

Enhanced Joint Sparsity via Iterative Support Detection

Ya-Ru Fan^a, Yilun Wang^{a,b,c,*}, Ting-Zhu Huang^a

^a *School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, Sichuan, 611731 China*

^b *PrinceTechs LLC., Shenzhen, Guangdong, 518101 P. R. China.*

^c *Center for Applied Mathematics, Cornell University, Ithaca, NY, 14853, USA*

Abstract

Joint sparsity has attracted considerable attention in recent years in many fields including sparse signal recovery in compressive sensing, statistics, and machine learning. Traditional convex models with joint sparsity suffer from the suboptimal performance though enjoying tractable computation. In this paper, we propose a new non-convex joint sparsity model, and develop a corresponding multi-stage adaptive convex relaxation algorithm. This method extends the idea of iterative support detection (ISD) from the single vector estimation to the multi-vector estimation by considering the joint sparsity prior. We provide some preliminary theoretical analysis including convergence analysis and a sufficient recovery condition. Numerical experiments from both compressive sensing and multi-task feature learning show the better performance of the proposed method in comparison with several state-of-the-art alternatives. Moreover, we demonstrate that the extension of ISD from the single vector to multi-vector estimation is not trivial. While ISD doesn't well reconstruct the single channel sparse Bernoulli signal, it does achieve significantly improved performance when recovering the multi-channel sparse Bernoulli signal thanks to its ability of natural incorporation of the joint sparsity structure.

Keywords: Iterative support detection, joint sparsity, $\ell_{2,1}$ -norm minimization, non-convex optimization, compressive sensing, multi-task feature learning

*Corresponding author: yilun.wang@rice.edu (Yilun Wang)
E-mail addresses: yarufanfan@163.com (Ya-Ru Fan), yilun.wang@rice.edu (Yilun Wang), tingzhuhuang@126.com (Ting-Zhu Huang)

1. Introduction and Contributions

In the last decade, sparsity made it possible for us to reconstruct the high dimensional data with just few samples or measurements. The key of the sparse estimation problem is to stress the identification of the support, which denotes the indices of the nonzeros. If the support is known, the estimation of the sparse vectors reduces to a standard overdetermined linear inverse problem [24].

In order to enhance the estimation, many recent studies tend to consider the structure information of the solutions. For example, group sparsity structure [8, 14] widely appears in many applications [32, 43, 35, 31], where the components of solutions are likely to be either all zero or all nonzero in a group. By employing the grouping prior, ones aim to decrease the dispersion to facilitate recovering a much better solution. Here, we focus on joint sparsity, which is a special case of the group sparsity. Joint sparsity means that multiple unknown sparse vectors share a common unknown nonzero support set. Unlike the many group sparsity situations where the grouping information is unknown, the joint sparsity provides us the group information. In the following section, we will introduce the background of joint sparsity via two important applications, i.e. compressive sensing and multi-task feature learning.

In compressive sensing, joint sparsity aims to reconstruct unknown signals from m measurement vectors based on a common measurement matrix. This is also called the multiple measurement vectors (MMV) problem [7, 9, 36]. Given the observation vectors $b_j \in \mathbb{R}^m$ ($j = 1, \dots, \ell$) and a measurement matrix $A \in \mathbb{R}^{m \times n}$, we want to recovery the signal $x_j \in \mathbb{R}^n$ from the noisy underdetermined systems $b_j = Ax_j + e_j$, where $e_j \in \mathbb{R}^m$ is the noise. All the signal vectors x_1, \dots, x_ℓ share the sparsity pattern M , which implies the nonzero entries of x_1, \dots, x_ℓ almost appear on the same position. A common signal recovery model is

$$\min_{x_j} |M| \quad s.t. \quad b_j = Ax_j + e_j, \quad j = 1, \dots, \ell, \quad (1)$$

where $|M|$ is the cardinality of M [17]. In theory, we can recover the signals $X = [x_1, \dots, x_\ell]$ with rank $rank(X) = K$ if and only if

$$|M| < \frac{spark(A) - 1 + K}{2}, \quad (2)$$

where $\text{spark}(A)$ is the smallest set of linearly dependent columns of A [33]. Since problem (1) is NP-hard, it is usually relaxed with a convex alternative which is computationally efficient at the cost of more required measurements. Like ℓ_1 -norm being the convex relaxation of ℓ_0 -norm [2], the $\ell_{2,1}$ -norm is widely used as the convex replacement of $|M|$ as below:

$$\begin{aligned} \min_X \quad & \|X\|_{2,1} := \sum_{i=1}^n \|x^i\|_2 \\ \text{s.t.} \quad & b_j = Ax_j + e_j, j = 1, \dots, \ell, \end{aligned} \quad (3)$$

where $x^i \in \mathbb{R}^\ell$ and $x_j \in \mathbb{R}^n$ denote the i -th row and the j -th column of X , respectively.

Several fast algorithms have been proposed to solve problem (3) [40] such as greedy pursuit methods, iterative shrinkage algorithm [1] and alternating direction method (ADM) [26]. The greedy pursuit methods such as matching pursuit and orthogonal matching pursuit (OMP) [36] tend to require fewer computations but at the expense of slightly more measurements.

Multi-task learning has attracted much attention in machine learning [13, 39]. It aims to learn the shared information among related tasks in order for the improved performance than considering each learning task individually. Recently, multi-task feature learning based on the $\ell_{2,1}$ -norm regularization has been studied. An underlying property of the $\ell_{2,1}$ -norm regularization is that it urges multiple features from different tasks to share similar sparsity patterns [38]. Given ℓ learning tasks associated with training data $\{(A^1, b_1), \dots, (A^\ell, b_\ell)\}$, where $A^j \in \mathbb{R}^{m_j \times n}$ is the data matrix of the j -th task with each row as a sample and each column as a feature; $b_j \in \mathbb{R}^{m_j}$ is the response of the j -th task with biases e_j ; n is the number of features; and m_j is the number of samples for the j -th task, we would like to learn a weight matrix (sparsity pattern) $X = [x_1, \dots, x_\ell] \in \mathbb{R}^{n \times \ell}$ ($x_j \in \mathbb{R}^n$ consists of the weight vectors for ℓ linear predictive models $b_j = A^j x_j + e_j$) by solving the following optimization problem:

$$\begin{aligned} \min_X \quad & \|X\|_{2,1} := \sum_{i=1}^n \|x^i\|_2 \\ \text{s.t.} \quad & b_j = A^j x_j + e_j, j = 1, \dots, \ell, \end{aligned} \quad (4)$$

where $x^i \in \mathbb{R}^\ell$ and $x_j \in \mathbb{R}^n$ denote the i -th row and the j -th column of X , respectively. In this situation we assume that these different tasks share the same significant features, which leads to a joint sparsity problem.

The unconstrained formula corresponding to problems (3) and (4) can be written as the following unified form

$$\min_X L(X) + \rho \|X\|_{2,1}, \quad (5)$$

where $\rho > 0$ is the regularization parameter, and $L(X)$ is a smooth convex loss function such as the least square loss function or the logistic loss function. For example, $L(X) = \sum_{j=1}^{\ell} \|A\mathbf{x}_j - \mathbf{b}_j\|_2^2$ for problem (3), and $L(X) = \sum_{j=1}^{\ell} \frac{1}{\ell m_j} \|A^j \mathbf{x}_j - \mathbf{b}_j\|_2^2$ for problem (4).

