LEARNING CIRCULANT SENSING KERNELS

YANGYANG XU*, WOTAO YIN* , AND STANLEY OSHER†

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 [7], 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 Gaussian random sensing matrix. In addition, by exploiting the circulant structure, we improve the learning from the patch scale in [7] 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.

Key words. compressive sensing, sensing operator learning, sensing kernel learning, circulant matrix, Toeplitz matrix, dictionary learning

1. Introduction. Compressive sensing (CS) ([6, 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 \( \Psi \), i.e., one can write \( \bar{x} = \Psi \theta \) with a sparse \( \theta \). Let \( b := \Phi \bar{x} \) represent a set of \( m \) linear projections of \( \bar{x} \). The basis pursuit problem

\[
\text{BP: } \min_{\theta} \|\theta\|_1 \quad \text{s.t. } \Phi \Psi \theta = b, 
\]

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

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 \( \Phi \) 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. Toeplitz and circulant matrices have the forms

\[
T = \begin{bmatrix} t_n & t_{n-1} & \cdots & t_1 \\ t_{n+1} & t_n & \cdots & t_2 \\ \vdots & \vdots & \ddots & \vdots \\ t_{2n-1} & t_{2n-2} & \cdots & t_n \end{bmatrix} \quad \text{and} \quad C = \begin{bmatrix} t_n & t_{n-1} & \cdots & t_1 \\ t_1 & t_n & \cdots & t_2 \\ \vdots & \vdots & \ddots & \vdots \\ t_{n-1} & t_{n-2} & \cdots & t_n \end{bmatrix},
\]

respectively. When matrix \( T \) satisfies the additional property that \( t_i = t_{n+i}, \forall i \), it becomes a circulant matrix \( C \). 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 \( c \ast \bar{x} \) for a certain vector \( c \); hence, if \( P \) is a row-selection (downsampling)

---

*yangyang.xu@rice.edu and wotao.yin@rice.edu. Department of Applied and Computational Mathematics, Rice University.
†sjo@math.ucla.edu. Department of Mathematics, UCLA, and Level Set Systems, Inc., Los Angeles, CA.

\(^1\)By “a partial matrix”, we refer to a submatrix formed by selecting \( m \) out of the \( n \) rows of the original matrix, where \( m < n \). Given a row selection operator \( P \) and an original matrix \( M \), \( PM \) is a partial matrix.
Learning Circulant Sensing Kernels

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 [7], 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 Gaussian random sensing matrix. In addition, by exploiting the circulant structure, we improve the learning from the patch scale in [7] 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.
operator, $PCx$ becomes circulant CS measurements of $x$. Using either Toeplitz or circulant matrices, Tropp et al.\cite{24} describes a random filter for acquiring a signal $x$; Haupt et al.\cite{10} 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.\cite{18,19} 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 \cite{3,12,16,15,23}. Furthermore, random convolutions can be realized by an optical correlator \cite{23}. Since any matrix $C$ in (1.2) can be diagonalized by a Fourier transform, i.e., obeying

$$C = FDF^*, \quad (1.3)$$

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). Recently, the use of Toeplitz and circulant matrices has been proposed for compressive MR imaging by Liang et al.\cite{13}. A good review can be found in \cite{26}.

There are rich theoretical results on circulant matrices for CS. In \cite{2}, Toeplitz measurement matrices are constructed with i.i.d. random entries of $\pm 1$ or $\{-1,0,1\}$; their downsampling effectively selects the first $m$ rows; and the number of measurements needed for stable $\ell_1$ recovery is shown to be $m \geq O(k^3 \cdot \log n/k)$, where $k$ is the signal sparsity. \cite{11} 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 $\ell_1$ recovery. The work \cite{19} establishes stable recovery under the condition $m \geq O(k^2 \log(n/k))$. In \cite{23}, 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 basis with high probability and $\ell_1$ recovery is stable given $m \geq O(k \cdot \log n + \log^3 n)$. Recent results in \cite{22} show that several random circulant matrices satisfy the restricted isometry property (RIP) in expectation given $m \geq O(\max\{s^{3/2} \log^{3/2} n, k \log^2 k \log^2 n\})$ with arbitrary downsampling. 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., \cite{25,27,28,31,29}) for CS are based on performing operations including multiplications involving with $\Phi$ and $\Phi^*$. For partial circulant matrix $\Phi = PC$, $\Phi x$ 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 \cite{30} 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 \cite{21,9,20} 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 $\Psi$ and a sensing matrix $\Phi$: with
\( D := \Phi \Psi = [d_1, \ldots, d_K] \), it is defined as \([14]\)

\[
\mu(D) := \max_{i,j, 1 \leq i, j \leq K} \frac{|d_i^*d_j|}{||d_i||_2||d_j||_2}. \tag{1.4}
\]

A smaller \( \mu(D) \) tends to allow more \( \Psi \)-sparse signals \( \bar{x} \) to be successfully recovered from measurements \( \Phi \bar{x} \) via various CS algorithms. Hence, Elad \([8]\) seeks means to reduce \( \mu(D) \). He considers the Gram matrix \( G = D^*D \) and proposes to reduce \( \mu_4(D) := \langle (\sum_{i,j, 1 \leq i, j \leq K}(g_{ij} > t)) \cdot |g_{ij}| \rangle / (\sum_{i,j, 1 \leq i, j \leq K}(|g_{ij}| > t)) \). His work demonstrates improved recovery quality with learned (non-circulant) sensing matrices \( \Phi \), and it has motivated the subsequent work \([7]\). Work \([7]\) is not based on minimizing \( (\Phi \Psi)^*(\Phi \Psi) \approx I \), where \( I \) denotes the identity matrix. The results of \([7]\) are even better since \( \mu(D) \) is a worst-case characteristic whereas \( (\Phi \Psi)^*(\Phi \Psi) \approx I \) is more toward improving the overall performance. Also, the latter is easier to compute.

