A Compressed Sensing-Based Low-Density Parity-Check Real-Number Code

Zaixing HE[1], *Takahiro OGAWA*[2], *Miki HASEYAMA*[2], *Xinyue ZHAO*[1*], *Shuyou ZHANG*[1]

[1]Department of Mechanical Engineering, Zhejiang University, Hangzhou, 310027 P. R. China
[2]Graduate School of Information Science and Technology, Hokkaido University, Sapporo, 0600814 Japan

zaixinghe@zju.edu.cn, ogawa@lmd.ist.hokudai.ac.jp, miki@ist.hokudai.ac.jp, zhaoxinyue@zju.edu.cn, zsy@zju.edu.cn

Abstract. *In this paper, we propose a novel low-density parity-check real-number code, based on compressed sensing. A real-valued message is encoded by a coding matrix (with more rows than columns) and transmitted over an erroneous channel, where sparse errors (impulsive noise) corrupt the codeword. In the decoding procedure, we apply a structured sparse (low-density) parity-check matrix, the Permuted Block Diagonal matrix, to the corrupted output, and the errors can be corrected by solving a compressed sensing problem. A compressed sensing algorithm, Cross Low-dimensional Pursuit, is used to decode the code by solving this compressed sensing problem. The proposed code has high error correction performance and decoding efficiency. The comparative experimental results demonstrate both advantages of our code. We also apply our code to cryptography.*

Keywords

Real-number error correcting codes, compressed sensing, sparse recovery, cross low-dimensional pursuit, cryptography.

1. Introduction

Error correcting codes are usually defined over finite fields, e.g., the binary field $\mathbb{F}_2$. Those defined over the real field, however, have been recognized as advantageous and studied for about three decades [1]–[6]. The target problem of decoding such codes is to recover an input real-valued message encoded by a coding matrix (codebook) and corrupted by gross errors, i.e., impulsive noise. At the beginning, commonly used real-number codes are based on orthogonal transforms, e.g., DFT [1]. Unfortunately, the stability of those codes is very poor for some error patterns, e.g., burst errors [4]. To improve the stability, Chen *et al.* proposed a code based on random matrices [4] such that the code is independent of error patterns. However, this code was proposed for correcting erasures (errors at known positions). Therefore, it cannot deal with errors with unknown positions. In recent years, highly robust codes based on compressed sensing have emerged [5]. In a compressed sensing-based code, high rate of errors can be corrected regardless of the error pattern.

The core problem of decoding compressed sensing-based real-number codes is to reconstruct the sparse error vector from a system of underdetermined equations [5], which is the well-known sparse recovery (or compressed sensing) problem. A common way to solve this problem is to find the sparsest solution to the underdetermined system. Unfortunately, directly searching for the sparsest solution in high dimensions is computationally intractable. However, many indirect methods are available for solving this problem. Commonly used methods include, but are not limited to, ℓ_1 minimization (ℓ_1-min) algorithms [7]–[10], nonconvex algorithms [11], [12], greedy algorithms [13]–[19], and special algorithms that are based on certain matrices [20]–[22]. All of these algorithms have their own comparative advantages of either high error correcting ability or low complexity. However, their complexities are still relatively high since these methods solve the problem in high dimensions, and their correcting abilities are relatively low because they solve the compressed sensing problem in indirect ways, e.g., minimizing the ℓ_1-norm, instead of the direct way. Here the direct way means combinationally searching for the sparsest solution from all subsets.

In this paper, we propose a novel compressed sensing-based code whose decoding algorithm is based on a low-density parity-check (LDPC) matrix: the Permuted Block Diagonal (PBD) matrix that we first proposed in [23]. We apply the PBD matrix to the corrupted output and use the Cross Low-dimensional Pursuit (CLP) algorithm that we first proposed in [24], [25], for reconstructing the error vector to recover the encoded message. During decoding, the high-dimensional compressed sensing problem is converted into groups of highly low dimensional problems. Thus, the decoding method has a very low complexity compared to existing algorithms. CLP can, in fact, perform in linear time. Furthermore, since in low dimensions, the compressed sensing problem can be solved in direct ways, the method has much higher error correcting ability than the existing algo-

[*] Corresponding author: zhaoxinyue@zju.edu.cn

rithms including ℓ_1-min and nonconvex algorithms. The experimental results demonstrate the advantages of fast decoding and powerful error correcting of our code. In addition, we apply our code to cryptography and the proposed cryptographic scheme is reliable since the corruption rate used in the scheme is beyond the error correcting abilities of existing algorithms.

Note that the LDPC code proposed in this paper is quite different from the well-known LDPC codes, whose recent variations were also used in compressed sensing, e.g., [26]. First, the proposed code is defined over the real field, while the well-known LDPC codes are defined over the finite field of $\mathbb{F}_2$, a binary field containing elements of {0,1}. Second, the constructions of the pairs of parity-check and coding matrices are different. The parity-check matrix used in the proposed code is the newly proposed PBD matrix, and the ways of constructing the coding matrix (presented in Section 3.1.2) are also different from those of the well-known LDPC codes, e.g., Gaussian elimination. Third, the decoding of the proposed code is distinct from that of the well-known LDPC codes. We decode the code by solving a compressed sensing problem. Therefore, the CLP algorithm for solving the compressed sensing problem is different from those for decoding the well-known LDPC codes, such as the belief propagation algorithm, the message passing algorithm, and the sum-product algorithm.

The remainder of this paper is organized as follows. In Section 2, notations are defined and the background of compressed sensing-based real-number codes is introduced. In Section 3, a novel LDPC real-number code is proposed. Experimental results are shown in Section 4 to verify the improvement of the proposed code. In Section 5, we apply the proposed code to cryptography. Section 6 is the conclusion and discussion of the dense small noise case. Some description of the proposition is briefly presented, since part of this work has been published in preliminary form in the IEEE International Conference on Acoustics, Speech, and Signal Processing (IEEE ICASSP 2011), Prague, Czech Republic, May 2011 [24].

2. Compressed Sensing-Based Real-Number Codes

In this section, we introduce the compressed sensing-based real-number codes. Before the introduction of these codes, the notations used in this paper are defined.

2.1 Notations

In this paper, we use the following notations: $||\mathbf{e}||_0$ denotes the ℓ_0-norm of vector $\mathbf{e}$, counting the number of nonzero entries of $\mathbf{e}$. $\lfloor r \rfloor$ is the largest integer not greater than real number r. $\mathbf{W}_I$ denotes the submatrix (or subvector) of $\mathbf{W}$, consisting of the columns (or elements) in index set I. $|I|$ is referred to as the number of elements in set I. $\mathbf{W}^{\top}$ is the transpose of matrix (or vector) $\mathbf{W}$. $\mathbf{W}(\mathbf{p})$ denotes the permuted matrix (or vector) generated from $\mathbf{W}$ and the permutation vector $\mathbf{p}$, where $\forall i, \mathbf{W}(\mathbf{p})_i = \mathbf{W}_{\mathbf{p}(i)}$. $\mathbf{p}^{-1}$ is referred to as the inverse permutation vector of $\mathbf{p}$ such that $\forall j = \mathbf{p}(i)$, $\mathbf{p}^{-1}(j) = i$.

2.2 Compressed Sensing

Compressed sensing is a new paradigm of data acquisition [27],[28]. In this new paradigm, a sparse signal is sensed by a measurement matrix (with fewer rows than columns) to both sample and compress it in a single operation, and a small number of linear measurements are obtained. Furthermore, the original sparse signal can be recovered from these incomplete linear measurements when the measurement matrix satisfies certain conditions and the original signal is sufficiently sparse.