While the convexity of $\ell_{2,1}$ -norm regularization provides computational efficiency, it also gives rise to the inherited bias issue. Similar with the ℓ_1 -norm regularized model which only achieves suboptimal recovery performance compared with the original cardinality regularized model from the theoretical viewpoint [6], $\ell_{2,1}$ -norm regularized model also only achieves suboptimal performance compared with the cardinality based model (1).

Recently, several computational advances have been made in the non-convex sparse regularization since its performance is better than that of the convex sparse regularization [12, 11]. For instance, for the single vector recovery, the non-convex ℓ_p -norm ($0 < p < 1$) based sparsity regularization usually obtains better performance than ℓ_1 -norm based sparsity regularization [5, 18, 21]. For the joint sparsity, the $\ell_{q,p}$ -norm is applied in a similar way, where $0 < p < 1$ and $q \geq 1$. The non-convex sparse regularization needs less strict recovery requirements and usually achieves a better performance than the convex alternatives. While there have existed many algorithms for solving the non-convex sparse regularized models, it is still a challenging problem to obtain the global optimal solution efficiently. The behavior of a local solution is hard to analyze and more seriously structural information of the solution is also hard to be incorporated into these algorithms.

Contribution : To achieve a better tradeoff between the recovery quality and the computational efficiency, we propose a non-convex joint sparsity regularized model and a multi-stage convex relaxation algorithm to solve the model. Motivated by the iterative support detection (ISD) [34] for sparse signal reconstruction, we extend the idea of ISD in our method from common sparsity to joint sparsity, from compressive sensing to feature learning. We present some new insights about why ISD achieves better performance than its convex alternatives, its key differences with other weighting based alternatives, and its flexibility in support detection implementation. Moreover, we provide the preliminary theoretical results including the convergence

analysis and a sufficient recovery condition.

More importantly, we discover some advantages of ISD which are not observed in the single vector recovery. In particular, for the single channel sparse signal estimation, ISD depends on the assumption of the fast decaying property of the nonzero components of the underlying true sparse signal and does not work for non-decaying signals. However, we empirically show that this assumption is no longer necessary for multi-channel sparse signal recovery, because the joint sparsity structure is adopted in the specific implementation of support detection. This implies that ISD might be naturally fused with the general structural sparsity, which leads to the enhanced performance.

Organization : The remainder of this paper is organized as follows: in Section 2, we propose a non-convex joint sparsity model and a corresponding algorithm based on ISD to solve the model. In Section 3, some preliminary theoretical results are presented. In Section 4, we show numerical experiments on both compressive sensing and multi-task feature learning to demonstrate the effectiveness of the proposed method. Section 5 is devoted to the conclusion and future works.

2. The Proposed Model and Corresponding Algorithm

We take the model (3) of compressive sensing as the example to show how ISD is extended to the joint sparsity model. Similar idea can be expanded to the multi-task feature learning model (4), as well as the unconstrained version (5).

2.1. Truncated Joint Sparsity Model

The proposed model based on the original joint sparsity model (3) is given as follows:

$$\begin{aligned} \min_{X(\omega)} \quad & \|X\|_{w,2,1} := \sum_{i=1}^n w_i \|x^i\|_2 \\ \text{s.t.} \quad & B = AX + E, \end{aligned} \tag{6}$$

where $B = [b_1, \dots, b_\ell]$ is the observation matrix, E is the noise matrix and $w = [w_1, w_2, \dots, w_n]$ is a weight parameter vector. Compared with the model (3), the main difference is the introduction of the weight vector w . Note that our model (6) prefers a specific 0-1 weighting scheme, i.e. w_i is either 0 or 1,

though most existing weighted models define weight as positive continuous real values.

Let T be the set of the indices of the nonzero rows of X , and the model (6) can be rewritten as

$$\min_{X(T)} \|X\|_{T,2,1} := \sum_{i \in T} \|x^i\|_2 \quad (TJS) \quad (7)$$

$$s.t. \quad B = AX + E.$$

We call it as truncated joint sparsity (TJS) model.

Intuitively, if we believe that x^i is true nonzero, it should be not forced to move closer to 0 and therefore we remove it from the regularization term, i.e. its corresponding w_i is set as 0. While many existing works assume that partial support information about underlying true sparse signal is already known [20, 30], the assumption may not hold in practice because w_i is not given beforehand. ISD, as a self-learning scheme, aims to gradually detect partial support information. It is a multi-stage alternative optimization procedure, which repeatedly executes the following two steps when applied to model (7):

- Step 1: we optimize x^i with w (or T) fixed (initially $\vec{1}$): this is a convex problem in terms of X .
- Step 2: we update w using the current X as reference via a support detection operation. The w will be used in the Step 1 of the next iteration.

Step 2 estimates the true nonzero rows from the rough intermediate estimated results of Step 1, and therefore called “support detection”. Our algorithm starts from initializing $w^{(0)} = \vec{1}$. In the first iteration, we obtain a solution $X^{(1)}$, which is the solution obtained by solving the plain $\ell_{2,1}$ model (3). Then we achieve the weight $w^{(1)}$ using $X^{(1)}$ as the reference. In the following iterations, we refine the intermediate solutions with the updated weights. In fact ISD decouples the estimation of w and X by an alternative scheme. We denote this multi-stage convex relaxation procedure as iterative support detection based joint sparsity algorithm (ISDJS).

2.2. Step 1: Solving Truncated Joint Sparsity Model

The $\ell_{2,1}$ -norm based joint sparsity model (3) leads to a convex optimization problem, and there are many efficient first-order algorithms to solve it in different application fields [19, 22, 29], which mostly try to make use of

the sparsity of the solutions in varied ways. In compressive sensing, one of the most popular algorithms is the ADM method [37, 8]. In [37], Yong et al. used the ADM technique to solve the ℓ_1 -norm based optimization problem for compressed sensing and developed the corresponding Matlab package termed Your ALgorithms for L_1 (YALL1). Furthermore, Deng et al. extended the YALL1 to the group version for solving the group sparse optimization with $\ell_{2,1}$ -norm regularization in [8]. For feature learning, Liu et al. proposed an efficient algorithm based on the Nesterovs method and the Euclidean projection in [25].

It is quite straightforward to extend these methods from $\ell_{2,1}$ -norm based models to truncated or weighted $\ell_{2,1}$ -norm regularized models. We take the YALL1 group algorithm for solving the plain $\ell_{2,1}$ regularized compressive sensing model as an example, and the resulted variant of the YALL1 group algorithm for the truncated joint sparsity model (6) is summarized in Algorithm 1, where $\Lambda_1 \in R^{n \times l}$, $\Lambda_2 \in R^{m \times l} > 0$ are multipliers in the ADM method, $\beta_1, \beta_2 > 0$ are penalty parameters, $\gamma_1, \gamma_2 > 0$ are step lengths, and $Z := X$ is an auxiliary variable.

