PAPER

Cross Low-Dimension Pursuit for Sparse Signal Recovery from Incomplete Measurements Based on Permuted Block Diagonal Matrix

Zaixing HE[†a)], ***Student Member***, **Takahiro OGAWA**[†b)], ***and*** **Miki HASEYAMA**[†c)], ***Members***

SUMMARY In this paper, a novel algorithm, Cross Low-dimension Pursuit, based on a new structured sparse matrix, Permuted Block Diagonal (PBD) matrix, is proposed in order to recover sparse signals from incomplete linear measurements. The main idea of the proposed method is using the PBD matrix to convert a high-dimension sparse recovery problem into two (or more) groups of highly low-dimension problems and crossly recover the entries of the original signal from them in an iterative way. By sampling a sufficiently sparse signal with a PBD matrix, the proposed algorithm can recover it efficiently. It has the following advantages over conventional algorithms: (1) low complexity, i.e., the algorithm has linear complexity, which is much lower than that of existing algorithms including greedy algorithms such as Orthogonal Matching Pursuit and (2) high recovery ability, i.e., the proposed algorithm can recover much less sparse signals than even ℓ^1-norm minimization algorithms. Moreover, we demonstrate both theoretically and empirically that the proposed algorithm can reliably recover a sparse signal from highly incomplete measurements.

key words: *sparse recovery, sparsest solution, compressed sensing, permuted block diagonal matrix, greedy algorithms, orthogonal matching pursuit, ℓ^1-norm minimization, basis pursuit*

1. Introduction

Recovering a sparse signal from incomplete linear measurements is of significant importance. It is used in many areas such as compressed sensing [1], [2], decoding real codes [3], sparse representation [4], and data stream computing [5]. A way for recovering a signal that is sufficiently sparse is to find the sparsest solution to a system of under-determined equations, and it is known as an NP-hard problem. However, recent studies have shown that it can be solved in reasonable time when the signal is highly sparse. In the literature, most attention has been focused on designing good measurement matrices and researching efficient and powerful recovery algorithms.

Many researchers utilize matrices with i.i.d. elements such as Gaussian matrices as measurement matrices [2], [6]–[8], since these matrices provide high recovery ability. However, the heavy computational burdens and huge storage requirement make them impractical. In order to reduce the computational time and storage, Candès et al. use the Partial Fourier (PF) matrix to measure signals [1]. Matrices generated from other orthogonal transform matrices, e.g., the Partial Hadamard matrix [9], are also efficient.

Based on such matrices, greedy algorithms [7], [8], [10]–[14], e.g., Orthogonal Matching Pursuit (OMP) [7], [11], and ℓ^1-norm minimization algorithms [15]–[18], which solve a Basis Pursuit (BP) problem [15], can successfully recover sparse signals. BP algorithms are well-known for their high recovery ability. On the other hand, OMP is well-known for its low complexity. However, as the technique advances, higher demands and larger scale applications make both of them not satisfactory. Thus, many new algorithms have been proposed to improve recovery ability and speed [8], [12]–[14], [17]–[19]. For example, Subspace Pursuit (SP) [14], which is a variation of OMP, has high recovery ability close to that of BP algorithms, while its speed is much faster than that of OMP. Unfortunately, the improvements are still not satisfactory. Although some algorithms achieved higher recovery ability than that of BP algorithms in many situations [19]–[22], their complexity is still high, higher than that of greedy algorithms. For example, it has been reported that an Iteratively Reweighted Least Squares (IRLS) algorithm can recover a less sparse (Gaussian random) signal than can BP algorithms [19]. However, its speed is still slow. On the other hand, although some of the algorithms achieved lower complexity than that of OMP, their recovery ability is low, at least lower than or comparable to that of BP algorithms. Therefore, a new algorithm with both high recovery ability and low complexity is strongly desired.

In order to address this issue, we propose a novel algorithm, Cross Low-dimension Pursuit (CLP), for the sparse recovery problem based on a new sparse matrix, Permuted Block Diagonal (PBD) matrix. Unlike conventional methods pursuing the sparsest solution in high dimensions, the main idea of our method is to convert a high-dimension sparse recovery problem into two (or more) groups of low-dimension problems. Then we recover entries of the original signal from these small systems of under-determined equations. The proposed algorithm outperforms existing algorithms in both recovery ability and complexity. The proposed algorithm has (1) high recovery ability, i.e., it can recover much less sparse signals than can existing algorithms and (2) low complexity, i.e., it has linear complexity, which is much lower than that of existing algorithms.

Manuscript received December 27, 2010.
Manuscript revised April 19, 2011.

[†]The authors are with the Graduate School of Information Science and Technology, Hokkaido University, Sapporo-shi, 060-0814 Japan.
a) E-mail: ka@lmd.ist.hokudai.ac.jp
b) E-mail: ogawa@lmd.ist.hokudai.ac.jp
c) E-mail: miki@ist.hokudai.ac.jp

DOI: 10.1587/transfun.E94.A.1793

Our algorithm is aimed at exact sparse signal recovery based on a specifically structured measurement matrix. Many classical algorithms do not rely on any specified matrix and can exactly recover the sparse signal when there is no noise and approximate the target signal when there is small noise. Compared with such algorithms, the application areas of our algorithm are limited. Our algorithm is suitable for applications, where exact recovery is needed and choices of measurement matrices are available. Nevertheless, the proposed algorithm still makes significant sense since such (or potential) applications are broad. Here, we just give three examples of such application areas, where the classical sparse recovery algorithms are used or can be used. The first area is decoding real codes [3]. When transmitting a real-valued message by encoding it with a full *rank* matrix (with more rows than columns) and there is impulsive noise over the communication channel, it is shown in [3] that to exactly decode such codes, it is sufficient to exactly recover the sparse error vector from an under-determined system. Consider constructing a PBD matrix as a parity-check matrix and generating a coding matrix whose columns span the null space of the PBD matrix, e.g., using the QR decomposition. A real-valued message is encoded by this coding matrix and a fraction of the entries are corrupted during transmission. Then at the receiver side, the PBD matrix is applied to the corrupted output such that the CLP algorithm can be used to solve a sparse recovery problem for reconstructing the sparse error vector and consequently the original message (refer to [3] for more details of decoding real codes by solving a sparse recovery problem). Note that unlike usual real codes based on sparse recovery, for a PBD-CLP code, a parity-check matrix is constructed firstly and a coding matrix is generated consequently. In [23], the authors pointed out another potential application area, cryptography, where the above model of decoding real codes can be used. In such a cryptographic scheme, a secret message (the plaintext) is encrypted by multiplying it with a coding matrix whose columns span the null space of a PBD matrix and corrupting a fraction of entries artificially. With knowledge of the PBD matrix, the plaintext can be finally recovered from the ciphertext by the CLP algorithm in the same way as the above application of decoding real codes. The third example is data stream computing [5]. In this area, the vectors are usually very large and sparse. It is preferred to maintain a low-dimensional vector by multiplying it with a (sparse) matrix and exactly reconstruct it when needed [24]. Thus, the PBD matrix is suitable for this application since it is highly sparse. Furthermore, sparse vectors can be reconstructed efficiently by solving a sparse recovery problem with the CLP algorithm. In all of these areas, the two advantages of the proposed algorithm are of significant importance.

The remainder of this paper is organized as follows. In Sect. 2, the notations used throughout this paper are defined. The CLP algorithm is proposed in Sect. 3. In Sect. 4, results of analysis of effectiveness and complexity are presented to clarify the high reliability and low complexity of the proposed algorithm. In Sect. 5, experimental results are shown to verify the improvement of the proposed method, and conclusions and future work are presented in Sect. 6.

2. Notations

The following notations are used throughout this paper.

$\lfloor r \rfloor$ **:** Given a real number r, $\lfloor r \rfloor$ is the maximum integer which is no larger than r.

$\overline{I}$ **:** Given a set I, $\overline{I}$ is referred to as the *complement* of I.

$|I|$ **:** $|I|$ denotes the number of the elements in I.