### 1.3. Contributions

We are motivated by \([7]\) and propose numerical methods to minimize \( \|((\Phi \Psi)^*(\Phi \Psi)−I)\|_F \), where \( \Phi \) is either a full or partial circulant matrix. Like \([7]\), we also learn \( \Phi \) and \( \Psi \) together, performing the so-called the coupled learning of \( \Phi \) and \( \Psi \). While results of \([7]\) are limited to one dimensional case, 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. Current atoms are patches, e.g., of size \( 8 \times 8 \). We further address the patch-scale limitation of \([7]\), namely, the sensing matrix size \( n \) must equal the dictionary atom size; namely, if the dictionary for a \( 512 \times 512 \) image is formed by \( 8 \times 8 \) patches, the sensing matrix generated in \([7]\) has only \( 64 = 8 \times 8 \) columns and applies to vectors of length \( 64 \), instead of \( 512 \times 512 \). 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 \([17]\). 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. \(\text{conj}(\cdot)\) stands for the conjugate operator. Define \( A \cdot B = \sum_{i,j} A_{ij}B_{ij} \) and \( \langle A, B \rangle = \sum_{i,j} \text{conj}(A_{ij})B_{ij} \) for any two matrices \( A, B \) of the same dimension. \( \text{vec}(A) \) is a vector formed by stacking all the columns of \( A \), and \( \text{diag}(\cdot) \) is defined in the same way as MATLAB function \( \text{diag} \), 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 describes how the involved optimization problems are solved. Numerical results are presented in Section 5.

### 2. Overview of Circulant Correlations

Given a kernel \( v \in \mathbb{C}^n \), the circulant correlation \( x \ast v \) for a vector \( x \in \mathbb{C}^n \) is defined as

\[
(x \ast v)_k = \sum_{i=1}^{n} x_i v_{k_i}, \text{ for } k = 1, \ldots, n,
\tag{2.1}
\]

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

\[
(x \ast c)_k = \sum_{i=1}^{n} x_i c_{k_i}, \text{ for } k = 1, \ldots, n,
\]
where \( k_i = \text{mod}(n + k - i - 1, n) + 1 \).

We next introduce three methods to compute \( x \ast v \). First, introducing the \( n \times n \) cyclic permutation matrix

\[
P_n = \begin{bmatrix}
0 & 0 & \cdots & 0 & 1 \\
1 & 0 & \cdots & 0 & 0 \\
0 & \ddots & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \ddots & 0 \\
0 & \cdots & 0 & 1 & 0
\end{bmatrix},
\]

we can write formula (2.1) as

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

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 \( x = [-1, 0, 1]^\top \), \( y = x \ast v \) equals

\[
y_1 = x^\top P_n^0 v = -1 \cdot 1 + 0 \cdot 2 + 1 \cdot 3 = 2,
\]

\[
y_2 = x^\top P_n^3 v = -1 \cdot 3 + 0 \cdot 1 + 1 \cdot 2 = -1,
\]

\[
y_3 = x^\top P_n^2 v = -1 \cdot 2 + 0 \cdot 3 + 1 \cdot 1 = -1.
\]

Secondly, we can compute \( x \ast v \) via the matrix-vector multiplication \( Cx \) with the circulant matrix

\[
C = \begin{bmatrix}
v_1 & v_2 & \cdots & v_n \\
v_n & v_1 & \cdots & v_{n-1} \\
\vdots & \vdots & \ddots & \vdots \\
v_2 & v_3 & \cdots & v_1
\end{bmatrix}.
\]

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

**Lemma 2.1.** Any circulant matrix \( C \) in (2.3) can be written as \( C = FDF^* \), where \( F \) is the \( n \times n \) unitary discrete Fourier transformation matrix and \( D = \text{diag}(d) \) where \( d = \sqrt{n}Fv \).

**Remark 2.1.** The matrix \( C \) is real-valued if \( d \) is conjugate symmetric, namely, \( d_i = \text{conj}(d_{i'}) \) for every \( i \) and \( i' = \text{mod}(n - i + 1, n) + 1 \). 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 \( Y = X \ast M \) for a given matrix \( X \in \mathbb{C}^{n_1 \times n_2} \) is defined by

\[
(X \ast M)_{ts} = \sum_{i,j} X_{ij} M_{t_i,s_j}, \text{ for } 1 \leq t \leq n_1, 1 \leq s \leq n_2,
\]

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

Similar to 1D circulant, \( X \ast M \) can be computed in three ways. First, with cyclic permutation matrices \( P_{n_1} \) and \( P_{n_2} \), formula (2.4) can be written as

\[
(X \ast M)_{ts} = X \ast (P_{n_1}^{t-1} M (P_{n_2}^\top)^{s-1}), \text{ for } 1 \leq t \leq n_1, 1 \leq s \leq n_2.
\]

For instance, if

\[
X = \begin{bmatrix}
1 & -1 & 0 \\
-2 & 0 & 1
\end{bmatrix} \text{ and } M = \begin{bmatrix}
1 & 2 & 3 \\
4 & 5 & 6
\end{bmatrix},
\]
then $Y = X \ast M$ has components

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

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

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

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

$$
C_2 = [\text{vec}(M^{1,1}), \text{vec}(M^{2,1}), \ldots, \text{vec}(M^{n_1,1}), \ldots, \text{vec}(M^{1,s}), \ldots, \text{vec}(M^{n_1,s}), \ldots, \text{vec}(M^{n_1,n_2})] ^\top ,
$$

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

$$
C_2 = \begin{bmatrix} 1 & 4 & 2 & 5 & 3 & 6 \\ 4 & 1 & 5 & 2 & 6 & 3 \\ 3 & 6 & 1 & 4 & 2 & 5 \\ 6 & 3 & 4 & 1 & 5 & 2 \\ 2 & 5 & 3 & 6 & 1 & 4 \\ 5 & 2 & 6 & 3 & 4 & 1 \end{bmatrix}
$$

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

Thirdly, we can quickly compute $X \ast M$ 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 $C_2$ on $\mathbb{C}^{n_1 \times n_2}$ by $C_2(X) = X \ast M$ for $X \in \mathbb{C}^{n_1 \times n_2}$. Then $C_2$ can be represented as $C_2 = F_2 V F_2^\top$, where $F_2$ is the orthogonal 2D discrete Fourier transformation operator on $\mathbb{C}^{n_1 \times n_2}$, $F_2^\top$ is the adjoint operator of $F_2$ as well as its inverse operator, and $V$ is an operator on $\mathbb{C}^{n_1 \times n_2}$ defined by $V(X) = V \odot X$ for any $X \in \mathbb{C}^{n_1 \times n_2}$ with $V = \sqrt{m_1 m_2} F_2(M)$.