Compressed sensing includes two stages: sensing and reconstruction. In sensing, M linear measurements are obtained using an $M \times N$ ($M \ll N$) matrix $\mathbf{D}$ to sample (multiply) a sparse vector $\mathbf{e}$:

$$\mathbf{s} = \mathbf{D}\mathbf{e}. \tag{1}$$

In the reconstruction stage, the original sparse signal $\mathbf{e}$ needs to be recovered from these incomplete linear measurements. A common way to recover $\mathbf{e}$ is to find the sparsest solution to (1) by solving the following optimization problem:

$$(P_0) \ : \ \min ||\mathbf{z}||_0 \quad \text{subject to} \quad \mathbf{D}\mathbf{z} = \mathbf{s}. \tag{2}$$

This optimization problem is well-known as an NP-hard problem.

A concrete example is as follows:

$$\begin{bmatrix} 32 & 16 & 28 & 5 & 31 & 15 & 4 & 9 \\ 22 & 11 & 17 & 29 & 27 & 1 & 18 & 20 \\ 6 & 30 & 14 & 21 & 26 & 23 & 24 & 10 \\ 3 & 7 & 8 & 25 & 19 & 2 & 13 & 12 \end{bmatrix} \times \mathbf{e} = \begin{bmatrix} 20 \\ 29 \\ 54 \\ 20 \end{bmatrix} \tag{3}$$

where $\mathbf{e} = [0\ 1\ 0\ 0\ 0\ 0\ 1\ 0]^{\top}$. We try to recover $\mathbf{e}$ by directly solving (2). Assuming that the number of nonzero elements in $\mathbf{e}$ is known to be two, all the submatrices, consisting of two columns, of $\mathbf{D}$ need to be searched. There are 28 submatrices to be searched. As the dimension and the number of nonzero elements increase, the number of such submatrices grows exponentially.

Therefore, directly solving such a problem in high dimensions is computationally intractable. Many efficient algorithms, which solve the problem in indirect ways, are proposed to find sparse solutions. An alternative to (2) is:

$$(P_p) \ : \ \min ||\mathbf{z}||_p^p \quad \text{subject to} \quad \mathbf{D}\mathbf{z} = \mathbf{s} \tag{4}$$

where $0 < p \leq 1$. When $p = 1$, (4) is convex and can be recast as a Linear Program (LP). The corresponding algorithms are called ℓ_1-min algorithms. When $p < 1$, (4) is non-convex, and it can be approximated by an Iteratively Reweighted Least Squares (IRLS) algorithm, e.g., [12]. The greedy algorithms find sparse solutions in a different way: the coordinates and amplitudes of non-zeros of $\mathbf{e}$ are determined step by step. The three families of algorithms do not rely on specific matrices (**D**). There is another family of algorithms that are based on sparse matrices in order to accelerate the solving procedure, e.g., Sequential Sparse Matching Pursuit (SSMP) [21].

2.3 Error Correction Based on Compressed Sensing

Consider transmitting a message $\mathbf{x} \in \mathbb{R}^K$ by encoding it with a full *rank* matrix $\mathbf{F} \in \mathbb{R}^{N\times K}(N > K)$. Furthermore, a small fraction of entries of the codeword are corrupted over the transmitting channel. Thus, the corrupted output can be written as:

$$\mathbf{y} = \mathbf{F}\mathbf{x} + \mathbf{e} \tag{5}$$

where $\mathbf{e} \in \mathbb{R}^N$ is a sparse error vector. The final objective is to exactly recover $\mathbf{x}$ with knowledge of the corrupted output $\mathbf{y}$ and coding matrix $\mathbf{F}$.

To reconstruct $\mathbf{x}$ from $\mathbf{y}$ and $\mathbf{F}$, one can first construct a matrix $\mathbf{D} \in \mathbb{R}^{M\times N}(M = N-K)$ such that $\mathbf{D}\mathbf{F} = \mathbf{0}$. $\mathbf{D}$ is a matrix whose rows span the null space of $\mathbf{F}^\top$. The matrix $\mathbf{D}$ can also be viewed as a parity-check matrix. Then one can apply $\mathbf{D}$ to the corrupted output $\mathbf{y}$ and obtain:

$$\begin{aligned} \mathbf{s} &= \mathbf{D}\mathbf{y} \\ &= \mathbf{D}\mathbf{e}. \end{aligned} \tag{6}$$

Note that reconstructing $\mathbf{e}$ is a sufficient condition for reconstructing $\mathbf{x}$, since

$$\mathbf{x} = (\mathbf{F}^\top\mathbf{F})^{-1}\mathbf{F}^\top(\mathbf{y}-\mathbf{e}). \tag{7}$$

Specifically, when the columns of $\mathbf{F}$ are orthonormal,

$$\mathbf{x} = \mathbf{F}^\top(\mathbf{y}-\mathbf{e}). \tag{8}$$

Therefore, the decoding problem is reduced to the problem of reconstructing a sparse vector $\mathbf{e}$ from an underdetermined system (6). Since (6) is the same as (1), it can be reconstructed by solving a compressed sensing problem, where the parity-check matrix and sparse error vector in decoding correspond to the measurement matrix and sparse signal in compressed sensing, respectively. The entire procedure of error correction via compressed sensing is illustrated in Fig. 1.

3. Proposed LDPC Real-Number Code

In this section, we present our encoding and decoding algorithms in 3.1 and 3.2, respectively.

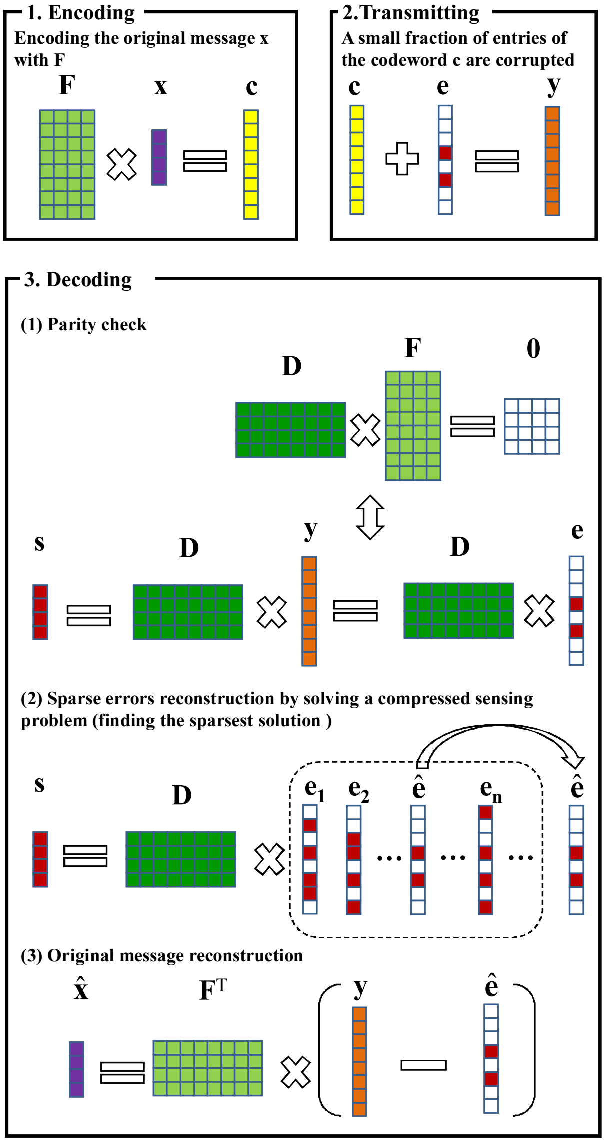

1. Encoding
Encoding the original message x with F
F
x
c
2.Transmitting
A small fraction of entries of the codeword c are corrupted
c
e
y
3. Decoding
(1) Parity check
D
F
0
s
D
y
D
e
(2) Sparse errors reconstruction by solving a compressed sensing problem (finding the sparsest solution)
s
D
e_1
e_2
$\hat{e}$
e_n
$\hat{e}$
(3) Original message reconstruction
$\hat{x}$
F^T
y
$\hat{e}$

Fig. 1. The procedure of error correction via compressed sensing (columns of **F** are orthonormal).