Algorithm 1 Solving step 1 (inner loop)

1. Initialize $X \in R^{n \times l}$, $\Lambda_1 \in R^{n \times l}$, $\Lambda_2 \in R^{m \times l} > 0$, $\beta_1, \beta_2 > 0$ and $\gamma_1, \gamma_2 > 0$;
 2. While stopping criterion is not met, do
 - (a) $X \leftarrow (\beta_1 I + \beta_2 A^T A)^{-1} (\beta_1 Z - \Lambda_1 + \beta_2 A^T B + A^T \Lambda_2)$,
 - (b) $Z \leftarrow \text{Shrink} (X + \frac{1}{\beta_1} \Lambda_1, \frac{1}{\beta_1} w)$,
 - (c) $\Lambda_1 \leftarrow \Lambda_1 - \gamma_1 \beta_1 (Z - X)$,
 - (d) $\Lambda_2 \leftarrow \Lambda_2 - \gamma_2 \beta_2 (AX - B)$,
- where Λ_1, Λ_2 are multipliers, β_1, β_2 are penalty parameters, γ_1, γ_2 are step lengths.
-

The only modification of the extension of the YALL1 group algorithm from the common joint sparsity to the truncated joint sparsity is the step of updating Z , which is implemented by a shrinkage operator:

$$z^i = \text{Shrink}(r^i, \frac{1}{\beta_1} w) = \max\{\|r^i\|_2 - \frac{w_i}{\beta_1}, 0\} \frac{r^i}{\|r^i\|_2}, \quad i = 1, \dots, n, \quad (8)$$

where

$$r^i := x^i + \frac{1}{\beta_1} \lambda_1^i. \quad (9)$$

It is well known that $\ell_{2,1}$ -norm based model, as its counterpart of ℓ_1 -norm based model, suffers from its uniform shrinkage on all its components, i.e., it shrinks the true nonzero components as well, and reduces the sharpness of the solution or introduces bias to the final solution. In fact, the true nonzero components should not be shrunk in order to avoid the possibly caused bias. The truncated model can partially reduce this bias, because it corresponds to a selective shrinkage procedure where the weight value w_i is either 1 or 0. The true large nonzero components are expected to have the 0 weights and thus will not be shrunk. Surely we need to have some knowledge about the support information of the underlying true solution in order for the appropriate settings of weights. The support detection is implemented in Step 2, which will be introduced in the following subsection.

2.3. Step 2: Weight Determination Based on the Iterative Support Detection

Step 2 is a vital part of the proposed algorithm. As mentioned above, our strategy obtains the partial support information by itself, rather than given beforehand. Concretely, based on the recent intermediate result, we infer the indexes of nonzero rows of the underlying unknown true solution \bar{X} . Once we believe that certain rows are nonzero in the true solution \bar{X} , we set the corresponding weights to be zeros, and the rest weights are all ones.

Since the intermediate results are not very accurate, a robust way to detect the correct information about the true nonzero rows is required. Some extra prior knowledge of the underlying \bar{X} is needed in order for reliable support detection. Recall that in the single channel sparse signal recovery case, the nonzero components of the sparse or compressible signal are assumed to have a fast decaying distribution for the effectiveness of the threshold based support detection scheme (*threshold-ISR*). As for the multi-vector estimation problem, *threshold-ISR*, can also be applied in a similar way. At the s -th stage, we have an intermediate solution $X^{(s)}$. We aim to obtain some correct support information about the true \bar{X} based on $X^{(s)}$, i.e. identify some truly nonzero rows. The set of indices of detected nonzero rows based on *threshold-ISR* is similarly defined as follows:

$$I^{(s+1)} := \{i : |t_i^{(s)}| > \epsilon^{(s)}\}, s = 0, 1, 2, \dots, \quad (10)$$

where $t_i = \|x^i\|_2$. The support set $I^{(s)}$ is not necessarily increasing and nested, which means that all s may be not in $I^{(s)} \subset I^{(s+1)}$. Because $I^{(s)}$ is not required to be monotonic, support detection can remove previous wrong detections, which makes $I^{(s)}$ less sensitive to $\epsilon^{(s)}$.

For the choice of $\epsilon^{(s)}$, the “first significant jump” heuristic was proposed in the original implementation of ISD [34]. Specifically, one first sorts sequence $|t_{[i]}^{(s)}|$ in ascending order ($|t_{[i]}|$ denotes the i -th largest component of t by magnitude). The “first significant jump” scheme looks for the smallest i such that

$$|t_{[i+1]}^{(s)}| - |t_{[i]}^{(s)}| > \tau^{(s)}, \quad (11)$$

where $\tau^{(s)}$ is a data-dependent prescribed value to detect the big jump of this sequence. There are several simple and heuristic methods to define $\tau^{(s)}$. For example, one can set

$$\tau^{(s)} = m^{-1} \|t^{(s)}\|_{\infty}, \quad (12)$$

where m is the number of measurements. Then we set $\epsilon^{(s)} = |t_{[i]}^{(s)}|$. Intuitively, the “first significant jump” scheme works well since the true nonzero entries of $t^{(s)}$ are large in magnitude and small in number, while the false ones are large in number and small in magnitude. Therefore, the magnitudes of the true nonzero entries are spread out, while those of the false ones are clustered.

Recall that for sparse Bernoulli signal, where the nonzero components have exactly the same magnitude and do not have the fast decaying property, *threshold*-ISD fails to achieve a better performance than its convex alternative in the single channel recovery. Namely *threshold*-ISD works well only for the sparse signal whose nonzero components have the fast decaying magnitudes as presented in [34]. However, for the joint sparsity situation, *threshold*-ISD naturally incorporates this extra joint sparsity structure in the implementation of support detection and succeeds achieving a better recovery quality, as experiments illustrated below in Section 4. Here we give an intuitive explanation. Indeed, if we consider to each column of \bar{X} separately and perform threshold based support detection on each column individually, lack of fast decaying easily results in a significant number of wrong detections together with correct detections. This is the reason why for a single channel recovery, threshold based support detection will not help achieve a better recovery performance than the plain ℓ_1 model when sparse Bernoulli signals are recovered. However, for multiple sparse vectors which share the same sparsity structure, the detected true non-zero positions of each individual vector belong to the same subset (which contains all the true non-zero rows of the true solution \bar{X}) while the detected false nonzero positions of each individual vector might be quite different. Therefore, $|t_i^{(s)}|$ corresponding to the truly nonzero rows are much more likely to be significantly larger than

those corresponding to the false nonzero rows. Therefore, if we adopt the formula (10) and (11) to perform support detection, a relatively accurate support detection can be expected with an appropriate choice of threshold value. In other words, we use the shared sparsity structure in the support detection procedure, i.e. the formula (10) and (11).