**Remark 2.2.** The block-circulant matrix $C_2$ corresponding to $C_2$ is real valued if $V$ is conjugate symmetric, namely, $V_{i,j} = V_{i',j'}$ for every pair of $i, j$ and $i' = \text{mod}(n_1 - i + 1, n_1) + 1, j' = \text{mod}(n_2 - j + 1, n_2) + 1$.

3. **Learning Circulant Sensing Matrix/Operator.** In this section, we illustrate how to learn the 1D kernel $v$ in formula (2.1) and the 2D kernel $M$ in formula (2.4), 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 $\mathcal{P}_\Omega C_2$, 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,

$$
X = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}
$$

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

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 [7], we would like to design a partial circulant sensing matrix $\Phi = PC$ such that $\Psi^\ast \Phi^\ast \Phi \Psi \approx I$, where $C$ is an $n \times n$ circulant matrix and $P$ is an $m \times n$ downsampling matrix. To construct a partial circulant matrix $\Phi$, we shall determine $P$ and $C$. We first learn the circulant matrix $C$ and then the downsampler $P$. 

5
According to Lemma 2.1, $C = FDF^*$ where the diagonal matrix $D = \text{diag}(d)$ with entries $d = \sqrt{n}Fv$ 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

$$
\min_C \|\Psi^* C^* C \Psi - I\|_F, \tag{3.1}
$$

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

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

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

$$
x := ([d_1]^2, [d_2]^2, \ldots, [d_n]^2)^\top \geq 0, \tag{3.2}
$$

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

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

Hence, we reduce (3.1) to

$$
\min_{x \geq 0} \frac{1}{2} x^\top \bar{B} x - x^\top \text{diag}(B). \tag{3.3}
$$

Problem (3.3) is a convex quadratic program and can be reformulated as a non-negative least-squares problem.

Given a solution $x = x_{\text{opt}}$ to (3.3), any $d$ obeying (3.2) gives $C = F \text{diag}(d) F^*$ as a solution to (3.1). 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. Models (3.1) and (3.3) lead to significant improvement in sensing efficiency as we shall demonstrate by numerical examples in Section 5.

**Remark 3.1.** The above procedure generates a complex-valued $C$. To obtain a real-valued $C$, one shall add constraints $x_i = x_{n-i+2}$ for $i = 2, \ldots, n$ in the problem (3.3) and then generate a conjugate symmetric $d$ from the solution $x_{\text{opt}}$.

We also tested an alternative model:

$$
\min_C \frac{1}{2} \|C \Psi - I\|_F^2. \tag{3.4}
$$

which is equivalent to

$$
\min_{d \in \mathbb{C}^n} \frac{1}{2} d^* A d - \frac{1}{2} d^\top \text{diag}(F^* \Psi F) - \frac{1}{2} \text{conj}(d^\top \text{diag}(F^* \Psi F)), \tag{3.5}
$$

where $A$ is a diagonal matrix with diagonal entries given by vector $\text{diag}(F^* \Psi F)$. Since (3.5) is component-wise separable in $d_i$, it is easy to derive its closed-form solution

$$
d_{\text{opt}} = \text{conj}(\text{diag}(F^* \Psi F)) \odot \text{diag}(F^* \Psi F). \tag{3.6}
$$
However, our numerical experience suggests that this solution (and thus model (3.4)) is not as effective as that of (3.1).

After the circulant matrix $C$ is determined, we now optimize $P$, and thus fully determine $\Phi$. 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 $\|\Psi^*\Phi^*\Phi - I\|_F$ so that $\Psi^*\Phi^*\Phi \approx I$. Let $p_i$ be the binary variable indicating the selection of element $i$, i.e.,

$$p_i = \begin{cases} 
1, & \text{matrix } P \text{ selects row } i, \\
0, & \text{otherwise}. 
\end{cases} \quad (3.7)$$

Since $\Phi = PC$, we shall solve

$$\min_p \frac{1}{2}\|\Psi^*C^*(P^*P)C\Psi - I\|_F^2 = \left\|\sum_{i=1}^np_i(q_iq_i^*) - I\right\|_F^2 \quad (3.8)$$

where $q_i$ is the $i$-th column of $\Psi^*C^*$. Model (3.8) is equivalent to the binary quadratic program

$$\min_p \|\text{vec}(q_1q_1^*), \ldots, \text{vec}(q_nq_n^*)\|_F - \|\text{vec}(I)\|_F^2, \text{ subject to } \sum_i p_i = m \text{ and } p_i \in \{0, 1\}, \forall i. \quad (3.9)$$

Here, $[\text{vec}(q_1q_1^*), \ldots, \text{vec}(q_nq_n^*)]$ is a matrix of $n^2$ rows and $n$ columns. By simple calculation, we can write (3.9) into the following equivalent problem

$$\min_p p^\top Hp - 2f^\top p, \text{ subject to } \sum_i p_i = m \text{ and } p_i \in \{0, 1\}, \forall i, \quad (3.10)$$

where $H = [G_{ij}]$, $f = \text{diag}(G)$ and $G = C\Psi^*C^*$. Problem (3.10) 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.