$r(\mathbf{W})$ **:** Given a matrix $\mathbf{W}$, $r(\mathbf{W})$ denotes its *rank*.

$\mathbf{W}^*$ **:** $\mathbf{W}^*$ is the conjugate transpose of matrix $\mathbf{W}$.

$\mathbf{w}_I$ **:** Given an index set I, and a matrix $\mathbf{w}$, $\mathbf{w}_I$ denotes the submatrix of $\mathbf{w}$ consisting of the columns with coordinates in I; or given the index set I and a vector $\mathbf{w}$, $\mathbf{w}_I$ denotes the subvector of $\mathbf{w}$ consisting of the elements with coordinates in I[†].

$\mathbf{w} \to \mathbf{p}$ **:** Given a permutation $\mathbf{p}$ of set $\{1, 2, \cdots, M\}$ and a matrix $\mathbf{w} \in \mathbb{R}^{k \times M}$, $\mathbf{w} \to \mathbf{p}$ denotes permuting the columns of $\mathbf{w}$ by $\mathbf{p}$; or given the permutation $\mathbf{p}$ and a vector $\mathbf{w} \in \mathbb{R}^M$, $\mathbf{w} \to \mathbf{p}$ denotes permuting the elements of $\mathbf{w}$ by $\mathbf{p}$[††].

$\|\mathbf{y}\|_0$ **:** $\|\mathbf{y}\|_0$ denotes the ℓ^0-norm of $\mathbf{y}$, counting the number of nonzero entries of $\mathbf{y}$.

$\binom{n}{k}$ **:** $\binom{n}{k}$ denotes the number of k-combinations of a set that has n elements.

3. CLP Algorithm Based on a PBD Matrix

The sparse recovery problem can be described as follows. Suppose a sparse signal $\mathbf{y} \in \mathbb{R}^M$ with $\|\mathbf{y}\|_0 \leq T$ (called T-sparse) is sampled by a measurement matrix $\mathbf{D} \in \mathbb{R}^{N \times M}$ ($N < M$), obtaining a measurement vector $\mathbf{s}$ ($\mathbf{s} = \mathbf{D}\mathbf{y}$). The target is to recover $\mathbf{y}$ from $\mathbf{s}$ and $\mathbf{D}$. A standard approach to recover $\mathbf{y}$ is to find the sparsest solution to the under-determined equations,

$$(P_0) \ : \ \min \|\mathbf{z}\|_0 \quad \text{subject to} \quad \mathbf{D}\mathbf{z} = \mathbf{s}. \tag{1}$$

When $\mathbf{y}$ is sufficiently sparse, with a matrix $\mathbf{D}$, which satisfies some suitable conditions, e.g., restricted isometry property (RIP) [3], the solution to (P_0) equals $\mathbf{y}$.

Designing measurement matrices is critical for solving

[†]For example, let $\mathbf{w}$ = [3.5 4.3 7.0 5.2] and $I = \{2, 4\}$. Then $\mathbf{w}_I$ = [4.3 5.2].

[††]For example, let $\mathbf{w}$ = [3.5 4.3 7.0 5.2] and $\mathbf{p}$ = [2 4 3 1]. Then $\mathbf{v} = \mathbf{w} \to \mathbf{p}$ = [4.3 5.2 7.0 3.5].

(P_0), since both the performance and speed of the recovery algorithms depend on them. In this section, we construct a new structured measurement matrix, PBD matrix, in 3.1. Then based on this matrix, a new algorithm, CLP, is proposed in 3.2.

3.1 Permuted Block Diagonal Matrix

Suppose that we have L matrices that are block diagonal, $\mathbf{W}_1 \in \mathbb{R}^{N_1 \times M} = diag(\mathbf{w}_1, \cdots, \mathbf{w}_1)$, $\cdots$, $\mathbf{W}_L \in \mathbb{R}^{N_L \times M} = diag(\mathbf{w}_L, \cdots, \mathbf{w}_L)$, where $N_1 + \cdots + N_L = N$ and $\mathbf{w}_1, \cdots, \mathbf{w}_L \in \mathbb{R}^{n \times m}$, and L different random permutations of set $\{1, \cdots, M\}$, namely $\mathbf{p}_1, \cdots, \mathbf{p}_L$. We construct a PBD matrix $\mathbf{D} \in \mathbb{R}^{N \times M}$ by permuting the matrices with these permutations independently. Mathematically,

$$\mathbf{D} := \begin{bmatrix} \mathbf{W}_1 \to \mathbf{p}_1 \\ \vdots \\ \mathbf{W}_L \to \mathbf{p}_L \end{bmatrix} = \begin{bmatrix} \begin{pmatrix} \mathbf{w}_1 & & \\ & \ddots & \\ & & \mathbf{w}_1 \end{pmatrix} \to \mathbf{p}_1 \\ \vdots \\ \begin{pmatrix} \mathbf{w}_L & & \\ & \ddots & \\ & & \mathbf{w}_L \end{pmatrix} \to \mathbf{p}_L \end{bmatrix} \tag{2}$$

The random permutations are often used for constructing measurement matrices to provide high-quality reconstructions, e.g., the scrambled Fourier matrix for compressed sensing of natural signals [25]. The blocks $\mathbf{w}_1, \cdots, \mathbf{w}_L$ are the following defined *full spark* matrices.

Definition 1. (spark and full spark) *Let* $\mathbf{w}$ *be a matrix* $\in \mathbb{R}^{n \times m}$. *The spark of* $\mathbf{w}$*, denoted by* $s(\mathbf{w})$*, is defined as the minimum number of columns that are linearly dependent in* $\mathbf{w}$ [4]; *if* $s(\mathbf{w}) > \min(n, m)$*,* $\mathbf{w}$ *is defined as a full spark matrix.*

Note that a full *spark* matrix $\mathbf{w}$ has full *rank* and $s(\mathbf{w}) = r(\mathbf{w}) + 1$. Although construction of a full *spark* matrix in a high dimension is very difficult, it is easy to construct one in a highly low dimension. For example, one can generate a (Gaussian) random matrix in a low dimension and examine whether it has full *spark*. Otherwise, repeat this procedure. To simplify the explanation, we set $L = 2$ for the proposed algorithm in this paper. Furthermore, in order to reduce the complexity, we suggest that n should be a small even number like two or four[†]. Compared with conventional measurement matrices, PBD matrices have important advantages such as easy construction and fast matrix-vector multiplication, which are helpful in sensing and recovery, respectively. The resource for constructing a PBD matrix is only those for generating two (or several) low-dimension blocks and permutations. Furthermore, since the PBD matrix is highly sparse, the corresponding matrix-vector multiplication is fast.

In the literature of compressed sensing, which is closely related to the sparse recovery problem, some researchers have proposed block diagonal matrices and corresponding algorithms for sensing and reconstructing natural signals [26], [27]. However, it should be noted that these frameworks are different from ours: we use a matrix generated from block diagonal matrices to measure a sparse signal directly, but these frameworks do not. In these frameworks, block diagonal matrices are utilized to measure a signal that is usually not sparse but can be represented as a sparse vector in some domain, e.g., wavelet coefficients. However, for the sparse vector, the measurement matrix to sample it is actually the matrix that is generated by multiplying the corresponding transform matrix with the block diagonal matrix. Therefore, the actual measurement matrix for sampling the sparse vector is not block diagonal.

3.2 Cross Low-Dimension Pursuit

In this subsection, we present the proposed algorithm. With the PBD matrix, the system of under-determined equations in Eq. (1), $\mathbf{Dz} = \mathbf{s}$, can be divided into two subsystems of equations. Furthermore, in each sub-system, the corresponding matrix is block diagonal. Obviously, solving a system of equations corresponding to a block diagonal matrix is the same as solving small systems of equations corresponding to the blocks independently. Thus, the high-dimension problem has been converted into two independent groups of quite low-dimension problems corresponding to the blocks. Moreover, although there may exist some unsolvable low-dimension problems in each subsystem of under-determined equations, they may become solvable when substituting the solutions from the other subsystem.