3.1 Encoding

Our encoding procedure is as follows: first, a PBD matrix $\mathbf{D}$ is generated as a parity-check matrix; then a coding matrix $\mathbf{F}$, whose columns span the null space of $\mathbf{D}$, is created; finally, a real-number input $\mathbf{x}$ is encoded by $\mathbf{F}$. The remainder of this subsection presents the construction of the PBD matrix and the corresponding coding matrices.

3.1.1 Construction of Parity-Check Matrix

The PBD matrix used in this work is different from that of [23] since it is non-binary. It can be generated as follows: first, we generate L matrices that are block diagonal: $\mathbf{W}_1 \in \mathbb{R}^{M_1\times N}, \cdots, \mathbf{W}_L \in \mathbb{R}^{M_L\times N}$, where $M_1 + \cdots + M_L = M$, and $\mathbf{W}_i = diag(\mathbf{w}_i, \cdots, \mathbf{w}_i)$ $(i \in \{1, \cdots, L\})$. The block $\mathbf{w}_i \in \mathbb{R}^{m\times n}$ $(m < n)$ is a low-dimensional full *spark* matrix, which is defined in [24]. A low-dimensional full *spark* matrix can be constructed as follows:

(1) generate an $m \times n$ matrix, e.g., a random matrix;

(2) examine whether all of the submatrices, consisting of m columns, have full *rank*; otherwise, repeat steps (1) and (2).

Next, L different random permutations of set $\{1, \cdots, N\}$, namely $\mathbf{p}_1, \cdots, \mathbf{p}_L$, are generated. Finally, a PBD matrix $\mathbf{D} \in \mathbb{R}^{M \times N}$ is constructed by permuting the block diagonal matrices with these permutations independently. Mathematically,

$$\mathbf{D} = \begin{bmatrix} \mathbf{W}_1(\mathbf{p}_1) \\ \vdots \\ \mathbf{W}_L(\mathbf{p}_L) \end{bmatrix}. \tag{9}$$

To reduce the complexity of decoding, we suggest m be small even numbers such as two or four [24].

3.1.2 Construction of Coding Matrix

There are several ways to construct a coding matrix $\mathbf{F}$ satisfying $\mathbf{DF} = \mathbf{0}$. Here, we show three simple approaches:

Construction 1: QR decomposition

Let $\mathbf{Q} \in \mathbb{R}^{N \times N}$ and $\mathbf{R} \in \mathbb{R}^{N \times M}$ be the QR decomposition matrices of $\mathbf{D}^\top$. A coding matrix $\mathbf{F}$ with orthonormal columns can be generated by choosing the columns of $\mathbf{Q}$ with indices from $M+1$ to N.

Construction 2: Gram-Schmidt orthogonalization

First, a random matrix $\mathbf{G} \in \mathbb{R}^{N \times K}$ with i.i.d. (independent and identically distributed) random entries, e.g., the Gaussian random matrix, is generated. Let $\mathbf{A} = [\mathbf{D}^\top \ \mathbf{G}]$. $\mathbf{A}$ has full *rank* with high probability since $\mathbf{G}$ is random. Then the columns of $\mathbf{A}$ are orthogonalized by the Gram-Schmidt process. A coding matrix $\mathbf{F}$ with orthonormal columns can be generated by choosing the columns of orthogonalized $\mathbf{A}$ with indices from $M+1$ to N. Note that the Gram-Schmidt process can also be used in QR decomposition. This method is similar to Construction 1.

Construction 3: Projection operator

Let $\mathbf{P}$ be the projection operator over $\mathbf{D}^\top$: $\mathbf{P} = \mathbf{D}^\top(\mathbf{D}\mathbf{D}^\top)^{-1}\mathbf{D}$. An $N \times K$ matrix $\mathbf{G}$ is generated such that $\mathbf{A}$, $\mathbf{A} = [\mathbf{D}^\top \ \mathbf{G}]$, has full *rank*. A coding matrix $\mathbf{F}$ can be generated as follows: $\mathbf{F} = \mathbf{G} - \mathbf{PG}$. $\mathbf{G}$ can be generated by randomly choosing K columns of some orthogonal transform, e.g., the Hadamard transform. In this case, the matrix-vector multiplication $\mathbf{Fx}$ or $\mathbf{F}^\top\mathbf{y}$ can be implemented implicitly: the operator $\mathbf{G}$ can be implemented by using the corresponding fast transform algorithm and the operator $\mathbf{P}$ can be implemented by using some numerical method. Therefore, the memory requirement is low and high-dimensional encoding and decoding are possible.

Constructions 1 and 2 generate coding matrices whose columns are orthonormal such that the encoded message can be reconstructed by (8). One disadvantage of Construction 2 is that it is not unambiguous, since we do not know which of the original signals were taken as the first. Construction 3 generates a coding matrix whose columns are non-orthogonal; thus, the encoded message must be reconstructed by (7), which takes more computation than (8). The advantage of Construction 3 is that the corresponding coding matrix can be implemented implicitly such that much less memory is consumed compared to Constructions 1 and 2 and high-dimensional messages can be encoded and decoded.

3.2 Decoding

As introduced in the previous section, the main problem of decoding a compressed sensing-based code is to recover the sparse error vector $\mathbf{e}$ from the underdetermined system,

$$\mathbf{Dz} = \mathbf{s} \tag{10}$$

where $\mathbf{s} = \mathbf{De}$. The conventional methods recover $\mathbf{e}$ in high dimensions. However, when $\mathbf{D}$ is a PBD matrix, there is no need to solve the high-dimensional problem directly. The PBD matrix converts the high-dimensional system of equations into L subsystems; in each subsystem, the corresponding matrix is block diagonal. Thus, each subsystem can be considered as a group of highly low dimensional systems corresponding to the blocks; entries can be recovered by solving these highly low dimensional systems. Moreover, the entries solved from each subsystem can be substituted into the other subsystems to solve more entries.