2D circulant operator $C_2$ is often used to sense images and videos. 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 $Q$ on $\mathbb{C}^K$ by $Q(\theta) = \sum_{i=1}^K \theta_i \psi_i$ for $\theta \in \mathbb{C}^K$. An array $X \in \mathbb{C}^{n_1 \times n_2}$ is sparse with respect to the dictionary $\Psi$ if it can be represented as $X = Q(\theta)$ 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

$$\min_\theta \|\theta\|_1, \text{ subject to } \mathcal{P}_\Omega C_2 Q(\theta) = b, \quad (3.11)$$

where $b = \mathcal{P}_\Omega C_2(X)$ contains the measurements of $X$. To improve the recoverability of the $\ell_1$ optimization problem (3.11), we shall try to make $(C_2^* Q)^* C_2 Q$ close to the identity operator $I$ on $\mathbb{C}^K$. The adjoint operator of $Q$ is defined as

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

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

$$(C_2 Q)^* C_2 Q(\theta) = \begin{bmatrix} \langle C_2(\psi_1), \sum_{i=1}^K \theta_i C_2(\psi_i) \rangle, \langle C_2(\psi_2), \sum_{i=1}^K \theta_i C_2(\psi_i) \rangle, \ldots, \langle C_2(\psi_K), \sum_{i=1}^K \theta_i C_2(\psi_i) \rangle \end{bmatrix}^\top,$$

and making $(C_2 Q)^* C_2 Q \approx I$ is equivalent to making $\langle C_2(\psi_j), \sum_{i=1}^K \theta_i C_2(\psi_i) \rangle \approx \theta_j$ for $1 \leq j \leq K$. Toward this goal, we solve

$$\min_G \|G - I\|_F^2, \text{ subject to } G_{ls} = \langle C_2(\psi_l), C_2(\psi_s) \rangle, \quad (3.12)$$
which can simplified as follows. According to Lemma 2.2, we have

$$\langle C_2(\psi_1), C_2(\psi_s) \rangle = \langle F_2 V F_2^*(\psi_1), F_2 V F_2^*(\psi_s) \rangle = \langle V F_2^*(\psi_1), V F_2^*(\psi_s) \rangle = \langle V^* V F_2^*(\psi_1), F_2^*(\psi_s) \rangle.$$  

Let $U = \text{conj}(V) \odot V$ and define $U := V^* V$, where $V^*$ is the adjoint operator of $V$. We have $U(X) = U \odot X$ 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 $y^s = \text{vec}(F_2^*(\psi_s))$ for $s = 1, 2, \ldots, K$. Then

$$G_{ts} = \langle V^* V F_2^*(\psi_1), F_2^*(\psi_s) \rangle = (u \odot y^t)^* y^s = u^* (\text{conj}(y^t) \odot y^s).$$

Hence,

$$||G||_F^2 = \sum_{t,s} \text{conj} (G_{ts}) \cdot G_{ts}$$

$$= \sum_{t,s} \text{conj} (u^* (\text{conj}(y^t) \odot y^s)) \cdot (u^* (\text{conj}(y^t) \odot y^s))$$

$$= \sum_{t,s} u^* ((\text{conj}(y^t) \odot y^s)(\text{conj}(y^t) \odot y^s))^* u$$

$$= \sum_{t,s} u^* (\text{conj}(y^t(y^t)^*)(y^s(y^s)^*)) u = u^* A u,$$

where $A := \sum_{t,s} \text{conj}(y^t(y^t)^*)(y^s(y^s)^*) = \text{conj}(YY^*) \odot (YY^*)$. 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 (3.12) is equivalent to the convex quadratic program

$$\min_{u \geq 0} \frac{1}{2} u^T A u - f^T u$$  \hspace{1cm} (3.13)$$

in the same form of (3.3), 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 (3.13), we can obtain matrix $U$ via $u = \text{vec}(U)$. Any $V$ satisfying $U = \text{conj}(V) \odot V$ defines $C_2 = F_2 V F_2^*$ as a solution to (3.12). Like done in Section 3.1 for 1D circulant, we choose to generate the phase of each entry $V_{ns}$ uniformly at random.

**Remark 3.2.** The above procedure gives a complex-valued $C_2$. To obtain a real-valued $C_2$, we can add constraints $u_{n_1-1} = u_{n_1-1}^{t'}$ for every pair of $i, j$ and $t' = \text{mod}(n_1 - i + 1, n_1) + 1, j' = \text{mod}(n_2 - j + 1, n_2) + 1$ to (3.13), and then generate a conjugate symmetric $V$ from the solution $u_{\text{opt}}$.

Given $C_2$, we can choose $\Omega$ uniformly at random or optimize $P_\Omega$ toward $(P_\Omega C_2 Q)^*(P_\Omega C_2 Q) \approx I$, which can be achieved by solving

$$\min_{G} ||G - I||_F^2, \text{ subject to } G_{ts} = \langle P_\Omega C_2(\psi_1), P_\Omega C_2(\psi_s) \rangle.$$  \hspace{1cm} (3.14)$$

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

$$(D_\Omega)_{k_i k_j} = \begin{cases} 1, & \text{if } (i, j) \in \Omega, \\ 0, & \text{otherwise}, \end{cases} \quad \text{for } 1 \leq i \leq n_1, 1 \leq j \leq n_2, \quad (3.15)$$

where $k_{ij} = i + (j - 1)n_1$. Then $\langle D_\Omega \text{vec}(X), D_\Omega \text{vec}(Y) \rangle = \langle P_\Omega X, P_\Omega Y \rangle$ for any $X, Y \in \mathbb{C}^{n_1 \times n_2}$. Letting $y^t = \text{vec}(C_2(\psi_t))$ for $t = 1, \ldots, K$, we have

$$G_{ts} = \langle P_\Omega C_2(\psi_1), P_\Omega C_2(\psi_s) \rangle = \langle D_\Omega y^t, D_\Omega y^s \rangle = (y^t)^* D_\Omega y^s.$$
Let \( Y = \sum_s y_s(y^s)^* \). Then
\[
\|G\|_F^2 = \sum_{i,s} \text{conj}(G_{is}) \cdot G_{is} = \sum_{i,s} ((y^i)^* D_\Omega y^i) \cdot ((y^s)^* D_\Omega y^s)
\]
\[
= \sum_i (y^i)^*(D_\Omega Y D_\Omega) y^i = \sum_i \text{tr} (D_\Omega Y D_\Omega (y^i)^*) = \text{tr} (D_\Omega Y D_\Omega Y)
\]
\[
= \sum_{i,j} (D_\Omega Y D_\Omega)_{ij} = \sum_{i,j} (D_\Omega Y)_{ij} \cdot (D_\Omega Y)_{ji} = \sum_{i,j} (D_\Omega)_{ij} \cdot Y_{ij} \cdot (D_\Omega)_{ji} \cdot Y_{ji}
\]
\[
= \sum_{i,j} (D_\Omega)_{ij} \cdot Y_{ij} \cdot \text{conj}(Y_{ij}) \cdot (D_\Omega)_{ji} = p^\top H p,
\]
where \( H = [|Y_{ij}|^2] \) and \( p = \text{diag}(D_\Omega) \). In addition,
\[
\text{tr}(G) = \sum_i G_{ii} = \sum_i (y^i)^* D_\Omega y^i = \sum_i \text{tr}(D_\Omega y^i (y^i)^*) = \text{tr}(D_\Omega Y) = f^\top p,
\]
where \( f = \text{diag}(Y) \). Hence, (3.14) is equivalent to
\[
\min_p \frac{1}{2} p^\top H p - f^\top p, \quad \text{subject to } \sum_k p_k = |\Omega| \text{ and } p_k \in \{0, 1\}, \forall k. \tag{3.16}
\]