Based on the basic idea presented above, the proposed algorithm consists of two stages. **The first stage** is crossly solving the two sub-systems of under-determined equations with block diagonal matrices. Each sub-system of under-determined equations is solved by solving the low-dimension problems corresponding to the blocks. **The next stage** is to solve the residual equation (if there is one) after the first stage.

This subsection is organized as follows. In 3.2.1, we present a method for solving low-dimension problems in a sub-system of under-determined equations with a block diagonal matrix. In 3.2.2, we present a cross procedure, which crossly solves the two sub-systems of under-determined equations of the first stage. We then present a method for solving the residual equation of the second stage and detailed description of the proposed algorithm in 3.2.3.

3.2.1 Low-Dimension Recovery

Assume that $\mathbf{W}$ is a block diagonal matrix, $\mathbf{W} = diag(\mathbf{w}_1, \cdots, \mathbf{w}_K)$, and all of the blocks have n rows and

[†]The reason why n should be even is that the method that we used to find the sparsest solutions (in a low dimension) corresponding to the blocks $\mathbf{w}_k$ can recover a $\lfloor \frac{n}{2} \rfloor$-sparse signal ($n < m$). When $\frac{n}{2}$ is an integer, it performs best. See 3.2.1 for details.

have full *spark*. We aim to recover a sparse signal $\mathbf{y}$ from incomplete measurements,

$$\mathbf{Wz} = \mathbf{s}, \tag{3}$$

where $\mathbf{s} = \mathbf{Wy}$. Obviously, solving Eq. (3) is the same as solving the following low-dimension equations separately,

$$\begin{aligned} &\mathbf{w}_1\mathbf{z}_1 = \mathbf{s}_1, \\ &\quad\vdots \\ &\mathbf{w}_K\mathbf{z}_K = \mathbf{s}_K, \end{aligned} \tag{4}$$

where $\mathbf{s}_k$ is the kth segment of $\mathbf{s}$ corresponding to $\mathbf{w}_k$ ($k \in \{1, \cdots, K\}$). The kth segment of $\mathbf{y}$, $\mathbf{y}_k$, can be recovered by solving $\mathbf{w}_k\mathbf{z}_k = \mathbf{s}_k$ in the following situations.

1. $\mathbf{w}_k$ has no more than n columns. In this case, $\mathbf{w}_k$ has full column *rank* since it has full *spark*. Thus, these low-dimension equations are determined. The solution is $\hat{\mathbf{y}}_k = \mathbf{w}_k^{-1}\mathbf{s}_k$ when $\mathbf{w}_k$ is square or $\hat{\mathbf{y}}_k = (\mathbf{w}_k^*\mathbf{w}_k)^{-1}\mathbf{w}_k^*\mathbf{s}_k$ when $\mathbf{w}_k$ is not square[†].
2. $\mathbf{w}_k$ has more than n columns and $\mathbf{y}_k$ is $\lfloor\frac{n}{2}\rfloor$-sparse. In this case, we can recover $\mathbf{y}_k$ by solving a small (P_0) as follows:

$$(mP_0): \quad \min \|\mathbf{z}_k\|_0 \quad \text{s.t.} \quad \mathbf{w}_k\mathbf{z}_k = \mathbf{s}_k. \tag{5}$$

A natural direct way to solve (mP_0) is exhaustive searches over all subsets of $\lfloor\frac{n}{2}\rfloor$ columns of $\mathbf{w}_k$. Let I be a subset of $\{1, \cdots, m\}$ (m being the number of columns in $\mathbf{w}_k$), and $|I| = \lfloor\frac{n}{2}\rfloor$. We can solve a Least Squares (LS) problem of the following equation:

$$(\mathbf{w}_k)_I\mathbf{z}_k = \mathbf{s}_k. \tag{6}$$

If the solution $\hat{\mathbf{z}}_k$ ($\hat{\mathbf{z}}_k = ((\mathbf{w}_k)_I^*(\mathbf{w}_k)_I)^{-1}(\mathbf{w}_k)_I^*\mathbf{s}_k$) satisfies $(\mathbf{w}_k)_I\hat{\mathbf{z}}_k = \mathbf{s}_k$, then $\mathbf{y}_k$ is recovered by setting $(\hat{\mathbf{y}}_k)_I = \hat{\mathbf{z}}_k$ and $(\hat{\mathbf{y}}_k)_{\overline{I}} = \mathbf{0}$. We call it Exhaustive Subset Searching (ESS). Although there are $\binom{m}{\lfloor\frac{n}{2}\rfloor}$ such subsets to be searched and the ESS method has exponential complexity, solving (mP_0) takes little computation due to the low dimension.

Theorem 1. *Let $\mathbf{w}_k \in \mathbb{R}^{n\times m}$ ($n < m$) be a full spark matrix. The ESS method exactly recovers $\mathbf{y}_k$ from $\mathbf{w}_k\mathbf{z}_k = \mathbf{s}_k$ whenever $\|\mathbf{y}_k\|_0 \le \lfloor\frac{n}{2}\rfloor$.*

Actually, ESS finds a solution whose ℓ^0-norm is no larger than $\lfloor\frac{n}{2}\rfloor$. Therefore, to prove Theorem 1 is the same as to prove that $\mathbf{y}_k$ is the unique solution with ℓ^0-norm no larger than $\lfloor\frac{n}{2}\rfloor$, which is also equivalent to proving the following lemma.

Lemma 1. *Let $\|\mathbf{y}_k\|_0 = t$ ($t \le \lfloor\frac{n}{2}\rfloor$), and let $\hat{\mathbf{y}}_k$ be another solution to $\mathbf{w}_k\mathbf{z}_k = \mathbf{s}_k$. Then $\|\hat{\mathbf{y}}_k\|_0 > \lfloor\frac{n}{2}\rfloor$.*

Proof. Suppose I ($|I| = t$) and $\hat{I}$ are the index sets of nonzero entries of $\mathbf{y}_k$ and $\hat{\mathbf{y}}_k$, respectively. Since $\mathbf{w}_k$ is a *full spark* matrix, $s(\mathbf{w}_k) = n + 1$. Therefore, $t \le \lfloor\frac{n}{2}\rfloor < \frac{s(\mathbf{w}_k)}{2}$. We prove that $|\hat{I}| > \frac{s(\mathbf{w}_k)}{2}$ ($> \lfloor\frac{n}{2}\rfloor$).

Since $\mathbf{s}_k = \mathbf{w}_k\mathbf{y}_k = \mathbf{w}_k\hat{\mathbf{y}}_k$, we have $\mathbf{w}_k(\mathbf{y}_k - \hat{\mathbf{y}}_k) = 0$. $\mathbf{y}_k - \hat{\mathbf{y}}_k$ is supported on $I \cup \hat{I}$. Obviously, $|I \cup \hat{I}| \ge s(\mathbf{w}_k)$. On the other hand, $|I \cup \hat{I}| \le |I| + |\hat{I}|$. Thus, $|\hat{I}| \ge |I \cup \hat{I}| - |I| \ge s(\mathbf{w}_k) - t > \frac{s(\mathbf{w}_k)}{2}$. □

When $\mathbf{w}_k$ has more than n columns and $\mathbf{y}_k$ is not $\lfloor\frac{n}{2}\rfloor$-sparse, there is no hope of recovering $\mathbf{y}_k$ by the ESS method. However, our proposed algorithm solves the equations that have not been solved yet in an iterative way as will be shown in the following cross procedure.

3.2.2 Cross Procedure

As shown above, a single block diagonal matrix converts a large-scale sparse recovery problem into a group of low-dimension problems. Moghadam et al. take benefits of this just as we do [28]. However, when some of the low-dimension equations are unsolvable, the original sparse signal cannot be entirely recovered. In our method, since the PBD matrix is created from two block diagonal matrices, there are two different groups of low-dimension problems constructed. Thus, the unsolvable equations may become solvable when substituting the solutions from the other group. Moreover, this can be done crossly in an iterative way until there is no new entry recovered or all of the entries are recovered. At each iteration, we use the low-dimension recovery presented above to recover part of the entries from these two groups of equations. Now we consider the cross procedure with a PBD matrix. First of all, we give the definition of *inverse permutation*.