Consider the small example of (3) again. If we use a PBD matrix as the matrix $\mathbf{D}$, the linear system becomes

$$\begin{bmatrix} 0 & 0 & 6 & 0 & 0 & 4 & 8 & 3 \\ 0 & 0 & 5 & 0 & 0 & 7 & 2 & 1 \\ 6 & 3 & 0 & 4 & 8 & 0 & 0 & 0 \\ 5 & 1 & 0 & 7 & 2 & 0 & 0 & 0 \end{bmatrix} \times \mathbf{e} = \begin{bmatrix} 8 \\ 2 \\ 3 \\ 1 \end{bmatrix}. \tag{11}$$

It is equivalent to the following two low-dimensional systems:

$$\begin{bmatrix} 6 & 4 & 8 & 3 \\ 5 & 7 & 2 & 1 \end{bmatrix} \times \begin{bmatrix} e_3 \\ e_6 \\ e_7 \\ e_8 \end{bmatrix} = \begin{bmatrix} 8 \\ 2 \end{bmatrix} \tag{12}$$

and

$$\begin{bmatrix} 6 & 3 & 4 & 8 \\ 5 & 1 & 7 & 2 \end{bmatrix} \times \begin{bmatrix} e_1 \\ e_2 \\ e_4 \\ e_5 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}. \tag{13}$$

We can find the sparsest solution to each system directly since the dimension is very low. The solutions are $[e_3\ e_6\ e_7\ e_8]^\intercal = [0\ 0\ 1\ 0]^\intercal$ and $[e_1\ e_2\ e_4\ e_5]^\intercal = [0\ 1\ 0\ 0]^\intercal$. In this way, all the elements in $\mathbf{e}$ can be recovered.

The presentation of the algorithm is organized as follows: first, we describe the entire procedure of the algorithm; next, we describe the details of recovering entries from highly low dimensional systems.

3.2.1 Whole Decoding Procedure

For the PBD matrix, using $\mathbf{D}$ to multiply $\mathbf{e}$ can be considered as the following procedure. The vector $\mathbf{e}$ is permuted by the L inverse permutations $\mathbf{p}_1^{-1}, \cdots, \mathbf{p}_L^{-1}$ independently, and then multiplied by the L block diagonal matrices: $\mathbf{W}_1, \cdots, \mathbf{W}_L$. Mathematically,

$$\mathbf{De} = \begin{bmatrix} \mathbf{W}_1\mathbf{e}(\mathbf{p}_1^{-1}) \\ \vdots \\ \mathbf{W}_L\mathbf{e}(\mathbf{p}_L^{-1}) \end{bmatrix} = \mathbf{s}. \tag{14}$$

Thus, the entries of $\mathbf{e}$ can be solved from the following L systems separately:

$$\begin{aligned} \mathbf{W}_1\mathbf{z} &= \mathbf{s}_1, \\ &\vdots \\ \mathbf{W}_L\mathbf{z} &= \mathbf{s}_L, \end{aligned} \tag{15}$$

where $\mathbf{s}_i$ ($i \in \{1, \cdots, L\}$) denotes the i-th segment of $\mathbf{s}$. In each system, since matrix $\mathbf{W}_i$ is block diagonal, the high-dimensional system can be divided into a group of highly low-dimensional systems. Therefore, part of the entries can be easily solved in low dimensions. Furthermore, we can repeat solving these systems step by step: in each step, the entries that have already been solved can be substituted into these systems one by one to solve more entries. This is the cross solving procedure [24]. We named this algorithm Cross Low-dimensional Pursuit. The entire procedure of the algorithm is shown in Fig. 2. The CLP algorithm has linear complexity; its complexity is much lower than those of existing algorithms.The complexity analysis can be found in [24].

3.2.2 Low-Dimensional Recovery

As previously presented, a block diagonal matrix converts a high-dimensional system of equations into groups of highly low-dimensional systems; we recover the entries from the low-dimensional systems. Here we present the method to recover entries from low-dimensional systems. Suppose that a low-dimensional vector $\mathbf{f}$ is measured by a matrix $\mathbf{w}$: $\mathbf{wf} = \mathbf{u}$, where $\mathbf{w} \in \mathbb{R}^{m \times n}$ has full *rank* ($m \geq n$) or full *spark* ($m < n$). We wish to recover $\mathbf{f}$ from the following low-dimensional system:

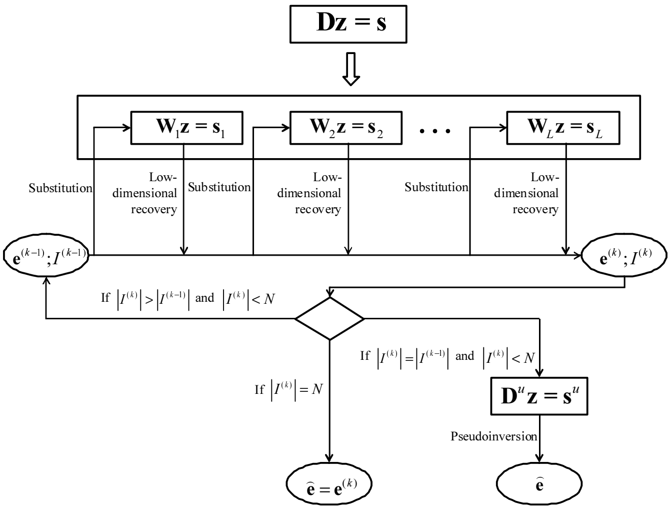

$\mathbf{Dz} = \mathbf{s}$
$\mathbf{W}_1\mathbf{z} = \mathbf{s}_1$
$\mathbf{W}_2\mathbf{z} = \mathbf{s}_2$
. . .
$\mathbf{W}_L\mathbf{z} = \mathbf{s}_L$
Substitution
Low-dimensional recovery
Substitution
Low-dimensional recovery
Substitution
Low-dimensional recovery
$\mathbf{e}^{(k-1)}; I^{(k-1)}$
$\mathbf{e}^{(k)}; I^{(k)}$
If $|I^{(k)}| > |I^{(k-1)}|$ and $|I^{(k)}| < N$
If $|I^{(k)}| = |I^{(k-1)}|$ and $|I^{(k)}| < N$
If $|I^{(k)}| = N$
$\mathbf{D}^u\mathbf{z} = \mathbf{s}^u$
Pseudoinversion
$\hat{\mathbf{e}} = \mathbf{e}^{(k)}$
$\hat{\mathbf{e}}$

Fig. 2. Procedure of the CLP algorithm. $\mathbf{e}^{(k)}$ and $I^{(k)}$ denote the subvector of recovered entries and their corresponding indices set in the k-th step of the cross solving procedure, respectively.

$$\mathbf{wz} = \mathbf{u} \tag{16}$$

where $\mathbf{u} = \mathbf{wf}$. When $m \geq n$, the system can be easily solved: $\hat{\mathbf{f}} = (\mathbf{w}^\intercal\mathbf{w})^{-1}\mathbf{w}^\intercal\mathbf{u}$ when $m > n$ or $\hat{\mathbf{f}} = \mathbf{w}^{-1}\mathbf{u}$ when $m = n$.

When $m < n$, (16) is an underdetermined system. In this case, we find the sparsest solution as follows:

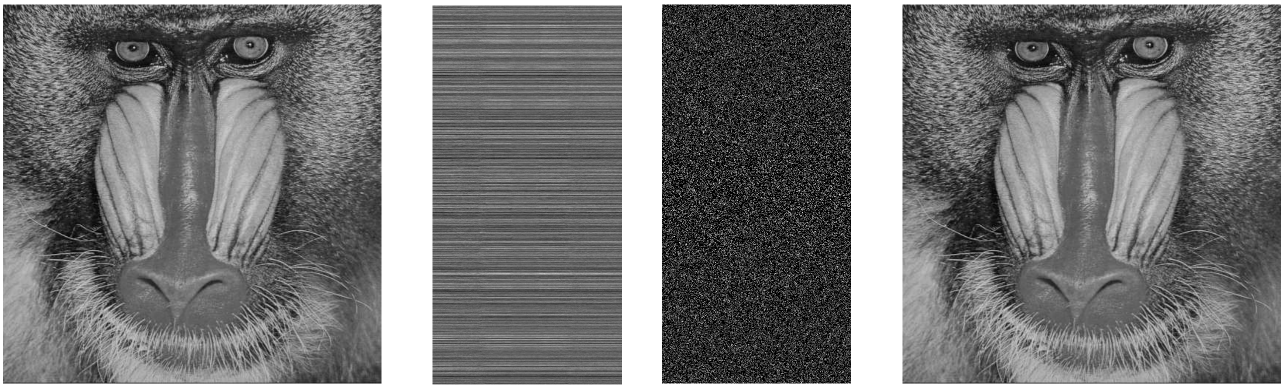
$$(mP_0) : \quad \min \|\mathbf{z}\|_0 \quad \text{s.t.} \quad \mathbf{wz} = \mathbf{u}. \tag{17}$$