We need to point out the difference of the 0-1 weighting scheme with a popular weighting method proposed in [2] where the weight is determined as follows:

$$w_i^{(s)} = \frac{1}{|x_i^{(s)}| + \xi},$$

where the choice of $\xi > 0$ is a key for the performance of its corresponding algorithm. If the ξ is too small, then too much noisy information is taken into consideration. If the ξ is too big, much of the information about the true nonzero elements is filtered out. An appropriate way might be to gradually decrease ξ from a large number to a small one as s increases. However, the determination of ξ is hard. While we know ξ should be data-adaptive, a feasible practical scheme to determine ξ is not easy to design. On the contrary, the Step 2 is much easier to obtain a data-adaptive scheme. The overall procedure of ISDJS is summarized in Algorithm 2.

In addition, an advantage of the proposed method is that the implementation of support detection is very flexible. Besides the above heuristic (12), one can try other alternative ideas. For example, one dimensional edge detection methods such as those proposed in [41, 3] can also be adopted to detect the “first significant jump”, i.e. determine an appropriate $\tau^{(\ell)}$ value for (11). In the following numerical experiments, while the heuristic rule (12) will be mostly adopted.

Algorithm 2 The ISDJS algorithm (outer loop)

Input: measurement matrix A and observation matrix B ,

1. Set the iteration number $s \leftarrow 1$ and initialize $w^{(0)} = \vec{1}$;

2. While stopping condition is not met, do

(a) $X^{(s)} \leftarrow$ solve problem (7) via Algorithm 1 for $w = w^{(s-1)}$;

(b) $w^{(s)} \leftarrow T^{(s)} := (I^{(s)})^C = \{1, 2, \dots, n\} \setminus I^{(s)}$;

 where $I^{(s)} \leftarrow$ compute approach (10) for $X = X^{(s)}$;

(c) $s \leftarrow s + 1$.

The main computational cost of Algorithm 2 stems from computing the X in Algorithm 1. We only need to compute the matrix inverse or do the matrix

factorization once, whose complexity per iteration is $\mathcal{O}(mn)$ [8]. Although ISDJS is a multi-stage procedure, the iteration number is small empirically, like around 4 as the following numerical experiments presented in Section 4. Thus, the total complexity of the proposed method is approximately $\mathcal{O}(mn)$.

3. Preliminary Theoretical Analysis

Some preliminary theoretical analysis including convergence analysis and a sufficient recovery condition are presented, for the proposed TJS model and the ISDJS algorithm.

3.1. Convergence Analysis

We assume that $A \in \mathbb{R}^{m \times n}$ and $A^j \in \mathbb{R}^{m_j \times n}$ follow the continuous probability distribution. This convergence analysis applies to both the compressive sensing and the multi-task feature learning situations. In fact, ISDJS only runs a few steps and s is smaller than 5 in general, and these steps can be considered to determine a proper threshold value ϵ for support detection (10). For simplicity of proof, when considering the convergence analysis where s goes to infinity, we assume that the threshold value $\epsilon^{(s)}$ used in support detection (10) is fixed as $\bar{\epsilon}$ when $s > \bar{s}$ ($\bar{s} = 5$, for example). This assumption is slightly biased from the truth, because the threshold value could keep changing as the iteration of ISDJS proceeds. However, it is also reasonable and acceptable to certain degree because in practice, ISDJS only runs a very limited number of steps, i.e. ISDJS will stop when s is not very big. Moreover, even as s goes to infinity, the threshold value will not change much empirically.

The main idea of the following proof refers to [15]. However, unlike [15] where a truncated $\ell_{1,1}$ model is considered, we consider a truncated $\ell_{2,1}$ model. A locally linear approximation is presented as a preparation for the following convergence proof.

First, for any $\epsilon > 0$, we consider the following unstrained weighted $\ell_{2,1}$ regularized model corresponding to model (5):

$$\min_X L(X) + \rho \sum_{i=1}^n w_i \|\mathbf{x}^i\|_2 \quad (13)$$

where $L(X)$ is a quadratic cost function of X and $\rho (> 0)$ is a parameter. In terms of ISDJS algorithm we denote $w_i = T(\|\mathbf{x}^i\|_2 < \epsilon)$ and $T(\cdot)$ denotes

the $\{0, 1\}$ -valued indicator function, which is consistent with $T = I^C$ in Algorithm 2.

The solution of problem (13) is equivalent to the solution of the following problem:

$$\min_X L(X) + \rho \sum_{i=1}^n \min(\|\mathbf{x}^i\|_2, \epsilon) \quad (14)$$

We assume that $\epsilon^{(s)}$ is kept unchanged after several stages of ISDJS. That is to say when s is big enough, $\epsilon^{(s)} \doteq \bar{\epsilon}$ in Algorithm 2.

Second, we define two auxiliary functions:

$$\mathbf{h} : R^{n \times l} \mapsto R_+^n, \mathbf{h}(X) = [\|\mathbf{x}^1\|_2, \|\mathbf{x}^2\|_2, \dots, \|\mathbf{x}^n\|_2]^T, \quad (15)$$

$$g_\epsilon : R_+^n \mapsto R_+, g_\epsilon(\mathbf{u}) = \sum_{i=1}^n \min(u_i, \epsilon). \quad (16)$$

Note that $g_\epsilon(\cdot)$ is a concave function [15]. (When s is large enough, $g_{\epsilon^{(s)}} = g_{\bar{\epsilon}}$.) We know that a vector $\mathbf{z} \in R^n$ is a sub-gradient of g at $\mathbf{v} \in R_+^n$, if for all vector $\mathbf{u} \in R_+^n$, the following inequality holds:

$$g_\epsilon(\mathbf{u}) \leq g_\epsilon(\mathbf{v}) + \langle \mathbf{z}, \mathbf{u} - \mathbf{v} \rangle, \quad (17)$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. Based on the functions defined above, problem (14) is equivalent to the following problem

$$\min_X L(X) + \rho g_\epsilon(\mathbf{h}(X)). \quad (18)$$

Then we can obtain an upper bound of $g_\epsilon(\mathbf{h}(X))$ using a locally linear approximation at $\mathbf{h}(X^{(s)})$ based on the inequality (17):

$$g_\epsilon(\mathbf{h}(X)) \leq g_{\epsilon^{(s)}}(\mathbf{h}(X^{(s)})) + \langle \mathbf{z}^{(s)}, \mathbf{h}(X) - \mathbf{h}(X^{(s)}) \rangle, \quad (19)$$

where $\mathbf{z}^{(s)} = [T(\|(\mathbf{x}^{(s)})^1\|_2 < \epsilon), \dots, T(\|(\mathbf{x}^{(s)})^n\|_2 < \epsilon)]^T$ is a sub-gradient of $g_\epsilon(\mathbf{u})$ at $\mathbf{u} = \mathbf{h}(X^{(s)})$. Furthermore, for $\forall \mathbf{h}(X)$ we obtain an upper bound of the optimization problem (14):

$$L(X) + \rho g_\epsilon(\mathbf{h}(X)) \leq L(X) + \rho g_{\epsilon^{(s)}}(\mathbf{h}(X^{(s)})) + \rho \langle \mathbf{z}^{(s)}, \mathbf{h}(X) - \mathbf{h}(X^{(s)}) \rangle. \quad (20)$$

Since ρ and $h(X^{(s)})$ are constant with respect to X , we have

$$X^{(s+1)} = \arg \min_X L(X) + \rho g_{\epsilon^{(s)}}(\mathbf{h}(X^{(s)})) + \rho \langle \mathbf{z}^{(s)}, \mathbf{h}(X) - \mathbf{h}(X^{(s)}) \rangle$$

$$= \arg \min_X L(X) + \rho(\mathbf{z}^{(s)})^T \mathbf{h}(X), \quad (21)$$

which corresponds to the step 2 (a) of the Algorithm 2. Therefore, we intuitively consider that the Algorithm 2 minimizes an upper bound in each step.