4. Algorithm and Implementation. With a given dictionary \( \Psi \), Algorithm 1 outlines our approach for optimizing a partial circulant matrix \( \Phi \).

**Algorithm 1**

1. Solve (3.3) for \( x \), generate randomly-phased \( d \) from \( x \) via (3.2), and then form \( C = F\text{diag}(d)F^* \).
2. Solve (3.10) for \( p \) or randomly generate \( p \) and then form \( P \) from \( p \) via (3.7).
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 \texttt{quadprog} to solve (3.3) with its default settings. At Step 2, we use the commercial code Gurobi [4] with MATLAB interface [32] to solve the binary program (3.10). 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, (3.10) was only approximately solved.

We used both synthetic and real data for test. For synthetic data, \( \Psi \) was Gaussian randomly generated basis or discrete Fourier basis, and \( \Phi \) was learned by Algorithm 1. For real data, \( \Psi \) was learned in either an uncoupled way or a coupled way. In the uncoupled test, we first learned \( \Psi \) from a set of training data and then compute \( \Phi \) via Algorithm 1. In the coupled test, we simultaneously learned a pair of \( \Psi \) and \( \Phi \) in a way similar to \([7]\). Specifically, during each loop of the coupled algorithm, we first compute \( \Phi \) via Algorithm 1 with a fixed \( \Psi \), then update \( \Theta \) by solving
\[
\min_{\Theta} \|X - \Psi \Theta\|_F^2, \quad \text{subject to } \|\Theta\|_0 \leq S, \forall i \tag{4.1}
\]
for a given sparsity level \( S \), where \( \Theta = [\Theta_1, \ldots, \Theta_L] \) and \( \|\Theta_i\|_0 \) denotes the number of non-zeros of \( \Theta_i \). Then we learned \( \Phi \) by Algorithm 1. In the coupled test, we simultaneously learned a pair of \( \Psi \) and \( \Phi \) in a way similar to \([7]\). Specifically, during each loop of the coupled algorithm, we first compute \( \Phi \) via Algorithm 1 with a fixed \( \Psi \), then update \( \Theta \) by solving
\[
\min_{\Theta} \alpha \|X - \Psi \Theta\|_F^2 + \|\Phi X - \Phi \Psi \Theta\|_F^2, \quad \text{subject to } \|\Theta\|_0 \leq S, \forall i \tag{4.2}
\]
with respect to \( \Theta \), where \( \alpha > 0 \) is a weight parameter, and finally update \( \Psi \) and \( \Theta \) via solving (4.2) with respect to \( \Psi \) and the current nonzero entries of \( \Theta \) jointly.
### Table 5.1

<table>
<thead>
<tr>
<th>name</th>
<th>type</th>
<th>real/complex</th>
<th>kernel</th>
<th>downsampler</th>
<th>dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>rand-circ</td>
<td>circulant</td>
<td>complex</td>
<td>random</td>
<td>random</td>
<td>1D</td>
</tr>
<tr>
<td>real-rand-circ</td>
<td>circulant</td>
<td>real</td>
<td>random</td>
<td>random</td>
<td>1D</td>
</tr>
<tr>
<td>Gaussian</td>
<td>Gaussian</td>
<td>complex</td>
<td>———</td>
<td>random</td>
<td>1D</td>
</tr>
<tr>
<td>real-Gaussian</td>
<td>Gaussian</td>
<td>real</td>
<td>———</td>
<td>random</td>
<td>1D</td>
</tr>
<tr>
<td>opt-circ</td>
<td>circulant</td>
<td>complex</td>
<td>optimized</td>
<td>random</td>
<td>1D</td>
</tr>
<tr>
<td>real-opt-circ</td>
<td>circulant</td>
<td>real</td>
<td>optimized</td>
<td>random</td>
<td>1D</td>
</tr>
<tr>
<td>opt-circ-and-P</td>
<td>circulant</td>
<td>complex</td>
<td>optimized</td>
<td>optimized</td>
<td>1D</td>
</tr>
<tr>
<td>real-opt-circ-and-P</td>
<td>circulant</td>
<td>real</td>
<td>optimized</td>
<td>optimized</td>
<td>1D</td>
</tr>
<tr>
<td>opt-plus-rand-circ</td>
<td>circulant</td>
<td>complex</td>
<td>optimized+random</td>
<td>random</td>
<td>1D</td>
</tr>
<tr>
<td>coupled-plus-rand-circ</td>
<td>circulant</td>
<td>complex</td>
<td>optimized+random</td>
<td>random</td>
<td>1D</td>
</tr>
<tr>
<td>rand-2D-circ</td>
<td>circulant</td>
<td>complex</td>
<td>random</td>
<td>random</td>
<td>2D</td>
</tr>
<tr>
<td>opt-plus-rand-2D-circ</td>
<td>circulant</td>
<td>complex</td>
<td>optimized+random</td>
<td>random</td>
<td>2D</td>
</tr>
</tbody>
</table>

*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.