Different from the original high-dimensional optimization problem of (2), the dimension of this problem is very low. Thus, directly searching for the sparsest solution is possible. We solve it in a direct and natural way to find a $\lfloor \frac{m}{2} \rfloor$-sparse solution: exhaustive searches over all subsets of $\lfloor \frac{m}{2} \rfloor$ columns of $\mathbf{w}$. We call it Exhaustive Subset Searching (ESS). The details of ESS can be found in [24].

3.2.3 Practical Examples

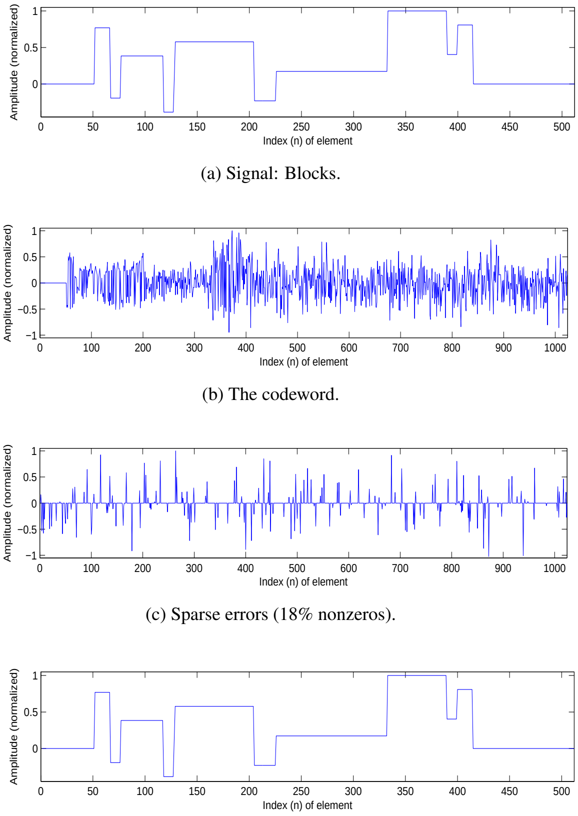
Two examples of our code are shown in Fig. 4 and Fig. 3. In the first example, the original signal Blocks [29] of length 512 was encoded by a 1024×512 coding matrix which was generated by Construction 1 (QR decomposition), and corrupted by a randomly generated sparse vector (Gaussian distribution) with a corruption rate of 18%*. The locations and values of the nonzeros of the sparse vector are determined as follows (in MATLAB): run the *randperm*(1024) function, and choose the first 18% indices as the locations of nonzeros; run the *randn*(184, 1), and take the output as the values of nonzeros. Finally, the original Blocks signal was exactly recovered by the CLP algorithm.

*Similar to other algorithms, the bound of corruption rate that can be handled by the proposed decoding algorithm depends on the ratio of $\frac{K}{N}$. In the next section, it will be shown that the bounds corresponding to the ratios of $\frac{7}{8}$ and $\frac{3}{4}$ are about 4% and 8%, respectively (Fig. 6 and Fig. 7). In the two examples, we choose 18% as the corruption rate, since it can be handled by the proposed code when the ratio of $\frac{K}{N}$ is $\frac{1}{2}$.



(a) Image: Mandrill (b) The codeword (c) Sparse errors (d) Exact reconstruction by CLP

Fig. 3. An example of the proposed code with the image Mandrill (the sparse errors have 18% nonzeros in each column).



(a) Signal: Blocks.

(b) The codeword.

(c) Sparse errors (18% nonzeros).

(d) Exact reconstruction by CLP.

Fig. 4. An example of the proposed code with the signal Blocks.

The second example is the 512 × 512 image Mandrill. We encoded each column of it with the same 1024 × 512 coding matrix which was generated by Construction 2 (Gram-Schmitt orthogonalization), and corrupted 18% entries of each column of the codeword by Gaussian distributed sparse errors. Finally, it was exactly recovered by the CLP algorithm.

4. Numerical Results

In this section, we conducted two experiments to investigate the practical error correcting ability and implementation time of CLP. Both experiments were performed in MATLAB 7.5 on a dual-core 2.66 GHz desktop computer. For CLP, we set $L = 4$ and $m = 2$. The main procedures of constructing the parity-check matrix and the decoding algorithm are shown in Fig. 5, where the main relevant MATLAB functions or operators are attached in brackets.

In the first experiment, we compare the error correcting ability of CLP with four well-known approaches: ℓ_1-min, IRLS (for solving the nonconvex optimization problem: (P_p) with $p < 1$), Subspace Pursuit (SP) [19], and SSMP [21]. The concrete solver for ℓ_1-min is PDCO [30], and the concrete IRLS algorithm is proposed by Daubechies *et al.* [12], where p gradually varies from 1 to 0.5. ℓ_1-min and IRLS algorithms were reported to have high sparse solution abilities (corresponding to high error correcting abilities). SP is a greedy algorithm and has lower complexity than the former approaches. SSMP is based on a binary sparse matrix and has sub-linear complexity; its complexity is the lowest among those of the four algorithms.

The program of ℓ_1-min is from the SparseLab package [31], and those of SP and SSMP are provided by the authors [32], [33]. We produced the program of IRLS. To ensure the reliability of the program, we have reproduced the results reported in [12] using our program. For ℓ_1-min, IRLS, and SP, we used Gaussian random matrices with normalized columns as parity-check matrices. The parity-check matrix for SSMP was "countmin8". We chose these matrices for the corresponding algorithms since they perform best with these matrices.

Our experiment, which is similar to [5], is as follows: (1) generate the pairs of parity-check matrices and coding