Definition 2. (inverse permutation) *Let $\mathbf{p}$ be a permutation of set $\{1, \cdots, M\}$. $\mathbf{p}^{-1}$ is defined as its inverse permutation such that $\forall j = \mathbf{p}(i)$, $\mathbf{p}^{-1}(j) = i$.*

From the structure of a PBD matrix, we can observe that the use of $\mathbf{D}$ to measure $\mathbf{y}$ can be considered as the following procedure. The signal is permuted by $\mathbf{p}_1^{-1}$ and $\mathbf{p}_2^{-1}$ independently and then measured by $\mathbf{W}_1$ and $\mathbf{W}_2$, respectively, as follows:

$$\mathbf{Dy} = \begin{bmatrix} (\mathbf{W}_1 \to \mathbf{p}_1)\mathbf{y} \\ (\mathbf{W}_2 \to \mathbf{p}_2)\mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{W}_1(\mathbf{y} \to \mathbf{p}_1^{-1}) \\ \mathbf{W}_2(\mathbf{y} \to \mathbf{p}_2^{-1}) \end{bmatrix} = \mathbf{s}. \tag{7}$$

Let $\mathbf{s}_1$ and $\mathbf{s}_2$ denote the first and second halves of $\mathbf{s}$, and initialize $\mathbf{D}^0 = \mathbf{D}$, $\mathbf{s}^0 = \mathbf{s}$, $\mathbf{W}_1^0 = \mathbf{W}_1$, $\mathbf{W}_2^0 = \mathbf{W}_2$. The first iteration is to recover entries from the following equation:

$$\mathbf{D}^0\mathbf{z} = \mathbf{s}^0. \tag{8}$$

It is the same as recovering them from the following equations:

$$\mathbf{W}_1^0\mathbf{z}_1 = \mathbf{s}_1^0, \tag{9}$$

[†]In the whole procedure of the CLP algorithm as will be shown in the following subsections, this situation does not occur at the first iteration since the original blocks are overcomplete (more columns than rows). However, it may occur at the next iterations when substituting some solutions from the other systems.

$$\mathbf{W}_2^0\mathbf{z}_2 = \mathbf{s}_2^0. \tag{10}$$

Since $\mathbf{W}_1^0$ and $\mathbf{W}_2^0$ are block diagonal matrices, we can use the low-dimension recovery to recover entries from both Eqs. (9) and (10) separately. Let $\hat{\mathbf{z}}$ and I be the vector of recovered entries of $\mathbf{y}$ and its index set, respectively. The residual equation, which is to be solved at the second iteration, is

$$\mathbf{D}^1\mathbf{z} = \mathbf{s}^1, \tag{11}$$

where $\mathbf{D}^1 = \mathbf{D}_{\overline{I}}$ and $\mathbf{s}^1 = \mathbf{s}^0 - \mathbf{D}_I\hat{\mathbf{z}}$. Obviously, more entries can be recovered from Eq. (11) in the same way as solving Eq. (8) at the first iteration. More generally, at the kth iteration, we recover entries from the following residual equation:

$$\mathbf{D}^{k-1}\mathbf{z} = \mathbf{s}^{k-1}. \tag{12}$$

This iteration stage can go on until there is no new entry recovered or all of the entries are recovered.

3.2.3 Solving the Last Residual Equation

The cross procedure can reliably recover a sufficiently sparse signal. When the signal is too dense to be recovered entirely in the cross procedure, there is still a way to recover the remaining entries. In such a case, we solve the Last Residual Equation (LRE). Suppose the number of total iterations in the cross procedure above is $\tilde{k}$. Then we have the following LRE:

$$\mathbf{D}^{\tilde{k}}\mathbf{z} = \mathbf{s}^{\tilde{k}}. \tag{13}$$

We simply apply the pseudoinversion process to Eq. (13) to recover the vector of remaining entries, $\hat{\mathbf{z}} = ((\mathbf{D}^{\tilde{k}})^*\mathbf{D}^{\tilde{k}})^{-1}(\mathbf{D}^{\tilde{k}})^*\mathbf{s}^{\tilde{k}}$. Obviously, when $\mathbf{D}^{\tilde{k}}$ has full column *rank*, Eq. (13) is determined, and all of the remaining entries have been recovered.

In conclusion, the proposed method pursues the sparsest solution in quite low dimensions and crossly solves two systems of equations. We name it Cross Low-dimension Pursuit. Figure 1 shows the whole procedure of the proposed algorithm, where $\hat{\mathbf{z}}$ denotes the vector of newly recovered entries at each iteration, I^k denotes the index set of $\hat{\mathbf{z}}$ at the kth iteration, I_{un}^k and I_{ed}^k denote the index sets of unrecovered and recovered entries after the kth iteration, respectively, $\mathbf{D}^k$ is the the submatrix of $\mathbf{D}$ consisting of the columns in I_{un}^k, $\mathbf{s}^k$ is the residual vector of $\mathbf{s}$ after the kth iteration, and $\tilde{k}$ is the total iteration number. Figure 2 shows a simple example of sparse signal recovery by CLP, where the original signal is a 1024-length sparse signal with 89 randomly located nonzeros.

4. Effectiveness and Complexity Analysis

In this section, we analyze the high reliability and low complexity of the proposed algorithm by effectiveness analysis and complexity analysis, respectively.

Input: A matrix $\mathbf{D}$ and a vector $\mathbf{s}$
Output: A sparse recovered vector $\hat{\mathbf{y}}$
Initialize: Set $\hat{\mathbf{y}} = \mathbf{0}$, $I_{ed}^0 = \emptyset$, $I_{un}^0 = \{1, \cdots, M\}$, $\mathbf{s}^0 = \mathbf{s}$, $\mathbf{D}^0 = \mathbf{D}$, and the counter $k = 1$.
Iteration:

1. Recover entries from $\mathbf{D}^{k-1}\mathbf{z} = \mathbf{s}^{k-1}$ by the cross recovery procedure presented in 3.2.2, obtaining newly recovered entries vector $\hat{\mathbf{z}}$ and index set I^k.
2. Update results. $\hat{\mathbf{y}}_{I^k} = \hat{\mathbf{z}}$, $I_{ed}^k = I_{ed}^{k-1} \cup I^k$, $I_{un}^k = \overline{I_{ed}^k}$.
3. Update residuals. $\mathbf{s}^k = \mathbf{s}^{k-1} - \mathbf{D}_{I^k}\hat{\mathbf{z}}$, $\mathbf{D}^k = \mathbf{D}_{I_{un}^k}$.
4. Repeat. Repeat 1st-3rd steps until $I^k = \emptyset$ or $I_{un}^k = \emptyset$.

Solve LRE: If $I_{un}^{\tilde{k}} \neq \emptyset$, solve the LRE of Eq. (13), obtaining solution $\hat{\mathbf{z}}$. And update the result, $\hat{\mathbf{y}}_{I_{un}^{\tilde{k}}} = \hat{\mathbf{z}}$.
Output: Return $\hat{\mathbf{y}}$.

Fig. 1 Procedure of the CLP algorithm.

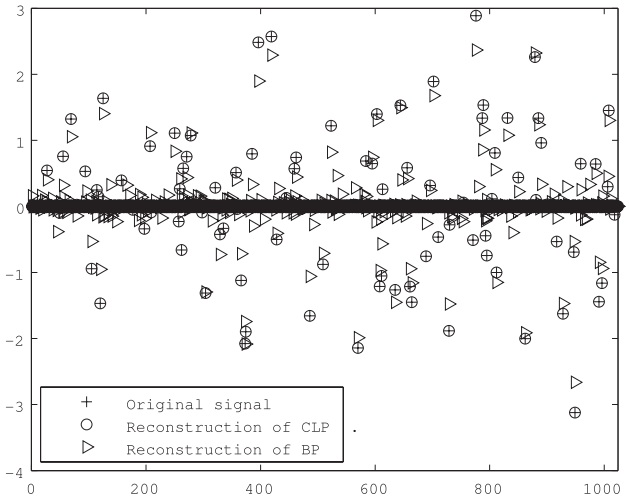


Fig. 2 An example of sparse signal recovery ($M = 1024, N = 256, T = 89$). The proposed method perfectly recovered the original signal, while BP failed since the signal has too many nonzeros.

