0.10	25.99	5.22e-3	27.48	1.80e-3
0.20	0.15	20.12	3.67e-2	23.84	4.13e-3	0.30	0.15	24.38	7.34e-3	25.94	2.56e-3
Gulf		*rand-2D-circ*		*opt-plus-rand-2D-circ*		Gulf		*rand-2D-circ*		*opt-plus-rand-2D-circ*	
SR	$\hat\sigma$	PSNR	MSE	PSNR	MSE	SR	$\hat\sigma$	PSNR	MSE	PSNR	MSE
0.20	0.00	21.33	1.61e-2	24.62	3.46e-3	0.30	0.00	25.58	5.00e-3	27.00	2.00e-3
0.20	0.01	21.38	1.59e-2	24.52	3.53e-3	0.30	0.01	25.66	4.67e-3	26.97	2.02e-3
0.20	0.05	21.19	1.62e-2	24.31	3.71e-3	0.30	0.05	25.32	4.95e-3	26.65	2.17e-3
0.20	0.10	20.68	1.72e-2	23.71	4.26e-3	0.30	0.10	24.34	6.24e-3	25.75	2.67e-3
0.20	0.15	20.02	1.87e-2	22.98	5.04e-3	0.30	0.15	23.28	7.90e-2	24.75	3.37e-3

REFERENCES

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

[2] W. Bajwa, J. Haupt, G. Raz, S. Wright and R. Nowak, Toeplitz-structured compressed sensing matrices, *IEEE Workshop on Statistical Signal Processing (SSP), Madison, Wisconsin, August 2007*, (2007), 294–298.

[3] R. Baraniuk and P. Steeghs, Compressive radar imaging, *IEEE Radar Conference, Waltham, Massachusetts*, (2007), 128–133.

[4] R. Bixby, Z. Gu and E. Rothberg, *Gurobi optimization*, www.gurobi.com, 2009–2011.

[5] E. Candes, J. Romberg and T. Tao, Stable signal recovery from incomplete and inaccurate information, *Communications on Pure and Applied Mathematics*, **59** (2006), 1207–1223.

[6] M. Combescure, Block-circulant matrices with circulant blocks, weil sums, and mutually unbiased bases. II. The prime power case, *Journal of Mathematical Physics*, **50** (2009), 1–12.

[7] D. Donoho, Compressed sensing, *IEEE Transactions on Information Theory*, **52** (2006), 1289–1306.

[8] J. M. Duarte-Carvajalino and G. Sapiro, Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization, *IEEE Transactions on Image Processing*, **18** (2009), 1395–1408.

[9] M. Elad, Optimized projections for compressed sensing, *IEEE Transactions on Signal Processing*, **55** (2007), 5695–5702.

[10] M. Elad and M. Aharon, Image denoising via sparse and redundant representations over learned dictionaries, *IEEE Transactions on Image Processing*, **15** (2006), 3736–3745.

[11] R. M. Gray, Toeplitz and circulant matrices: A review, *Communications and Information Theory*, **2** (2005), 155–239.

[12] J. Haupt, W. U. Bajwa, G. Raz and R. D. Nowak, Toeplitz compressed sensing matrices with applications to sparse channel estimation, *IEEE Transactions on Information Theory*, **56** (2010), 5862–5875.

[13] M. A. Herman and T. Strohmer, High-resolution radar via compressed sensing, *IEEE transactions on signal processing*, **57** (2009), 2275–2284.

[14] D. Liang, G. Xu, H. Wang, K. F. King and L. Ying, Toeplitz random encoding MR imaging using compressed sensing, *IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, (2009), 270–273.

[15] S. G. Mallat and Z. Zhang, Matching pursuits with time-frequency dictionaries, *IEEE Transactions on Signal Processing*, **41** (1993), 3397–3415.

[16] R. Marcia, Z. Harmany and R. Willett, Compressive coded aperture imaging, in *SPIE Electronic Imaging*, 2009.