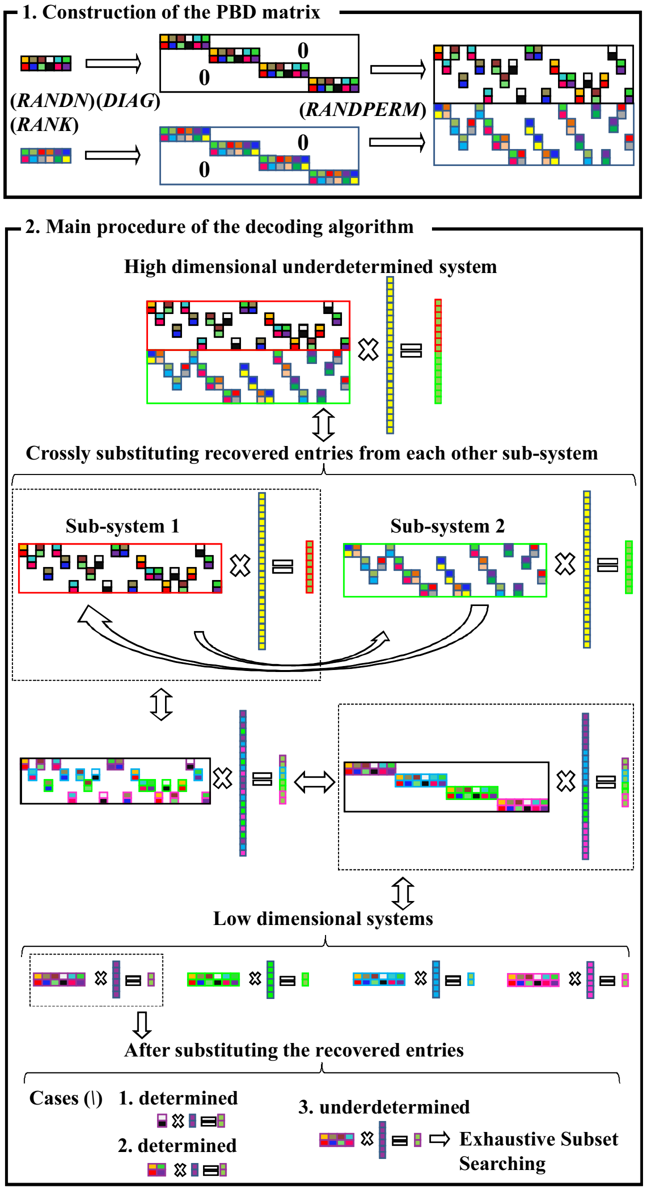

1. Construction of the PBD matrix
0
0
(RANDN)(DIAG)
(RANK)
(RANDPERM)
0
0
2. Main procedure of the decoding algorithm
High dimensional underdetermined system
Crossly substituting recovered entries from each other sub-system
Sub-system 1
Sub-system 2
Low dimensional systems
After substituting the recovered entries
Cases (Ⅰ) 1. determined
2. determined
3. underdetermined
Exhaustive Subset Searching

Fig. 5. The main procedures of constructing the parity-check matrix and the decoding algorithm ($L = 2$; Contents in brackets are the relevant MATLAB functions and operators).

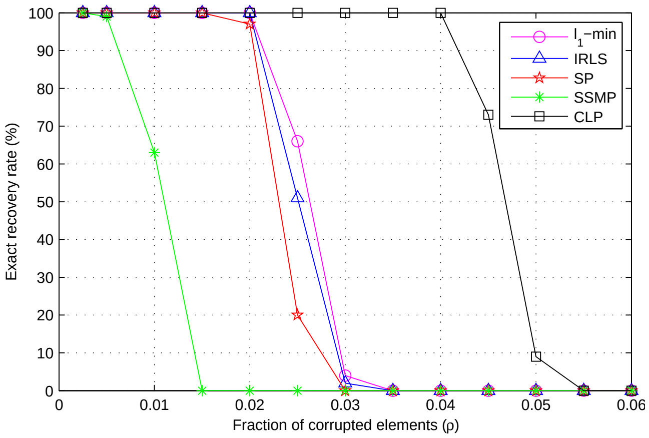


Fig. 6. Exact decoding rate versus fraction of corruption with $K = 1792$, $N = 2048$ ($\frac{K}{N} = \frac{7}{8}$).

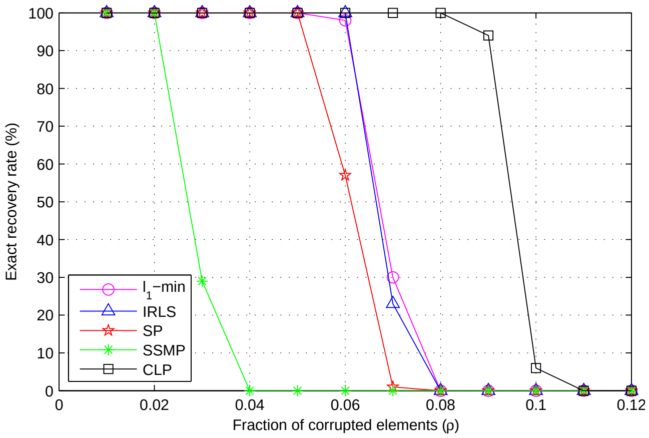


Fig. 7. Exact decoding rate versus fraction of corruption with $K = 1536$, $N = 2048$ ($\frac{K}{N} = \frac{3}{4}$).

matrices $\mathbf{D},\mathbf{F}$ by Construction 1 described in 3.1.2; (2) generate a random signal $\mathbf{x}$ and make $\mathbf{Fx}$; (3) generate a sparse error vector $\mathbf{e}$ with ρ percentage of randomly located ± 1* and make $\mathbf{Fx}+\mathbf{e}$; (4) decode the corrupted output by using $\mathbf{D}$ and the corresponding algorithms; (5) gradually increase ρ, and for each ρ, repeat steps (2)-(4) for 100 times to compute the percentage of exact recovery ($\frac{\|\hat{\mathbf{x}}-\mathbf{x}\|_2}{\|\mathbf{x}\|_2} \leq 10^{-5}$).

The second experiment was to investigate the practical efficiency of the decoding algorithm of the proposed code. We set $\frac{N}{M} = 4$ ($\frac{K}{N} = \frac{3}{4}$), and N varies from 2^9 to 2^{17}. A log-log plot shown in Fig. 8 describes the average computational time of CLP, compared to those of the ℓ_1-min, IRLS, and SP algorithms. Compared to the figure (Fig. 2) in [24], the problem size is much larger, and the comparative algorithms are slightly different, comparing IRLS instead of OMP. We can observe that CLP performed much faster than the other algorithms. In the experiment, we did not compare the SSMP algorithm, since its core program is produced in C language. Therefore, it is unfair to compare the four algorithms, whose programs are produced in MATLAB, with it. However, our algorithm has lower complexity than SSMP, since CLP has linear complexity. A normal plot shown in Fig. 9 verified the linear complexity of the CLP algorithm, while it was reported that the complexity of SSMP is sublinear [21].

The obtained results are shown in Fig. 6 and Fig. 7. It can be seen that CLP has the highest error correcting ability and performed much better than the other algorithms. Comparing the two algorithms that both use sparse parity-check matrices, CLP and SSMP, it can be seen that CLP corrected a much higher rate of errors.

*We use such a type of errors for experiments, since it is a challenging case for compressed sensing algorithms and is often used for comparative study.

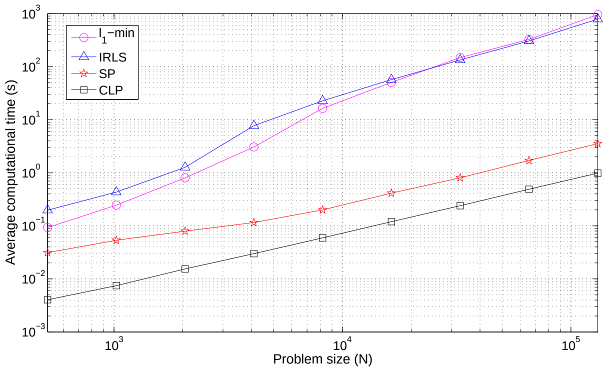


Fig. 8. Comparison of practical computational time ($\frac{K}{N} = \frac{3}{4}$).