4.1 Effectiveness Analysis

Before the analysis, we give a definition of an **oversampled block**. When the segment of a signal measured by the corresponding block has more than $\lfloor \frac{n}{2} \rfloor$ nonzeros, this block is defined as an **oversampled block**. Obviously, oversampled blocks are those of unsolvable low-dimensional systems. We now consider the distribution of nonzero entries into the blocks. In each subsystem, T nonzeros are randomly distributed into B ($B = \frac{M}{m}$) blocks. This is a classical "balls in bins" problem [28], [29]. It has been proven in [29] that when T balls are randomly thrown into B bins, the expected fraction of bins containing i balls is:

$$f(i, T) = \frac{1}{i!}\left(\frac{T}{B}\right)^i e^{-\frac{T}{B}}. \tag{14}$$

Thus, the fraction of bins that contain more than r balls is:

$$F(> r, T) = 1 - \sum_{i=0}^{r} \frac{1}{i!}\left(\frac{T}{B}\right)^i e^{-\frac{T}{B}}. \tag{15}$$

We consider the number of oversampled blocks at the first iteration. In Eq. (9), the number of oversampled blocks is $b_1 = B \cdot F(> \lfloor \frac{n}{2} \rfloor, T)$. Therefore, there are $M \cdot F(> \lfloor \frac{n}{2} \rfloor, T)$ unrecovered entries. Among them, the number of unsolved nonzero entries is

$$G(T) = T - \sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} i \cdot B \cdot f(i, T). \tag{16}$$

Then consider solving Eq. (10). The unsolved entries after solving Eq. (9) can also be considered to be randomly distributed into the blocks. In the same way, we can calculate that the number of oversampled blocks in Eq. (10) is $\lfloor B \cdot F(> \lfloor \frac{n}{2} \rfloor, G(T)) \rfloor$ (note the number should be an integer). Consequently, the number of unrecovered entries after the first iteration is $|I^1_{un}| = m \cdot \lfloor B \cdot F(> \lfloor \frac{n}{2} \rfloor, G(T)) \rfloor$. Therefore, we have the following theorem.

Theorem 2. *The CLP algorithm successfully recovers* $\mathbf{y}$ *with high probability when* $T < \overline{T}$*, where* $\overline{T}$ *is defined as the minimum number such that* $m \cdot \lfloor B \cdot F(> \lfloor \frac{n}{2} \rfloor, G(\overline{T})) \rfloor = s(\mathbf{D})$.

Based on the above analysis, Theorem 2 is obvious. From the procedure of the CLP algorithm, we can observe that $s(\mathbf{D}^{\tilde{k}}) \geq s(\mathbf{D})$ when $\mathbf{D}^{\tilde{k}}$ has no fewer columns than $s(\mathbf{D}) - 1$, since $\mathbf{D}^{\tilde{k}}$ is a submatrix of $\mathbf{D}$; furthermore, if $|I^{\tilde{k}}_{un}| < s(\mathbf{D})$, CLP can recover $\mathbf{y}$ successfully. $|I^{\tilde{k}}_{un}| < s(\mathbf{D})$ holds since $|I^{\tilde{k}}_{un}| \leq |I^1_{un}| < m \cdot \lfloor B \cdot F(> \lfloor \frac{n}{2} \rfloor, G(\overline{T})) \rfloor$.

Theorem 2 gives a bound that the CLP algorithm successfully recovers the original signal with high probability. This bound is not sharp since we only use the results of the first iteration during the analysis. However, this theorem is still weak since it relies on $s(\mathbf{D})$. The difficulty is that calculation of the *spark* of a matrix is of exponential complexity. Therefore, it is difficult to analyze how large $s(\mathbf{D})$ can be. However, we still can ensure that the CLP algorithm successfully recovers the sparse signal when T is sufficiently small such that $\lfloor B \cdot F(> \lfloor \frac{n}{2} \rfloor, G(T)) \rfloor$ (the number of unsolvable low-dimension systems after the first iteration) is small enough, e.g., $\lfloor B \cdot F(> \lfloor \frac{n}{2} \rfloor, G(T)) \rfloor = 0$, which means there is no unsolved low-dimensional system after the first iteration. Note that $\lfloor B \cdot F(> \lfloor \frac{n}{2} \rfloor, G(T)) \rfloor$ is a non-negative and non-decreasing function of T. Let $\tilde{T}$ denote the maximum integer such that the function value equals 0; then it holds that the function value equals 0 whenever $T \leq \tilde{T}$. For example, let $M = 2048$, $n = 2$ and $m = 16$ (consequently $N = 512$). When $T \leq 50$, $\lfloor B \cdot F(> \lfloor \frac{n}{2} \rfloor, G(T)) \rfloor = 0$ holds. Compared with the numerical results in the following section, we can observe that the practical recovery ability of the proposed method is much higher.

4.2 Complexity Analysis

This subsection presents the complexity analysis of CLP. Without loss of generality, assume that $L = 2$ and $N_1 = N_2$. In addition, the number of measurements is assumed to grow proportionally as the signal length grows ($\frac{N}{M}$ is constant). The reason for a constant ratio $\frac{N}{M}$ is as follows. When we consider the computation of the algorithm as the problem size M increases, we have to keep the problem setting unchanged: the sparsity of $\mathbf{y}$, $\frac{T}{M}$, is constant. In other words, as the length (M) of the sparse signal $\mathbf{y}$ increases, the ℓ_0-norm of $\mathbf{y}$, T, increases proportionally to M. Under this condition, let us consider the number of needed measurements N. Usually, $O(T \log \frac{M}{T})$ measurements are needed (see [30] for example); thus, N also increases proportionally to M since $N = O(T \log \frac{M}{T}) = O(M \frac{T}{M} \log \frac{M}{T})$, where $\frac{T}{M}$ is constant.

In the CLP algorithm, the most time-consuming part is solving the small systems of low-dimension equations. Using ESS to solve such a system requires a search over the subsets that consist of $\lfloor \frac{n}{2} \rfloor$ columns and solving an LS problem of Eq. (6). The worst case is testing all of the $\binom{m}{\lfloor \frac{n}{2} \rfloor}$ possible subsets. The computation (in flops) is $H = \binom{m}{\lfloor \frac{n}{2} \rfloor} \cdot G$ (G denotes the computation of solving the LS problem of Eq. (6), which can be solved with marginal cost of $O(n \cdot \lfloor \frac{n}{2} \rfloor)$). Although H is exponential to n, H is not large compared with the signal size M since n is quite small. For example, when $n = 2$ and $m = 16$, H is only about 32 flops. Moreover, since the block size, $n \times m$, is fixed, n is changeless as the signal size M grows. What changes is the number of blocks in each sub-system of equations, $B = \frac{M}{m}$. Therefore, H is constant. Obviously, solving a small system of determined equations takes much less computation than H. Totally, there are $\frac{2M}{m}$ low-dimensional systems in these two groups of Eqs. (9) and (10). Assume that each low-dimensional system is solved twice on average. This assumption is quite reasonable since most of the systems of low-dimensional equations are solved at the first time at the first iteration (as has been shown in the effectiveness analysis above), and we solve an unsolved one again only when substituting the newly solved entries from the other groups of equations. Thus, the computation is $\frac{4H}{m}M$, which is proportional to M. Solving the LRE of Eq. (13) consumes the second largest computation time. Since $\mathbf{D}^{\tilde{k}}$ is highly sparse, the solution can be quickly obtained by a Conjugate Gradient (CG) method. It needs $O(z)$ flops, where z is the number of nonzero elements. In $\mathbf{D}^{\tilde{k}}$, $z < 2nN$, the complexity is $O(2nN)$, which is also proportional to M. Note that in most cases of highly sparse signals, solving the LRE is unnecessary. The total time complexity of CLP is $O(M)$, which means CLP can recover a sparse signal in linear time. The space complexity of CLP is proportional to the number of nonzero elements in $\mathbf{D}$, $2nM$.