Theorem 1. *Let $\bar{f}(X) = L(X) + \rho \sum_{i=1}^n \bar{w}_i \|\mathbf{x}^i\|_2$, where $\bar{w}_i = T(\|\mathbf{x}^i\|_2 < \bar{\epsilon})$. The sequence $\{\bar{f}(X^{(s)})\}$ is decreasing and convergent.*

Proof. When s is big enough, $\epsilon^{(s)} = \bar{\epsilon}$, and $g_{\epsilon^{(s)}} = g_{\bar{\epsilon}}$. We firstly verify the objective function value in (14) decreases monotonically based on locally linear approximation, when s is big enough, as follows:

$$\begin{aligned} L(X^{(s+1)}) + \rho g_{\bar{\epsilon}}(\mathbf{h}(X^{(s+1)})) &\leq L(X^{(s+1)}) + \rho g_{\bar{\epsilon}}(\mathbf{h}(X^{(s)})) + \rho \langle \mathbf{z}^{(s)}, \mathbf{h}(X^{(s+1)}) - \mathbf{h}(X^{(s)}) \rangle \\ &\leq L(X^{(s)}) + \rho g_{\bar{\epsilon}}(\mathbf{h}(X^{(s)})) + \rho \langle \mathbf{z}^{(s)}, \mathbf{h}(X^{(s)}) - \mathbf{h}(X^{(s)}) \rangle \\ &= L(X^{(s)}) + \rho g_{\bar{\epsilon}}(\mathbf{h}(X^{(s)})), \end{aligned}$$

where the first inequality is based on equation (20) and the second inequality follows (21), i.e., $X^{(s+1)}$ is a minimizer of the right hand side of equation (20). Then we have

$$\bar{f}(X^{(s+1)}) \leq \bar{f}(X^{(s)}).$$

Obviously, we observe that

$$\bar{f}(X) = L(X) + \rho \sum_{i=1}^n \bar{w}_i \|\mathbf{x}^i\|_2 \geq 0.$$

Thus $\{\bar{f}(X^{(s)})\}$ is bounded below. Therefore $\{\bar{f}(X^{(s)})\}$ is convergent. ■

3.2. A Sufficient Recovery Condition of TJS Model

Here we discuss the noiseless compressive sensing model (3). We first review the truncated null space property (t-NSP) proposed in [34], which is a generalization of the null space property (NSP).

Definition 1. *Matrix A satisfies the t -NSP of order L for $\gamma > 0$ and $0 < t \leq n$ if*

$$\|\eta_S\|_1 \leq \gamma \|\eta_{(T \cap S^c)}\|_1 \quad (22)$$

holds for all sets $T \subset \{1, \dots, n\}$ with $|T| = t$, all subsets $S \subset T$ with $|S| \leq L$, and all $\eta \in \mathcal{N}(A)$ — the null space of A .

For simplicity, we use t -NSP(t, L, γ) to denote the t -NSP of order L for γ and t , and use $\bar{\gamma}$ to replace γ and write t -NSP($t, L, \bar{\gamma}$) if $\bar{\gamma}$ is the infimum of all the feasible γ satisfying (22).

For the single channel sparse signal recovery problem

$$\begin{aligned} \min_x \quad & \|x_T\|_1 \\ \text{s.t.} \quad & b = Ax \end{aligned} \tag{23}$$

Theorem 3.2 in [34] has shown that if $A \in \mathbb{R}^{m \times n}$ ($m < n$) satisfies the t-NSP, then the true signal \bar{x} is the unique solution of model (23) if

$$\|\bar{x}_T\|_0 < k(d), \tag{24}$$

where $k(d) := c \frac{m-d}{1+\log(\frac{n-d}{m-d})}$, $d = n - t = n - |T|$, and $c > 0$ is absolute constant independent of the dimensions m , n and d . Let $d_c = |I \cap \text{supp}(\bar{t})|$ stand for the number of correct detections, and inequality (24) is equivalent to

$$\|\bar{x}\|_0 < k(d) + d_c, \tag{25}$$

due to $\|\bar{x}\|_0 = \|\bar{x}_T\|_0 + d_c$. In light of (25), to compare the common ℓ_1 model with truncated ℓ_1 model (23), we shall compare $k(0)$ with $k(d) + d_c$. In [34], the authors have shown that if there are enough correct detections (i.e., d_c/d is sufficiently large), then we get $k(0) < k(d) + d_c$. That is to say, the truncated ℓ_1 model might be able to recovery more nonzeros than the common ℓ_1 model. We extend the above conclusion about the advantage of the truncated ℓ_1 model over the common ℓ_1 model from the single vector case to the multi-vector recovery case, i.e. the joint sparsity case. We will see that this kind of extension is feasible thanks to the theorem proved in [23], which is revisited as below.

Theorem 1.3 in [23] has proved the following two statements are equivalent.

(a) for all vectors $(\mathbf{u}^1, \dots, \mathbf{u}^r) \in (\mathcal{N}(A))^r \setminus \{(0, 0, \dots, 0)\}$

$$\sum_{j \in S} (\sqrt{u_{1,j}^2 + \dots + u_{r,j}^2}) < \sum_{j \in S^c} (\sqrt{u_{1,j}^2 + \dots + u_{r,j}^2}), \tag{26}$$

(b) for all vector $\mathbf{v} \in \mathcal{N}(A)$ with $v \neq 0$

$$\sum_{j \in S} |v_j| < \sum_{j \in S^c} |v_j|, \mathbf{v} = (v_1, \dots, v_n)^T \in \mathbb{R}^n. \tag{27}$$

They hold for all index sets $S \subset \{1, 2, \dots, n\}$ with $|S| \leq L$, where $\mathcal{N}(A)$ stands for the null space of A and S^c is the complement set of S . Namely,

the null space property of multiple systems of linear equations is equivalent to the null space property for the common ℓ_1 minimization subject to a single linear system.

During their proof of this equivalence, they only make use of the fact $S \cap S^c = \emptyset$. So we naturally have the following equivalence [23]:

(c) for all vectors $(\mathbf{u}^1, \dots, \mathbf{u}^r) \in (\mathcal{N}(A))^r \setminus \{(0, 0, \dots, 0)\}$

$$\sum_{j \in S} (\sqrt{u_{1,j}^2 + \dots + u_{r,j}^2}) < \sum_{j \in T \cap S^c} (\sqrt{u_{1,j}^2 + \dots + u_{r,j}^2}), \quad (28)$$

(d) for all vector $\mathbf{v} \in \mathcal{N}(A)$ with $v \neq 0$

$$\sum_{j \in S} |v_j| < \sum_{j \in T \cap S^c} |v_j|, \mathbf{v} = (v_1, \dots, v_n)^T \in R^n. \quad (29)$$

They hold for all index sets $S \subset \{1, 2, \dots, n\}$ with $|S| \leq L$. Thus, we similarly have the equivalence of the t-NSP of multiple systems of linear equations with the t-NSP for the common ℓ_1 minimization subject to a single linear system .