From the two experiments, we can observe that our decoding algorithm outperforms comparative algorithms in both decoding efficiency and correcting ability. All of the comparative algorithms reconstruct the error vector in high dimensions. However, the decoding method converts the high-dimensional problem into groups of highly low-dimensional problems. This makes the method much more efficient than the comparative algorithms. Furthermore, the comparative algorithms do not find the sparsest solution to the underdetermined system directly (solving (2)) since it is computationally intractable in high dimensions. They solve this problem by some other indirect methods which are computationally efficient in high dimensions; however, these indirect methods make a discount in performance compared to direct methods. For the decoding method of the proposed code, since the highly low-dimensional underdetermined systems can be solved directly (solving (17) by the ESS method), the performance of the algorithm is much higher than those methods to which it was compared.

5. Application to Cryptography

Possible cryptographic applications of compressed sensing or compressed sensing-based real-number codes have been studied [34]–[36]. Unfortunately, these cryptographic schemes are unreliable since they can prevent only ciphertext-only attacks. When the codebook (the coding matrix) is known, the encrypted message can be easily deciphered [35]. In the cryptographic scheme proposed by Ashino *et al.*, a plaintext $\mathbf{x}$ is encoded by a linear code $\mathbf{F}$ and a sparse vector $\mathbf{e}$ is added to corrupt the codeword $\mathbf{Fx}$ by the sender. Then, the codebook $\mathbf{F}$ (or the secret key for generating $\mathbf{F}$) and the ciphertext $\mathbf{y} = \mathbf{Fx} + \mathbf{e}$ are sent to the receiver. Furthermore, on the receiver side, the recipient recovers the plaintext by means of ℓ_1-min. The problem is that, when $\mathbf{F}$ and $\mathbf{y}$ are received by the attacker, he can easily decipher the message by using ℓ_1-min or other existing algorithms. Therefore, the scheme is unreliable.

In this section, we demonstrate a highly reliable cryptographic scheme using the proposed code. Suppose that person A (the receiver) wants to get some secret data $\mathbf{x}$ (denoted as a vector) from person B (the sender). Our scheme can be described as follows: person A generates a PBD matrix $\mathbf{D}$ and a corresponding codebook $\mathbf{F}$ in one of the ways presented in Subsection 3.1; person A sends $\mathbf{F}$ to person B; person B encodes the message $\mathbf{x}$ with $\mathbf{F}$ and corrupts the codeword with a sparse error vector $\mathbf{y} = \mathbf{Fx} + \mathbf{e}$, where the corruption rate ρ is within the error correcting ability of CLP and beyond those of existing algorithms, e.g., $\rho = 4\%$ when $\frac{K}{N} = \frac{7}{8}$ (see Fig. 6); person B sends $\mathbf{y}$ to person A; person A recovers $\mathbf{x}$ by the CLP algorithm with the parity-check matrix $\mathbf{D}$. Note that the cipher can prevent attacks from the adversary even with knowledge of the codebook. This is because although the attacker can receive the codebook $\mathbf{F}$, it is impossible to recover the parity-check matrix $\mathbf{D}$.* Thus, the attacker cannot use CLP to decode the data and only traditional methods are available. However, the corruption rate is beyond the error correcting abilities of existing algorithms.

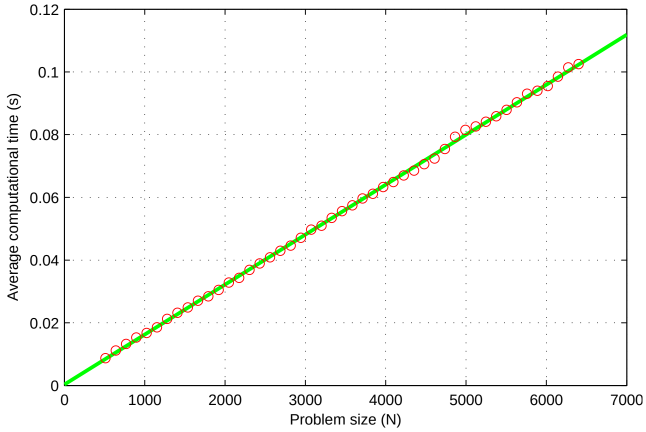


Fig. 9. Practical computational time of CLP versus problem size ($\frac{K}{N} = \frac{3}{4}$).

6. Conclusion and the Dense Small Noise Case

In this paper, we proposed an LDPC code over the real field based on compressed sensing. A LDPC matrix and a fast decoding algorithm were used. The proposed code has two important advantages over existing compressed sensing-based codes: high error correcting ability and decoding efficiency. We also conducted experiments to verify the proposed code. In addition, we applied the proposed code to cryptography; the proposed cryptographic scheme is reliable.

We discussed the decoding problem when sparse errors were added. In some other cases of interest, there are not only sparse errors that affect part of entries, but also small errors that affect all entries. Mathematically,

*Given a matrix $\mathbf{F}$, there are infinite matrices $\hat{\mathbf{D}}$ satisfying $\hat{\mathbf{D}}\mathbf{F} = \mathbf{0}$.

$$\mathbf{y} = \mathbf{Fx} + \mathbf{e} + \mathbf{c} \tag{18}$$

where **e** is a sparse vector of gross errors and **c** is a vector of small errors affecting all of the entries. In such a case, exact recovery is impossible and an approximation is desired. Therefore, CLP cannot be used for decoding. However, **x** can be approximated by traditional algorithms. For example, modify (4) with $p = 1$ to the following optimization problem to include a dense small noise allowance:

$$(P_{(1,\epsilon)}) \ : \ \min \|\mathbf{z}\|_1 \quad \text{subject to} \quad \|\mathbf{Dz} - \mathbf{s}\|_2 \leq \epsilon \tag{19}$$

where **D** is the corresponding PBD matrix that satisfies $\mathbf{DF} = \mathbf{0}$ and ϵ is a small parameter. Details of formulation of this optimization problem can be found in [37]. It can be cast as a convex quadratic program that can be solved by many standard approaches such as interior-point algorithms [7] and active-set methods. When **e** is sufficiently spare, a good approximation can be obtained by solving (19). Consequently, a good approximation of **x** can be obtained in the same way as without **c**. Note that PBD matrices can accelerate the decoding procedure and reduce memory consumption since they are highly sparse. A sparse matrix is very efficient to apply the matrix-vector multiplication and the memory consumption is low. For various algorithms either convex optimization or greedy methods, the resource requirement of matrix-vector multiplication is critical to their speeds and memory consumption.

Acknowledgements

This research was supported by the Fundamental Research Funds for the Central Universities and Zhejiang Provincial Natural Science Foundation of China (LQ13F020002 and LQ13F030003).

References

[1] WOLF, J. K. Redundancy, the discrete Fourier transform, and impulse noise cancellation. *IEEE Transactions on Communications*, 1983, vol. 31, no. 3, p. 458 - 461.

[2] MARSHALL, T. G. Codes for error correction based upon interpolation of real-number sequences. In *Proceedings of the 19th Asilomar Conference on Circuits, Systems, and Computers*. Pacific Grova (CA, USA), 1986, p. 202 - 206.

[3] SHIU, J., WU, J. L. Class of majority decodable real-number codes. *IEEE Transactions on Communications*, 1996, vol. 44, no. 3, p. 281 - 283.

[4] CHEN, Z., DONGARRA, J. Numerically stable real number codes based on random matrices. In *Proceeding of the 5th International Conference on Computational Science (ICCS2005)*. Atlanta (USA), 2005, p. 115 - 122.

[5] CANDÈS, E. J., TAO, T. Decoding by linear programming. *IEEE Transactions on Information Theory*, 2005, vol. 51, no. 2, p. 4203 - 4215.