Comparison of the complexities of different algorithms is shown in Table 1, where OMP is Cholesky factorization-based, and BP is recast as a Linear Programming (LP) problem and solved via a primal-dual barrier method for convex optimization (PDCO) [31]. The complexities of OMP, BP (PDCO), and SP are summarized from [32], [8], and [14], respectively. Refer to them for details of analysis.

Table 1 Comparison of complexity. G denotes the Gaussian random matrix.

	BP+G	OMP+G	SP+G	BP+PF	OMP+PF	SP+PF	CLP
Time complexity	$O(M^3)$	$O(TNM)$	$O(TNM)$	$O(M^2logM)$	$O(TMlogM + T^3)$	$O(TMlogM)$	$O(M)$
Space complexity	$O(NM)$	$O(\frac{T^2}{2} + NM)$	$O(NM)$	$O(MlogM)$	$O(\frac{T^2}{2} + MlogM)$	$O(MlogM)$	$O(M)$

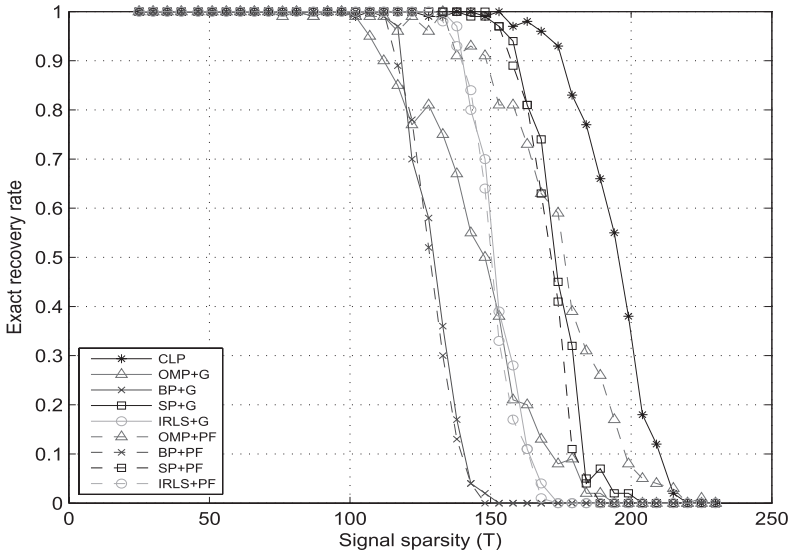


Fig. 3 Result of experiment 1: exact recovery rate versus signal sparsity T of Gaussian sparse signals ($M = 2048$, $N = 512$) when the comparative algorithms are with the Gaussian random matrix (denoted by G) and PF matrix.

5. Experimental Results

We performed three experiments in Matlab 7.5 on a dual-core 2.66 GHz desktop computer. In the first and second experiments, we compared the recovery abilities of different algorithms for different types of signals, and in the third experiment, we compared the execution times of different algorithms. In the experiments, we performed four other well-known algorithms: BP (PDCO solver), IRLS (ℓ_τ : τ gradually varies from 1 to 0.5), OMP, and SP with Gaussian random, PF, and PBD matrices. The programs of OMP and BP (PDCO) algorithms are from the SparseLab package (available from http://sparselab.stanford.edu), and the program of SP is provided by the authors (available from http://igorcarron.googlepages.com/cscodes)†. The program of IRLS was produced by ourselves. For the CLP algorithm, we set $n = 2$, $m = 16$, $L = 2$ for the PBD matrix in all experiments, and the blocks were $(n \times m)$ full *spark* Gaussian random matrices. Our experiments are of the following forms, similar to those of [1], [14]:

1. Choose the signal length M, the number of measurements N, and the signal sparsity T.
2. Construct a measurement matrix $\mathbf{D}$, where $\mathbf{D}$ is a Gaussian matrix, PF matrix, or PBD matrix.
3. Select the subset I ($|I| = T$) randomly from set $\{1, \cdots, M\}$.
4. Generate a sparse signal $\mathbf{y}$ by setting $\mathbf{y}_{\overline{I}} = \mathbf{0}$, and generating $\mathbf{y}_I$ in one of the following ways:
   - Gaussian sparse signals: draw the elements from independent Gaussian distributions with mean 0 and variance 1.
   - Bernoulli sparse signals: draw the elements from independent Bernoulli distributions, where the elements are 1 or -1 with equal probability.
5. Obtain the measurements $\mathbf{s} = \mathbf{D}\mathbf{y}$.
6. Recover $\mathbf{y}$ with the corresponding algorithm. An exact recovery is achieved when $\frac{\|\hat{\mathbf{y}}-\mathbf{y}\|_2}{\|\mathbf{y}\|_2} \le 10^{-3}$.
7. Repeat the 3rd to 6th *steps* r times ($r = 100$ in the first and second experiments and $r = 10$ in the third experiment).

In the first and second experiments, M=2048, N=512, and the signal sparsity T varied from 25 to 230 (0.05N to 0.45N). Note that the Bernoulli sparse signals are often used for comparative study, since they are challenging cases for recovery algorithms.

In the first experiment, we compared the recovery ability of CLP with those of the other algorithms implemented on classical measurement matrices, Gaussian random and PF matrices. Figures 3 and 4 show the results of the first

†Since some of the codes can not deal with the implicit matrix-vector multiplication (fast Fourier transform) or are not suitable for sparse matrices, we made some small modifications to make them suitable for such cases.

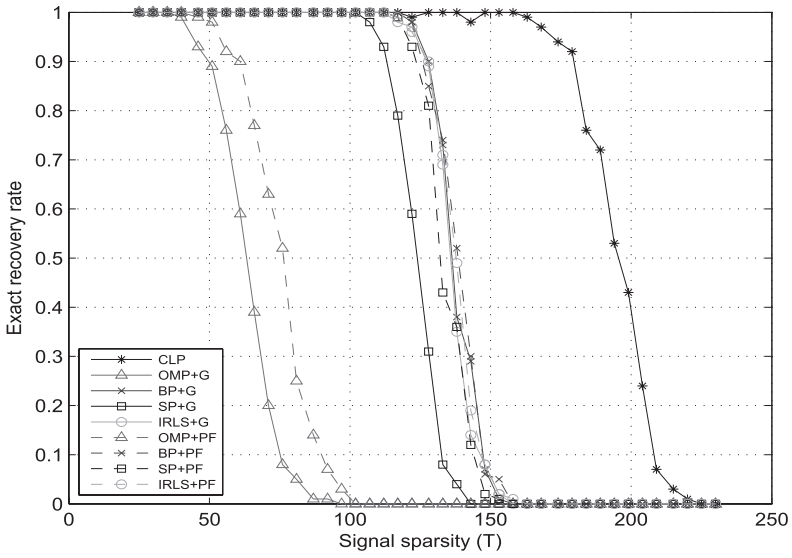


Fig. 4 Result of experiment 1: exact recovery rate versus signal sparsity T of Bernoulli sparse signals ($M = 2048$, $N = 512$) when the comparative algorithms are with the Gaussian random matrix (denoted by G) and PF matrix.