[17] R. Marcia and R. Willett, Compressive coded aperture superresolution image reconstruction, *ICASSP*, (2008), 833–836.

[18] D. Martin, C. Fowlkes, D. Tal and J. Malik, A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics, *ICCV*, **2** (2001), 416–423.

[19] J. Meng, Y. Li, N. Nguyen, W. Yin and Z. Han, High resolution OFDM channel estimation with low speed ADC using compressive sensing, *IEEE ICC 2011 Signal Processing for Communications Symposium*, (2011), 1–6.

[20] J. Meng, W. Yin, Y. Li, N. Nguyen and Z. Han, Compressive sensing based high resolution channel estimation for OFDM system, *IEEE Journal of Selected Topics in Signal Processing, Special Issue on Robust Measures and Tests Using Sparse Data for Detection and Estimation*, **6** (2012), 15–25.

[21] J. F. Murray and K. Kreutz-Delgado, Sparse image coding using learned overcomplete dictionaries, in *Proceedings of the 2004 14th IEEE Signal Processing Society Workshop*, 2004, 579–588.

[22] B. A. Olshausen and D. J. Field, Emergence of simple-cell receptive field properties by learning a sparse code for natural images, *Nature*, **381** (1996), 607–609.

[23] H. Rauhut, J. Romberg and J. A. Tropp, Restricted isometries for partial random circulant matrices, *Applied and Computational Harmonic Analysis*, **32** (2012), 242–254.

[24] J. Romberg, Compressive sensing by random convolution, *SIAM Journal on Imaging Sciences*, **2** (2009), 1098–1128.

[25] A. C. Sauve, A. O. Hero III, W. L. Rogers, S. J. Wilderman and N. H. Clinthorne, 3D image reconstruction for a compton spect camera model, *IEEE Transactions on Nuclear Science*, **46** (1999), 2075–2084.

[26] J. Tropp, Greed is good: Algorithmic results for sparse approximation, *IEEE Transactions on Information Theory*, **50** (2006), 2231–2242.

[27] J. Tropp, M. Wakin, M. Duarte, D. Baron and R. Baraniuk, Random filters for compressive sampling and reconstruction, in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Vol. 3*, Toulouse, France, 2006, 872–875.

[28] E. Van den Berg and M. P. Friedlander, SPGL1: A MATLAB solver for large-scale sparse reconstruction, 2007. Available from: `http://www.cs.ubc.ca/labs/scl/index.php/main/spgl1`.

[29] R. M. Willett, R. F. Marcia and J. M. Nichols, Compressed sensing for practical optical imaging systems: A tutorial, *Optical Engineering*, **50** (2011), 1–13.

[30] S. Wright, R. Nowak and M. Figueiredo, Sparse reconstruction by separable approximation, *IEEE Transactions on Signal Processing*, **57** (2009), 2479–2493.

[31] J. Yang and Y. Zhang, Alternating direction algorithms for ℓ_1-problems in compressive sensing, *SIAM Journal on Scientific Computing*, **33** (2011), 250–278.

[32] W. Yin, Analysis and generalizations of the linearized Bregman method, *SIAM Journal on Imaging Sciences*, **3** (2010), 856–877.

[33] W. Yin, S. P. Morgan, J. Yang and Y. Zhang, Practical compressive sensing with Toeplitz and circulant matrices, in *Proceedings of Visual Communications and Image Processing (VCIP)*, 2010.

[34] W. Yin, S. Osher, D. Goldfarb and J. Darbon, Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing, *SIAM Journal on Imaging Sciences*, **1** (2008), 143–168.

[35] W. Yin, Gurobi mex: A matlab interface for gurobi, 2009–2011. Available from: `http://www.convexoptimization.com/wikimization/index.php/gurobi_mex`.

E-mail address: yangyang.xu@rice.edu
E-mail address: wotaoyin@math.ucla.edu
E-mail address: sjo@math.ucla.edu