Therefore, the better recovery performance of the truncated ℓ_1 model over the common ℓ_1 model can be extended to our specific joint sparsity case, i.e. the multiple-vector compressive sensing problem. In other words, if there are enough correct detections, the truncated joint sparsity model (6) can recover more nonzero rows than the common $\ell_{2,1}$ model, based on the above simple and intuitive analysis. However, for multi-task learning problem, where different A_j exists, the theoretical judgement of the truncated ℓ_1 model over the common ℓ_1 model needs further investigation.

4. Numerical Experiments

We show some numerical experiments to demonstrate the better performance of the proposed ISDJS in comparison with several state-of-the-art algorithms. For compressive sensing, the YALL1 group [8], SOMP [33] and p-threshold [16] algorithms are compared. For multi-task feature learning, ISDJS is compared with the baseline algorithm for the common $\ell_{2,1}$ regularized model, whose baseline algorithm proposed in [25] is implemented in the software ‘‘MALSAR’’ [42]. We mainly focus on the recovery rate and accuracy. Due to the channel number has a great influence on recovery rate of joint sparsity, we provide several different channel number settings. In

addition, we also test the robustness of the competing approaches in different noise levels. The synthetic experiments and two realistic experiments on collaborative spectrum sensing [27] and multi-task feature learning, verify the effectiveness of ISDJS.

4.1. Parameter Settings of Synthetic Examples

Two synthetic examples are presented for compressive sensing. The true jointly sparse solution $\bar{X} \in R^{n \times \ell}$ is generated by randomly selecting k rows as nonzero rows whose entries follow the i.i.d Gaussian and Bernoulli distribution in test 1 and test 2, respectively. The rest rows of \bar{X} are set as zeros. Randomized partial Walsh-Hadamard transform matrix is utilized as the measurement matrix $A \in R^{m \times n}$ in compressive sensing, because it is suitable for large-scale computation and has the property $AA^T = I$. For SOMP and p-thresholding algorithms, we use their default parameter settings in [33, 16]. We set the parameters of the YALL1 group algorithm and ISDJS referring to [8] as follows: $\beta_1 = 0.3 / \frac{1}{m\ell} \sum_{i=1}^m \sum_{j=1}^{\ell} |b_{ij}|$, $\beta_2 = 3 / \frac{1}{m\ell} \sum_{i=1}^m \sum_{j=1}^{\ell} |b_{ij}|$ (b_{ij} is the entries of matrix $B \in R^{m \times \ell}$) and $\gamma_1 = \gamma_2 = 1.618$. All involved algorithms are terminated when

$$\frac{\|t^{(k+1)} - t^{(k)}\|_2}{\|t^{(k+1)}\|_2} \leq \varepsilon. \quad (30)$$

For SOMP, p-threshold and YALL1 group algorithms, ε is set as 10^{-6} . As for ISDJS, in the first few outer loops, we only want to get an rough estimate of the support information of X , thus we just set a relatively loose tolerance such as $\varepsilon = 10^{-2}$. But in the last iterations, ε is also set as 10^{-6} for fair comparison. In all experiments, ISDJS runs no more than 5 outer loops.

The empirical recovery performance of all test algorithms in general becomes better as the number of channels gradually increases, though to varying degrees [10]. Therefore, different channel number settings are tried in our experiments, for example, $L = 1, 2, 4, 8, 16$, respectively. We also try different sparsity levels varying from $k = 80$ to 160, while fixing $n = 1024$, $m = 256$ in all tests excluding Figs 1, 2, 6, 7. The experimental results corresponding to compressive sensing are usually an average of 100 runs due to the involved randomness in the generation of A and \bar{X} .

4.1.1. Test 1: Compressive recovery of joint sparse Gaussian signals

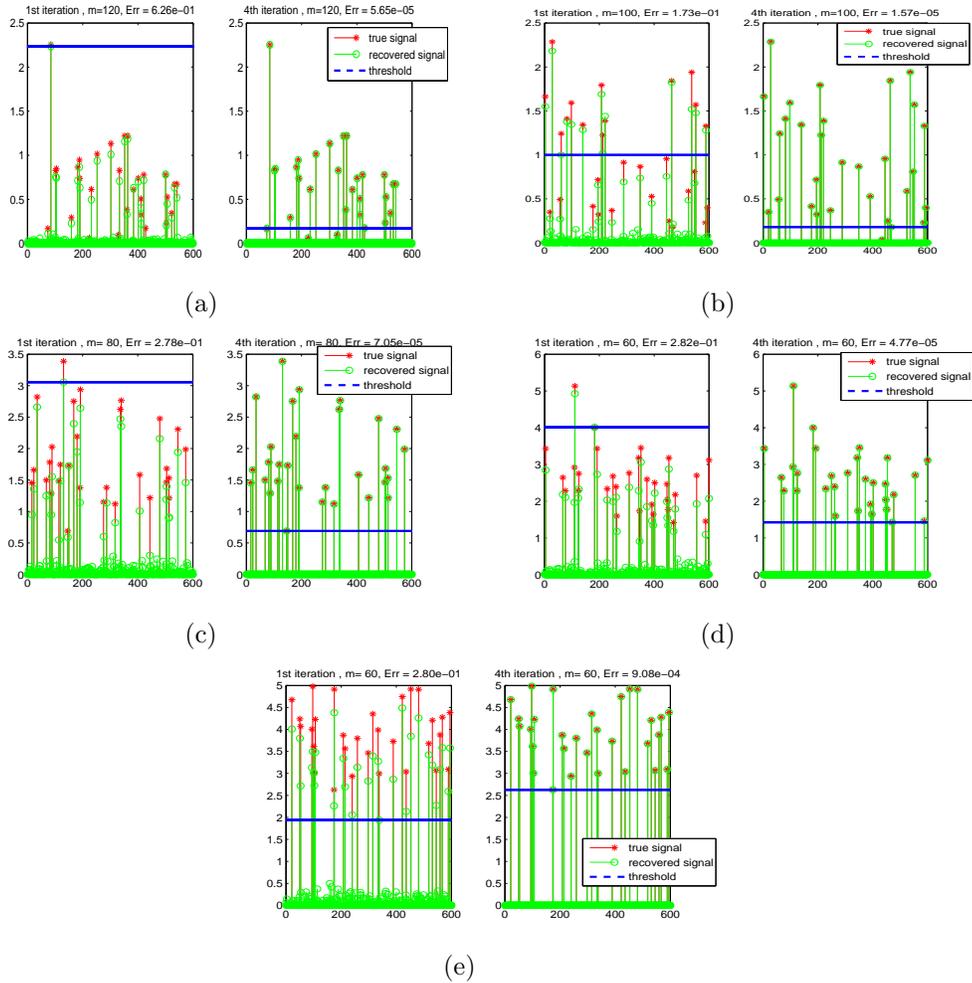
We perform a synthetic compressive sensing example to demonstrate some key aspects of ISDJS including the effectiveness of threshold based support

detection, the effect of the channel number. We also illustrate how ISDJS produces gradually improved intermediate solutions starting from the low quality initial point which is the solution of the convex alternative. We generate a sparse signal $\bar{X} \in R^{600 \times L}$ with $k = 30$ nonzero rows. The results of ISDJS in the first iteration and the fourth iteration for different channel numbers settings are depicted in the Fig 1, where we set \bar{t} (a vector of 2-norm of each row of \bar{X}) on behalf of the true signal \bar{X} and t (a vector of 2-norm of each row of X from ISDJS) on behalf of the recovered signal. We use the quadruplet “(Total, Detected, Correct, False)” and “Err” defined below to measure the accuracy of support detection.