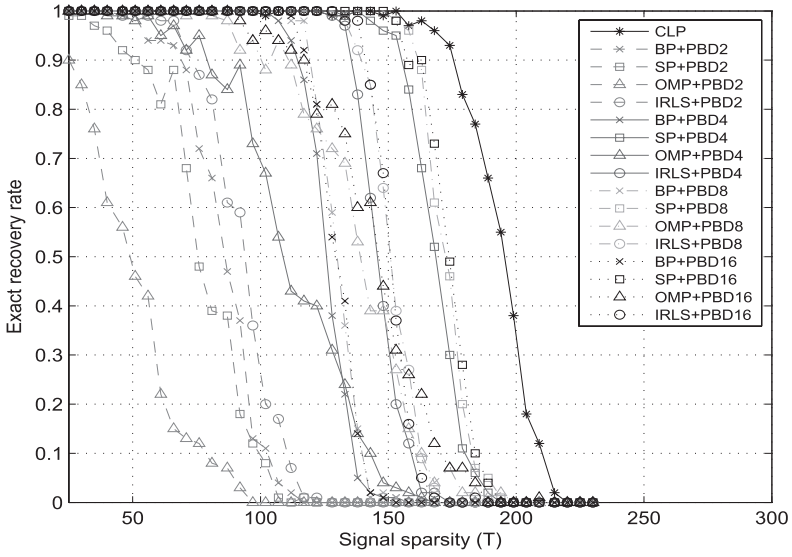


Fig. 5 Result of experiment 2: exact recovery rate versus signal sparsity T of Gaussian sparse signals ($M = 2048$, $N = 512$) when the comparative algorithms are with the PBD matrix.

experiment. In the case of Gaussian sparse signals, the recovery ability is in the order of CLP > SP > IRLS > BP ≈ OMP, while in the challenging case of Bernoulli sparse signals, the recovery ability is in the order of CLP > IRLS = BP > SP > OMP. We can observe that CLP has the highest recovery ability. Although the IRLS, OMP and SP algorithms achieved good results with Gaussian sparse signals, they did worse in the Bernoulli case. On the other hand, for BP and CLP, the performances are the same regardless of the signal type.

In the second experiment, we studied the performances of the algorithms with PBD matrices. It is obvious that the algorithms may work better with a dense PBD matrix than with a sparse one. For example, if we set $m = M$ ($n \cdot L = N$), then the PBD matrix becomes a fully dense matrix, which will provide optimal performances of these algorithms. However, the sparser the PBD matrix is, the lower complexities these algorithms have. Therefore, in this experiment, we gradually increased the fraction of nonzeros in the PBD matrix to study how sparse a PBD matrix can provide high performance. There are two ways to increase the denseness of PBD matrices. The first one is to increase n (fixing L), and the second one is to increase L (fixing n). However, based on numerical studies, we observed that increasing L gives faster growth of performance of the algorithms. This can be easily understood because a larger L

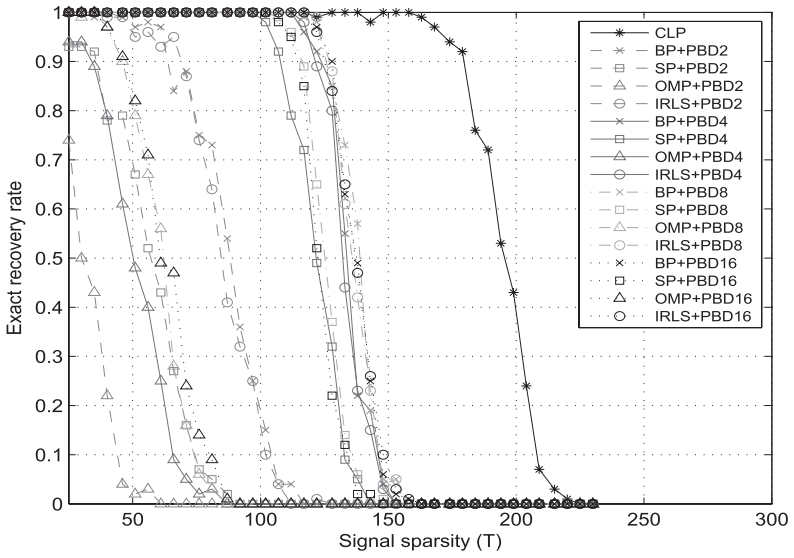


Fig. 6 Result of experiment 2: exact recovery rate versus signal sparsity T of Bernoulli sparse signals ($M = 2048$, $N = 512$) when the comparative algorithms are with the PBD matrix.

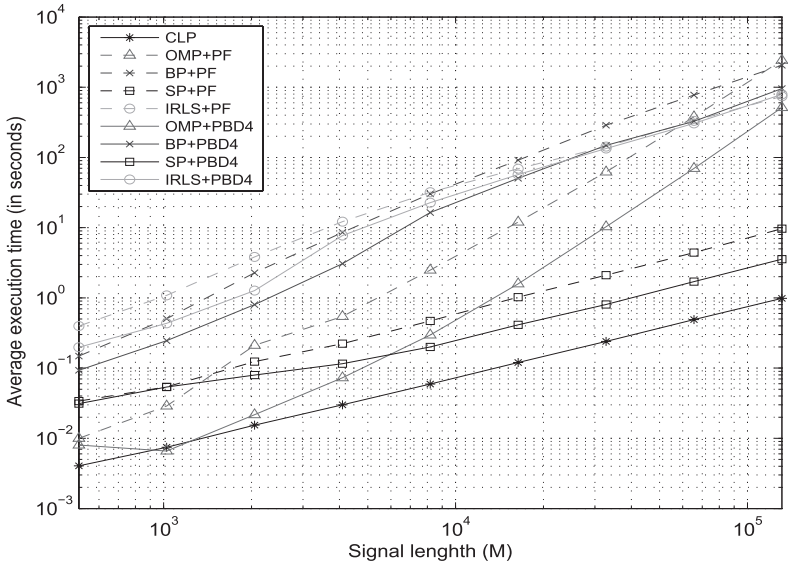


Fig. 7 Result of experiment 3: execution time versus signal length M ($N = 0.25M$, $T = 0.15N$).

gives more independently permuted sub-matrices. Therefore, in this experiment, we set $n = 2$ for all of the PBD matrices, and L gradually varied as 2, 4, 8, and 16. The corresponding PBD matrices are denoted by PBD2, PBD4, PBD8, and PBD16. The corresponding values of m are 16, 32, 64, 128, respectively ($m = \frac{Mn}{N}L$); the numbers of nonzeros in these matrices (Nm) are 8192, 16384, 32768, 65536, respectively. Figures 5 and 6 show the results of the second experiment. We can observe that all of the algorithms except OMP achieved the desired performance with PBD8, while PBD4 provides only slightly worse performance than PBD8. However, since PBD4 is sparser, it is a better choice as a measurement matrix.

In the third experiment, we compared the practical execution times of different algorithms, where the comparative algorithms were with the PF and PBD (PBD4) matrices. In this experiment, M varied from 2^9 to 2^{17}. We set the sampling rate $\frac{N}{M} = \frac{1}{4}$ and the signal sparsity $T = 0.15N$. In this experiment, we did not use the Gaussian random matrix since the complexity and storage requirement are too high to apply it in such large-scale simulations. Figure 7 shows the results of the third experiment. We can observe that CLP performed faster than SP and much faster than the other three algorithms. Moreover, as M varies, CLP took linear time to M, which can be observed more clearly in Fig. 8.

6. Conclusion and Future Work

In this paper, a novel algorithm, the CLP algorithm, for sparse signal recovery has been proposed on the basis of

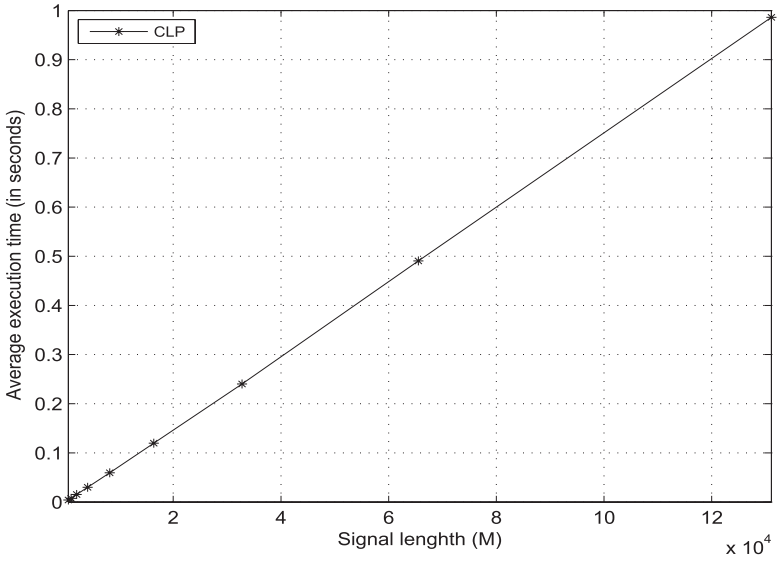


Fig. 8 Execution time of CLP versus signal length M ($N = 0.25M$, $T = 0.15N$).

a new structured sparse matrix, the PBD matrix. The proposed algorithm has advantages of low complexity and high sparse recovery ability. Both theoretical analysis and experimental results are shown to verify the validity of the proposed algorithm.