After obtaining $\Psi$ and $\Phi$, we used YALL1 [28] (version 1.4) to recover the sparse signal $\tilde{\theta}$ via solving

$$\min_{\theta} \|\theta\|_1 + \frac{1}{\rho} \|\Phi\Psi\theta - b\|_2^2, \quad (4.3)$$

where $b = \Phi\Psi\tilde{\theta} + \eta$ was the measurement contaminated by Gaussian noise $\eta \sim \mathcal{N}(0, \sigma I)$ and $\rho \geq 0$ was a parameter corresponding to $\sigma$. Throughout our tests, $\sigma$ was known, and the parameter $\rho$ was set to $\sigma$. If $\rho = 0$, problem (4.3) reduces to

$$\min_{\theta} \|\theta\|_1, \text{ subject to } \Phi\Psi\theta = b. \quad (4.4)$$

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 $10^4$.

### 5. Numerical Simulations.

We compared the performance of optimized circulant sensing matrices/operators to that of random ones on both synthetic data and real-world data. In addition, we tested a pair of circulant sensing matrix $\Phi$ and dictionary $\Psi$ learned together on real-world data. The tested sensing matrices and 2D operators are listed in Table 5.1. To be fair, except for the coupled learned circulant matrix $\text{coupled-plus-rand-circ}$, all sensing matrices or 2D operators were generated and tested with the same set of synthetic or learned dictionaries.

Throughout our tests, all columns of $\Psi$ and $\Phi$ were normalized to have the unit 2-norm. The normalization of $\Psi$ is only for convenience while that of $\Phi$ is critical, for otherwise the recovery by YALL1 or other solvers may become unstable. Similarly, we normalized all partial 2D circulant operators. We next introduce an efficient way to normalize 1D matrix $PC$ and 2D operator $P_0C_2$.

#### 5.1. Normalization of 1D matrix $PC$ and 2D operator $P_0C_2$.

Let $C$ be defined in (2.3) and $c = [v_1, v_n, v_{n-1}, \ldots, v_2]^T$ be its first column. It is easy to verify that the $j$th column of $C$ is $C_j = [c_{n-j+2}, c_{n-j+3}, \ldots, c_n, c_1, \ldots, c_{n-j+1}]^T$. Let $\ell = [\ell_1, \ldots, \ell_n]^T$ with $\ell_j$ being the Euclidean norm of the $j$th
column of $PC$. Recalling (3.7), we have

$$
\ell_j^2 = \|PC_j\|^2 = p_1c_{n-j+2}^2 + p_2c_{n-j+3}^2 + \ldots + p_{s}c_{n-j+1}^2 = \sum_{i=1}^{n} p_i c_{s_i j}^2, \text{ for } 1 \leq j \leq n,
$$

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

For $P\Omega C_2$, recall (2.5) and (3.15). Then normalizing $P\Omega C_2$ is equivalent to normalizing $D\Omega C_2$. Let $n = n_1 n_2$ and $\ell = [\ell_1, \ldots, \ell_n]^T$ with $\ell_j$ being the Euclidean norm of the $j$th column of $D\Omega C_2$. Note that for $1 \leq t, \tau \leq n_1$ and $1 \leq s, \kappa \leq n_2$,

$$(C_2)_{t,s,\tau,\kappa} = M_{t,s}^{\ell_j} = M_{\alpha_t, \beta_s}^{\ell_j},$$

where $i_t = t + (s-1)n_1$, $j_{\tau, \kappa} = \tau + (\kappa-1)n_1$ and $\alpha_{t, \tau} = \text{mod}(n_1 + \tau - t, n_1) + 1$, $\beta_{s, \kappa} = \text{mod}(n_2 + \kappa - s, n_2) + 1$. Hence, for each pair of $(\tau, \kappa)$, we can compute $\ell_j$ with $j = \tau + (\kappa-1)n_1$ via

$$
\ell_j = \sqrt{\sum_{t,s} (D\Omega)_{t,s,i_t,j_{\tau, \kappa}} M_{\alpha_{t, \tau}, \beta_{s, \kappa}}^2}.
$$

Therefore, we normalize $D\Omega C_2$ to $D\Omega C_2(\text{diag}(\ell))^{-1}$ and $P\Omega C_2$ to $P\Omega C_2 L$, where $L(X) = X \otimes L$ for any $X \in \mathbb{C}^{n_1 \times n_2}$ with $L \in \mathbb{C}^{n_1 \times n_2}$ obtained from $\ell$ via $\ell = \text{vec}(L)$.

5.2. Synthetic data. We tested different optimized circulant sensing matrices $\Phi$ on synthetic data along with two kinds of bases $\Psi$: 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 $512 \times 512$, and the sensing matrices had the size $64 \times 512$, 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 $5 \times 10^{-8}$. Figure 5.1 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 $\theta^*$ 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 5.1 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 performance of random circulant matrices can be found in [30]. 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.
Fig. 5.1. 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).

5.3. 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 of size $64 \times 256$ on real images. The dictionaries for all of them were learned from the same training set. The first three sensing matrices shared the same set of dictionaries learned by KSVD [1] with sparsity level $S = 6, 8$ in (4.1), and the last one was learned simultaneously with its corresponding dictionary by the coupled method described in Section 4 with sparsity level $S = 6, 8$ and $\alpha = \frac{1}{32}$ in (4.2). This choice of $\alpha$ was recommended in [7]. All images used in these tests were scaled to have the unit maximum pixel value. The training data consists of 20,000 $8 \times 8$ patches, that are 100 randomly extracted patches from each of the 200 images in the training set of the Berkeley segmentation dataset [17]. 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 5.2 depicts the mean squared error: $\text{MSE} = \frac{1}{\ell} \sum_{i=1}^{\ell} \| x^i - \Psi \theta^i \|_2^2 / (64\ell)$, for sparsity level $S = 6, 8$ and measurement size $m = 16, 24$. Here, $x^i$ is the vector obtained by reshaping the $i$th selected patch, $\ell = 600$ is the number of tested patches, and $\theta^i$ was the YALL1 solution of

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

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

---

2We 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.
Fig. 5.2. 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 $m = 16, 24$.

![Graphs showing MSE vs noise level for different sensing matrices and sampled row numbers](image)

(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.