- (Total, Detected, Correct, False):
 - Total: the number of total nonzero rows of the true signal \bar{X} ;
 - Detected: the number of detected nonzero rows, equal to $|I| = (Correct) + (False)$;
 - Correct: the number of correctly detected nonzero rows, i.e., $|I \cap \{i : \bar{t}^i \neq 0\}|$;
 - False: the number of falsely detected nonzero rows, i.e., $|I \cap \{i : \bar{t}^i = 0\}|$.
- Err: the relative error $\|X - \bar{X}\|_2 / \|\bar{X}\|_2$.

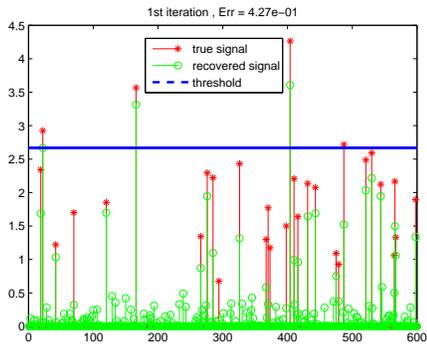
From Fig 1 (a), we can see that the output of the first iteration of ISDJS, which is the solution of the common convex $\ell_{2,1}$ -norm regularized model (3), is not good. Nevertheless the output of the fourth iteration of ISDJS could well match the true signal with a very small relative error. We also see that ISDJS is insensitive to a small number of false detections and has an attractive self-correction capacity. In particular, while it is difficult for the common $\ell_{2,1}$ model (3) to recover a signal with $k = 30$ nonzero entries from $m = 60$ measurements, ISDJS can finally return a satisfying result, as presented in Fig 1 (e). Note that when the measurements m decrease, ISDJS still returns a better result with channel numbers L increasing.

In order to better understand ISDJS, we show each outer iteration of it in Fig 2, by taking an example of $L = 4$ and $m = 80$. From the Fig 2 (a), ISDJS in the first iteration (i.e. YALL1 group algorithm), finds very few positions of correct nonzero rows and has a large relative error. However, a half positions of correct nonzero rows could be detected in the next iteration as exhibited in Fig 2 (b), and a significantly improved recovery is obtained, shown in Fig 2 (c). In the third iteration, our algorithm has already correctly detected the most nonzero positions, and therefore a good enough solution is obtained as illustrated in Fig 2 (d).

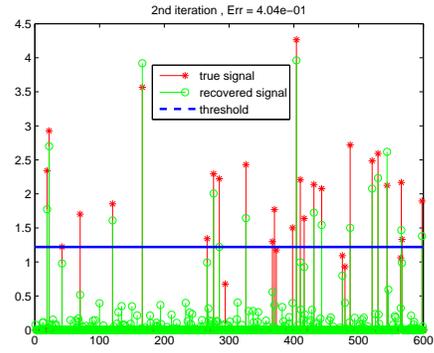


L-Iteration	Nonzeros				Relative error
	Total true	Detected	Correct	False	
1-1	30	37	29	8	6.26e-01
1-4	30	30	30	0	5.65e-05
2-1	30	34	28	6	3.73e-01
2-4	30	30	30	0	7.57e-05
4-1	30	38	30	8	2.78e-01
4-4	30	30	30	0	7.05e-05
8-1	30	34	30	4	2.82e-01
8-4	30	30	30	0	4.77e-05
16-1	30	33	30	3	2.80e-01
16-4	30	30	30	0	9.08e-05

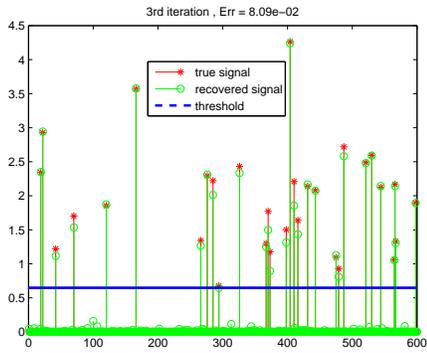
Fig 1. Compare the true Gaussian signals and recovered signals from ISDJS in different channels, where the two parts in each subplot are the results in the first iteration and the fourth iteration respectively. (a) $L=1, m=120$, (b) $L=2, m=100$, (c) $L=4, m=80$, (d) $L=8, m=60$, (e) $L=16, m=60$.



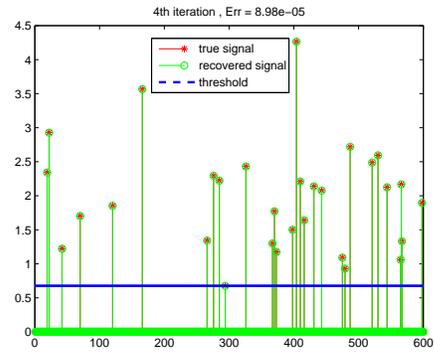
(a)



(b)



(c)



(d)

Iteration	Nonzeros				Relative error
	Total true	Detected	Correct	False	
1	30	41	27	14	4.27e-01
2	30	40	28	12	4.04e-01
3	30	30	30	0	8.09e-02
4	30	30	30	0	8.98e-05

Fig 2. Compare the true Gaussian signals and recovered signals obtained by ISDJS in each iteration with $L=4$ channels.