The problem of exactly recovering sparse signals from incomplete linear measurements without noise is considered in this paper. The proposed method can be used in several areas such as decoding real codes [3] and data stream computing [5]. In many other cases of interest, the measurements are with noise or the signal is not exactly but approximately sparse. Then exact recovery is impossible, and a good approximation is desired. For example, in compressed sensing of natural signals, the coefficients in the sparse domain (e.g., wavelet domain) need to be approximated. Therefore, our future work will focus on the following two directions. The first one is to apply the method proposed in this paper into applications, and the second one is to improve our method in approximation and noise cases.

Acknowledgements

This work was partly supported by Grant-in-Aid for Scientific Research (B) 21300030, Japan Society for the Promotion of Science (JSPS).

References

[1] E.J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," IEEE Trans. Inf. Theory, vol.52, no.2, pp.489–509, Feb. 2006.

[2] D.L. Donoho, "Compressed sensing," IEEE Trans. Inf. Theory, vol.52, no.4, pp.1289–1306, April 2006.

[3] E.J. Candès and T. Tao, "Decoding by linear programming," IEEE Trans. Inf. Theory, vol.51, no.12, pp.4203–4215, Dec. 2005.

[4] D.L. Donoho and M. Elad, "Optimally sparse representation from overcomplete dictionaries via ℓ^1-norm minimization," Proc. Nat. Acad. Sci. USA, vol.100, pp.2197–2002, March 2003.

[5] S. Muthukrishnan, Data Streams: Algorithms and Applications, ch. Syntax-directed program modularization, Now Publishers, 2005.

[6] E.J. Candès and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?," IEEE Trans. Inf. Theory, vol.52, no.12, pp.5406–5425, Dec. 2006.

[7] J.A. Tropp and A.C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," IEEE Trans. Inf. Theory, vol.53, no.12, pp.4655–4666, Dec. 2007.

[8] D.L. Donoho and Y. Tsaig, "Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit," Preprint, March 2006.

[9] Y. Tsaig and D.L. Donoho, "Extensions of compressed sensing," Signal Process., vol.86, pp.549–571, March 2006.

[10] S. Mallat and Z. Zhang, "Matching pursuits with timefrequency dictionaries," IEEE Trans. Signal Process., vol.41, no.12, pp.3397–3415, Dec. 1993.

[11] G. Davis, S. Mallat, and Z. Zhang, "Adaptive timefrequency decompositions," Opt. Eng., vol.33, pp.2183–2191, July 1994.

[12] D. Needell and R. Vershynin, "Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit," Found. Comput. Math., vol.9, pp.317–334, 2009.

[13] D. Needell and J. Tropp, "Cosamp: Iterative signal recovery from incomplete and inaccurate samples," Appl. Comput. Harmon. Anal., vol.26, pp.301–321, May 2009.

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

[15] S.S. Chen, D.L. Donoho, and M.A. Saunders, "Atomic decomposition by basis pursuit," SIAM J. Scientific Computing, vol.20, pp.33–61, 1999.

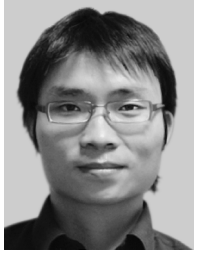
[16] R. Tibshirani, "Regression shrinkage and selection via the lasso," J.R. Statist. Soc. B, vol.58, pp.267–288, 1996.

[17] M.A.T. Figueiredo, R.D. Nowak, and S.J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," IEEE J. Sel. Top. Signal Process., Special Issue on Convex Optimization Methods for Signal Processing, vol.1, no.4, pp.586–598, Dec. 2007.

[18] D.L. Donoho and Y. Tsaig, "Fast solution of ℓ^1 norm minimization problems when the solution may be sparse," IEEE Trans. Inf. Theory, vol.54, no.11, pp.4789–4812, Nov. 2008.

[19] I. Daubechies, R. DeVore, M. Fornasier, and C.S. Gntrk, "Iteratively

reweighted least squares minimization for sparse recovery," Communications on Pure and Applied Mathematics, vol.63, pp.1–38, Jan. 2010.
[20] R. Chartrand, "Exact reconstruction of sparse signals via nonconvex minimization," IEEE Signal Process. Lett., vol.14, no.10, p.70710, 2007.
[21] E.J. Candès, M. Wakin, and S. Boyd, "Enhancing sparsity by reweighted l^1 minimization," J. Fourier Anal. Appl., vol.14, no.5, pp.877–905, 2008.
[22] H. Mohimani, M. Babaie-Zadeh, and C. Jutten, "A fast approach for overcomplete sparse decomposition based on smoothed ℓ^0 norm," IEEE Trans. Signal Process., vol.57, no.1, pp.289–301, Jan. 2009.
[23] R. Ashino, T. Nguyen-ba, and R. Vaillancourt, "Decoding low-dimensional linear codes by linear programming," Canadian Applied Mathematics Quarterly, vol.16, pp.241–254, 2008.
[24] A. Gilbert and P. Indyk, "Sparse recovery using sparse matrices," Proc. IEEE, vol.98, pp.937–947, 2010.
[25] E.J. Candès, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," Communications on Pure and Applied Mathematics, vol.59, pp.1207–1223, Aug. 2006.
[26] L. Gan, "Block compressed sensing of natural images," Proc. International Conference on Digital Signal Processing, pp.403–406, 2007.
[27] S. Mun and J.E. Fowier, "Block compressed sensing of images using directional transforms," Proc. IEEE International Conference on Image Processing, pp.3021–3024, 2009.
[28] A.A. Moghadam and H. Radha, "Randomness-in-structured ensembles for compressed sensing of images," Proc. IEEE International Conference on Image Processing, pp.3029–3032, 2009.
[29] http://www.mathpages.com/home/kmath199.htm
[30] P. Indyk, "Sparse recovery using sparse (random) matrices." http://people.csail.mit.edu/indyk/iisc.pdf
[31] M.A. Saunders, "Pdco: Primal-dual interior method for convex objectives." http://www.stanford.edu/group/SOL/software/pdco.html
[32] T. Blumensath and M.E. Davies, "Gradient pursuits," IEEE Trans. Signal Process., vol.56, no.6, pp.2370–2382, June 2008.



Zaixing He received his B.S. and M.S. degrees in Engineering from Zhejiang University, China in 2006 and 2008, respectively. He is currently a Ph.D. student in the Graduate School of Information Science and Technology, Hokkaido University. His research interests include sparse recovery, sparse representation, compressed sensing and their applications to image processing, signal processing and pattern recognition. He is a student member of the IEEE.

Takahiro Ogawa received his B.S., M.S. and Ph.D. degrees in Electronics and Information Engineering from Hokkaido University, Japan in 2003, 2005 and 2007, respectively. He is currently an assistant professor in the Graduate School of Information Science and Technology, Hokkaido University. His research interests are digital image processing and its applications. He is a member of the IEEE and Institute of Image Information and Television Engineers (ITE).

Miki Haseyama received her B.S., M.S. and Ph.D. degrees in Electronics from Hokkaido University, Japan in 1986, 1988 and 1993, respectively. She joined the Graduate School of Information Science and Technology, Hokkaido University as an associate professor in 1994. She was a visiting associate professor of Washington University, USA from 2005 to 2006. She is currently a professor in the Graduate School of Information Science and Technology, Hokkaido University. Her research interests include image and video processing and its development into semantic analysis. She is a member of the IEEE, Institute of Image Information and Television Engineers (ITE) and Acoustical Society of Japan (ASJ).