All the four pictures in Figure 5.2 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 $481 \times 321$, and the second one had the resolution of $321 \times 481$. For convenience, we removed the last row and the last column of both the two images and partitioned each images into $1,200 \times 8 \times 8$ patches. Then we used YALL1 to solve (5.1) for each patch with the dictionary $\Psi$ 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 5.2 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 5.3. 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 those with the coupled learned circulant matrices. In addition, the coupled approach did slightly better than the uncoupled one.
<table>
<thead>
<tr>
<th>Castle</th>
<th>rand-circ</th>
<th>Gaussian</th>
<th>opt-plus-rand-circ</th>
<th>coupled-plus-rand-circ</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td>σ</td>
<td>PSNR</td>
<td>MSE</td>
<td>PSNR</td>
</tr>
<tr>
<td>16</td>
<td>0.00</td>
<td>19.28</td>
<td>4.81e-002</td>
<td>25.60</td>
</tr>
<tr>
<td>16</td>
<td>0.01</td>
<td>18.60</td>
<td>4.81e-002</td>
<td>24.39</td>
</tr>
<tr>
<td>16</td>
<td>0.05</td>
<td>18.48</td>
<td>4.54e-002</td>
<td>24.09</td>
</tr>
<tr>
<td>16</td>
<td>0.10</td>
<td>18.12</td>
<td>4.35e-002</td>
<td>23.17</td>
</tr>
<tr>
<td>16</td>
<td>0.15</td>
<td>17.59</td>
<td>4.29e-002</td>
<td>22.26</td>
</tr>
<tr>
<td>24</td>
<td>0.00</td>
<td>25.65</td>
<td>8.57e-003</td>
<td>28.76</td>
</tr>
<tr>
<td>24</td>
<td>0.01</td>
<td>24.25</td>
<td>7.08e-003</td>
<td>27.29</td>
</tr>
<tr>
<td>24</td>
<td>0.05</td>
<td>23.99</td>
<td>7.90e-003</td>
<td>26.63</td>
</tr>
<tr>
<td>24</td>
<td>0.10</td>
<td>23.37</td>
<td>9.12e-003</td>
<td>25.51</td>
</tr>
<tr>
<td>24</td>
<td>0.15</td>
<td>22.70</td>
<td>1.07e-002</td>
<td>24.41</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gulf</th>
<th>rand-circ</th>
<th>Gaussian</th>
<th>opt-plus-rand-circ</th>
<th>coupled-plus-rand-circ</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td>σ</td>
<td>PSNR</td>
<td>MSE</td>
<td>PSNR</td>
</tr>
<tr>
<td>16</td>
<td>0.00</td>
<td>19.53</td>
<td>4.28e-002</td>
<td>25.61</td>
</tr>
<tr>
<td>16</td>
<td>0.01</td>
<td>18.96</td>
<td>4.15e-002</td>
<td>24.56</td>
</tr>
<tr>
<td>16</td>
<td>0.05</td>
<td>18.81</td>
<td>3.93e-002</td>
<td>24.22</td>
</tr>
<tr>
<td>16</td>
<td>0.10</td>
<td>18.51</td>
<td>4.05e-002</td>
<td>23.49</td>
</tr>
<tr>
<td>16</td>
<td>0.15</td>
<td>18.13</td>
<td>4.20e-002</td>
<td>22.60</td>
</tr>
<tr>
<td>24</td>
<td>0.00</td>
<td>25.96</td>
<td>7.67e-003</td>
<td>29.06</td>
</tr>
<tr>
<td>24</td>
<td>0.01</td>
<td>24.60</td>
<td>6.69e-003</td>
<td>27.73</td>
</tr>
<tr>
<td>24</td>
<td>0.05</td>
<td>24.33</td>
<td>7.23e-003</td>
<td>27.05</td>
</tr>
<tr>
<td>24</td>
<td>0.10</td>
<td>23.63</td>
<td>8.46e-003</td>
<td>25.94</td>
</tr>
<tr>
<td>24</td>
<td>0.15</td>
<td>23.12</td>
<td>9.54e-003</td>
<td>24.85</td>
</tr>
</tbody>
</table>

5.4. Image tests with 2D circulant. For the tests in Section 5.3, 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 5.3 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 5.3 for this test. Instead of solving (5.1), now YALL1 was employed to solve

\[
\min_{\theta} \|\theta\|_1 + \frac{1}{\sigma} \|P_{\Omega}C_2LQ(\theta) - b\|_2^2, \tag{5.2}
\]

where \( P_{\Omega}C_2L \) denotes normalized partial 2D circulant operator, \( Q \) is defined in Section 3.2, \( b = P_{\Omega}C_2L(X^i) + \eta \) with the same \( \eta \) and \( \sigma \) to those in Section 5.3, and \( X^i \) is the \( i \)th tested patch corresponding to the reshaped vector \( x^i \) in (5.1). Figure 5.4 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 5.5 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 5.3 where circulant sensing matrices were compared.

6. 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. For image recovery to go from the patch scale to the image scale, one approach is to make an image-scale dictionary under which the underlying image is sparse. In this section, we introduce how to generate an image-scale dictionary based on patches, and we compare two
Fig. 5.3. 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 $m = 24$

<table>
<thead>
<tr>
<th>Sensing Matrix</th>
<th>PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>rand-circ</td>
<td>26.78</td>
</tr>
<tr>
<td>Gaussian</td>
<td>28.30</td>
</tr>
<tr>
<td>opt-plus-rand-circ</td>
<td>26.85</td>
</tr>
<tr>
<td>coupled-plus-rand-circ</td>
<td>27.86</td>
</tr>
</tbody>
</table>

different partial 2D circulant operators based on the generated image-scale dictionary.

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.

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 the dictionary $\Psi^0 = [\psi_1, \ldots, \psi_K]$. We want to recover $\hat{X}$ from the measurements $b = G(\hat{X}) + \eta$, where $G = P_0C_2L$ is a normalized partial 2D circulant operator defined on $\mathbb{C}^{N_1 \times N_2}$ and $\eta$ is Gaussian noise.