[6] ZAYYANI, H., BABAIE-ZADEH, M., JUTTEN, C. Decoding real-field codes by an iterative expectation-maximization (em) algorithm. In *Proceeding of International Conference on Acoustics, Speech, Signal Processing*. Las Vegas (NV, USA), 2008, p. 3169 - 3172.

[7] CHEN, S. S., DONOHO, D. L., SAUNDERS, M. A. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 1999, vol. 20, p. 33 - 61.

[8] TIBSHIRANI, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 1996, vol. 58, p. 267 - 288.

[9] FIGUEIREDO, M. A. T., NOWAK, R. D., WRIGHT, S. J. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journal of Selected Topics in Signal Processing: Special Issue on Convex Optimization Methods for Signal Processing*, 2007, vol. 1, no. 4, p. 586 - 598.

[10] DONOHO, D. L., TSAIG, Y. Fast solution of ℓ^1 norm minimization problems when the solution may be sparse. *IEEE Transactions on Information Theory*, 2008, vol. 54, no. 11, p. 4789 - 4812.

[11] CHARTRAND, R. Exact reconstruction of sparse signals via nonconvex minimization. *IEEE Signal Processing Letters*, 2007, vol. 14, no. 10, p. 707 - 710.

[12] DAUBECHIES, I., DEVORE, R., FORNASIER, M., GUNTURK, C. S. Iteratively reweighted least squares minimization for sparse recovery. *Communications on Pure and Applied Mathematics*, 2010, vol. 63, , no. 1, p. 1 - 38.

[13] MALLAT, S., ZHANG, Z.. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 1993, vol. 41, no. 12, p. 397 - 415.

[14] DAVIS, G., MALLAT, S., ZHANG, Z. Adaptive time-frequency decompositions. *Optical Engineering*, 1994, vol. 33, no. 7, p. 183 - 191.

[15] NEEDELL, D., VERSHYNIN, R. Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit. *Foundations of Computational Mathematics*, 2009, vol. 9, p. 317 - 334.

[16] DONOHO, D. L., TSAIG, Y. Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit. *Preprint*, 2006.

[17] TROPP, J. A., GILBERT, A. C. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 2007, vol.53, no. 12, p. 4655 - 4666.

[18] NEEDELL, D., TROPP, J. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 2009, vol. 26, no. 3, p. 301 - 321.

[19] DAI, W., MILENKOVIC, O. Subspace pursuit for compressive sensing signal reconstruction. *IEEE Transactions on Information Theory*, 2009, vol. 55, no, 5, p. 2230 - 2249.

[20] BERINDE, R., INDYK, P., RUZIC, M. Practical near-optimal sparse recovery in the ℓ_1 norm. In *Proceedings 46th Annual Allerton Conference on Communication, Control, and Computing*. Illinois (USA), 2008, p. 198 - 205.

[21] BERINDE, R., INDYK, P. Sequential sparse matching pursuit. In *Proceedings of 47th Annual Allerton Conference on Communication, Control, and Computing*. Illinois (USA), 2009, p. 36 - 43.

[22] GILBERT, A., INDYK, P. Sparse recovery using sparse matrices. *Proceedings of the IEEE*, 2010, vol. 98, no. 6, p. 937 - 947.

[23] HE, Z., OGAWA, T., HASEYAMA, M. The simplest measurement matrix for compressed sensing of natural images. In *Proceedings of IEEE International Conference on Image Processing*. Hong Kong, 2010, p. 4301 - 4304.

[24] HE, Z., OGAWA, T., HASEYAMA, M. Linear time decoding of real-field codes over high error rate channels. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*. Prague (Czech Republic), 2011, p. 3172 - 3175.

[25] HE, Z., OGAWA, T., HASEYAMA, M. Cross low-dimension pursuit for sparse signal recovery from incomplete measurements based on permuted block diagonal matrix. *IEICE Transactions Fundamentals of Electronics, Communications and Computer Sciences*, 2011, vol. E94-A, p. 1793 - 1803.

[26] SARVOTHAM, S., BARON, D., BARANIUK, R. G. Compressed sensing reconstruction via belief propagation. *Technical report*. Rice University, Electrical and Computer Engineering Department, 2006.

[27] CANDÈS, E. J., ROMBERG, J., TAO, T. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 2006, vol. 52, no. 2, p.489 - 509.

[28] DONOHO. Compressed sensing. *IEEE Transactions on Information Theory*, 2006, vol. 52, p. 1289 - 1306.

[29] DONOHO, D. L., JOHNSTONE, I. M. Ideal spatial adaption via wavelet shrinkage. *Biometrika*, 1994, vol. 81, p. 425 - 455.

[30] SAUNDERS, M. A. *Pdco: Primal-Dual Interior Method for Convex Objectives*. [Online] Available at: http://www.stanford.edu/group/SOL/software/pdco.html.

[31] *Sparselab*. [Online] Available at: http://sparselab.stanford.edu.

[32] *Compressed Sensing Codes*. [Online] Available at: http://igorcarron.googlepages.com/cscodes.

[33] *Sparse Recovery Experiments with Sparse Matrices*. [Online] Available at: http://groups.csail.mit.edu/toc/sparse/wiki/index.php?title=Sparse_Recovery_Experiments.

[34] RACHLIN, Y., BARON, D. The secrecy of compressive sensing measurements. In *Proceedings of the 46th Annual Allerton Conference on Communication Control and Computing*. Illinois (USA), 2008, p. 813 - 817.

[35] ASHINO, R., NGUYEN-BA, T., VAILLANCOURT, R. Decoding low-dimensional linear codes by linear programming. *Canadian Applied Mathematics Quarterly*, 2008, vol. 16, p. 241 - 254.

[36] JUN LAI, M. On sparse solutions of underdetermined linear systems. *Preprint*, 2009.

[37] CANDÈS, E. J., RANDALL, P. A. Highly robust error correction by convex programming. *IEEE Transactions on Information Theory*, 2008, vol. 54, no. 7, p. 2829 - 2840.

About Authors . . .

Zaixing HE was born in Hunan, China. He received his B.Sc. and M.Sc. degrees in Mechanical Engineering from Zhejiang University, China in 2006 and 2008, respectively. He received his Ph.D. degree in 2012 from the Graduate School of Information Science and Technology, Hokkaido University, Japan. He is currently an assistant professor in the Department of Mechanical Engineering, Zhejiang University. His research interests include sparse recovery, sparse representation, compressed sensing and their applications to image processing, signal processing and pattern recognition.

Takahiro OGAWA was born in Hokkaido, Japan. He 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.

Miki HASEYAMA was born in Hokkaido, Japan. She 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.

Xinyue ZHAO was born in Shanxi, China. She received her M.S. degree in Mechanical Engineering from Zhejiang University, China in 2008, and her Ph.D degree in Graduate School of Information Science and Technology from Hokkaido University, Japan in 2012. She is currently an assistant professor in the Department of Mechanical Engineering, Zhejiang University, China. Her research interests include computer vision and image processing.

Shuyou ZHANG was born in Zhejiang, China. He received his M.S. degree in Mechanical Engineering and the Ph.D. degree in State Key Lab. Of CAD&CG from Zhejiang University, China, in 1991 and 1999, respectively. He is currently a professor in the Department of Mechanical Engineering, Zhejiang University, China. He is also the vice administer of Institute of Engineering & Computer Graphics in Zhejiang University, assistant director of Computer Graphics Professional Committee for China Engineering Graphic Society, member of Product Digital Design Professional Committee, and chairman of Zhejiang Engineering Graphic Society. His research interests include product digital design, design and stimulation for complex equipments, and engineering and computer graphics.