First, we construct an image-scale dictionary $\Psi$ from $\Psi^0$. 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 $N = n_r n_c + \text{sign}(s_r) n_c + \text{sign}(s_c) n_r + \text{sign}(s_r s_c)$. For each atom $\psi_k$, 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 $\Psi^k_t$
Fig. 5.4. 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

![MSE comparison graph]

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

Fig. 5.5. 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 \( \hat{\sigma} = 0.01 \)

![Recovered images]

rand-2D-circ, PSNR = 25.37
opt-plus-rand-2D-circ, PSNR = 27.70
rand-2D-circ, PSNR = 25.86
opt-plus-rand-2D-circ, PSNR = 27.96

be the matrix with all elements being zero except having \( \psi_k \) 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 \( \Psi_f^k \) be the matrix with all elements being zero except having the last \( s_r \) rows of \( \psi_k \) on the submatrix indexed by the \((n_r n_1 + 1)\)th through \( N_1 \)th 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 \( \psi_k \). If \( s_c s_r \neq 0 \), we form the last atom \( \Psi_N^k \) with all elements being zero except having the right-bottom \( s_r \times s_c \) submatrix of \( \psi_k \) on its right-bottom \( s_r \times s_c \) submatrix. Note that any two atoms generated from \( \psi_k \) have no common non-zero parts.

The image \( \hat{X} \) has a sparse representation under \( \Psi \) since each patch of \( \hat{X} \) is sparse under \( \Psi^0 \). Figure 6 illustrates how we form \( N \) image-scale atoms \( \Psi_1, \Psi_2, \ldots, \Psi_N \) from a small atom \( \psi \).

Then, we can recover \( \hat{X} \) from the measurements \( b = \mathcal{G}(\hat{X}) + \eta \) by solving

\[
\min_{\theta} \|\theta\|_1 + \frac{1}{\sigma} \|\mathcal{G}(\theta) - b\|_2^2,
\]

(6.1)
where $Q$ is a 2D operator defined in the same way as in Section 3.2, $\eta \sim \mathcal{N}(0, \sigma I)$ is Gaussian noise and $\sigma = \hat{\sigma} \|G(\hat{X})\|_\infty / \|\eta\|_\infty$. At first glance, the recovery by solving (6.1) may take a lot of time since the dictionary $\Psi$ has so many atoms. However, the structure of these atoms enables even faster recovery than that using patch-size dictionaries.

We compared rand-2D-circ and opt-plus-rand-2D-circ on two images. Both of them had the resolution of $241 \times 129$, and they were obtained by cropping 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 6.1 lists the average results of 20 independent trials, and Figure 6.2 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 optimized circulant operators did significantly better than the random ones on average.

**Acknowledgements.** The authors thank Prof. Guillermo Sapiro for his encouragement and very valuable suggestions. The authors also thank Dr. Ming Yan for his comments on complex-valued random sensing matrices. The authors’ work has been supported in part by AFOSR, ARL and ARO, DARPA, NSF, and ONR.

---

\textsuperscript{3}The optimized 2D circulant operator was obtained by the method discussed in Section 3.2, and it took us quite a long time. The learning on larger images would take much more time. To save time, we simply used the cropped images to test our idea.
Table 6.1

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

<table>
<thead>
<tr>
<th>Plane</th>
<th>rand-2D-circ</th>
<th>opt-plus-rand-2D-circ</th>
<th>Plane</th>
<th>rand-2D-circ</th>
<th>opt-plus-rand-2D-circ</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR</td>
<td>MSE</td>
<td>PSNR</td>
<td>MSE</td>
<td>PSNR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PSNR</td>
<td>MSE</td>
<td></td>
</tr>
<tr>
<td>SR</td>
<td>σ</td>
<td></td>
<td>SR</td>
<td>σ</td>
<td></td>
</tr>
<tr>
<td>0.20</td>
<td>0.00</td>
<td>21.16 4.58e-002</td>
<td>26.68</td>
<td>2.18e-003</td>
<td>0.30 0.00</td>
</tr>
<tr>
<td>0.20</td>
<td>0.01</td>
<td>21.35 4.30e-002</td>
<td>26.60</td>
<td>2.21e-003</td>
<td>0.30 0.01</td>
</tr>
<tr>
<td>0.20</td>
<td>0.05</td>
<td>20.62 4.27e-002</td>
<td>25.16</td>
<td>3.07e-003</td>
<td>0.30 0.05</td>
</tr>
<tr>
<td>0.20</td>
<td>0.10</td>
<td>19.32 4.58e-002</td>
<td>23.13</td>
<td>4.92e-003</td>
<td>0.30 0.10</td>
</tr>
<tr>
<td>0.20</td>
<td>0.15</td>
<td>18.15 4.96e-002</td>
<td>21.56</td>
<td>7.06e-003</td>
<td>0.30 0.15</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gulf</th>
<th>rand-2D-circ</th>
<th>opt-plus-rand-2D-circ</th>
<th>Gulf</th>
<th>rand-2D-circ</th>
<th>opt-plus-rand-2D-circ</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR</td>
<td>MSE</td>
<td>PSNR</td>
<td>MSE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PSNR</td>
<td>MSE</td>
<td></td>
</tr>
<tr>
<td>SR</td>
<td>σ</td>
<td></td>
<td>SR</td>
<td>σ</td>
<td></td>
</tr>
<tr>
<td>0.20</td>
<td>0.00</td>
<td>20.60 1.92e-002</td>
<td>24.50</td>
<td>3.57e-003</td>
<td>0.30 0.00</td>
</tr>
<tr>
<td>0.20</td>
<td>0.01</td>
<td>20.64 1.99e-002</td>
<td>24.42</td>
<td>3.63e-003</td>
<td>0.30 0.01</td>
</tr>
<tr>
<td>0.20</td>
<td>0.05</td>
<td>20.43 1.93e-002</td>
<td>24.05</td>
<td>3.95e-003</td>
<td>0.30 0.05</td>
</tr>
<tr>
<td>0.20</td>
<td>0.10</td>
<td>19.85 2.05e-002</td>
<td>23.20</td>
<td>4.80e-003</td>
<td>0.30 0.10</td>
</tr>
<tr>
<td>0.20</td>
<td>0.15</td>
<td>19.22 2.20e-002</td>
<td>22.32</td>
<td>5.89e-003</td>
<td>0.30 0.15</td>
</tr>
</tbody>
</table>

REFERENCES