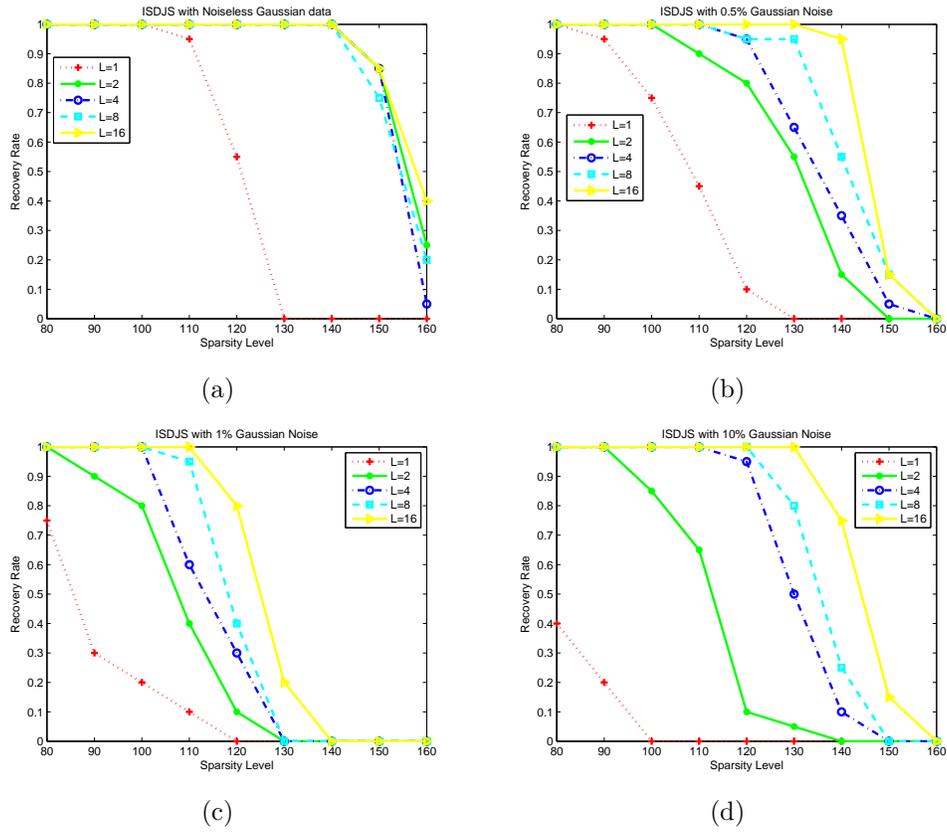


Fig 3. Compare the recovery rate of ISDJS with $L=1, 2, 4, 8, 16$ in different noise levels for Gaussian signals, (a)noiseless, (b)0.5% noise, (c)1% noise, (d)10% noise.

Fig 3 shows the performance of ISDJS with different channels in four different noise levels to verify its robustness. The proposed algorithm performs better for the multi-channel sparse signal recovery than the single channel sparse signal recovery even in the high level noise.

In Fig 4 and Fig 5, we compare the recovery rates and relative errors of all test algorithms, for noiseless case and noisy case (added Gaussian noise with standard variance 0.5%), respectively, when the channel number varies. We can see that ISDJS outperforms other algorithms in all involved different channels. While the common $\ell_{2,1}$ model behaviors worse than the SOMP in the cases of $L = 4$ and $L = 8$, ISDJS which applies ISD to the common $\ell_{2,1}$ model, works better than SOMP.

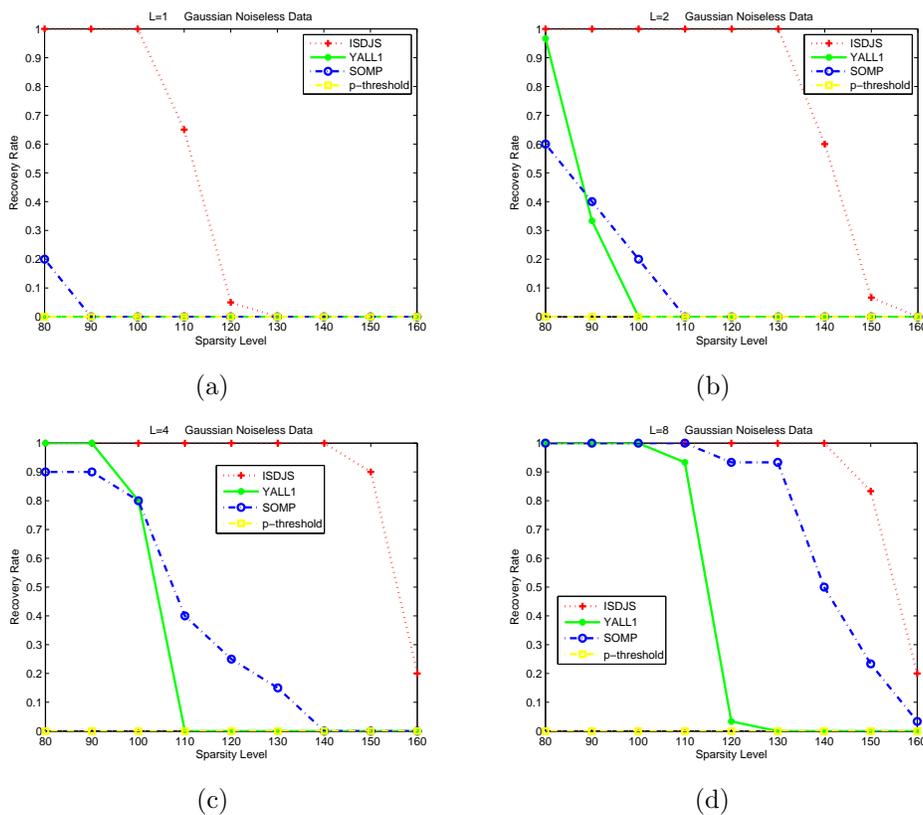
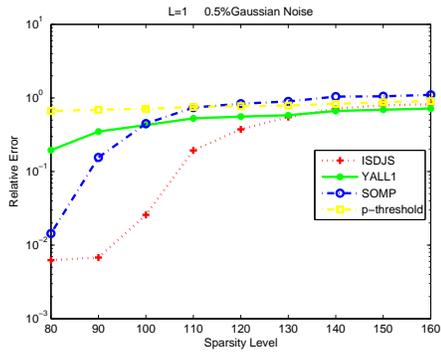
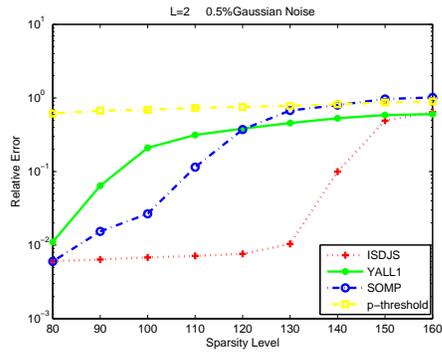


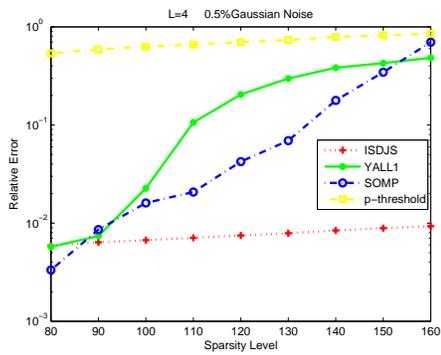
Fig 4. Compare the recovery rate of four algorithms in different channels for noiseless Gaussian signals, (a)L=1, (b)L=2, (c)L=4, (d)L=8.



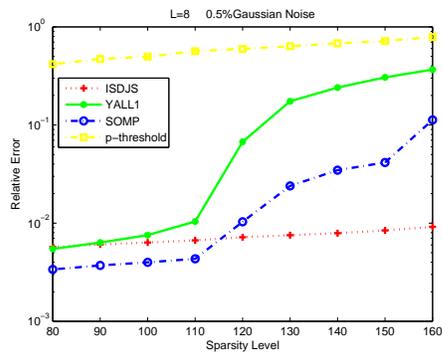
(a)



(b)

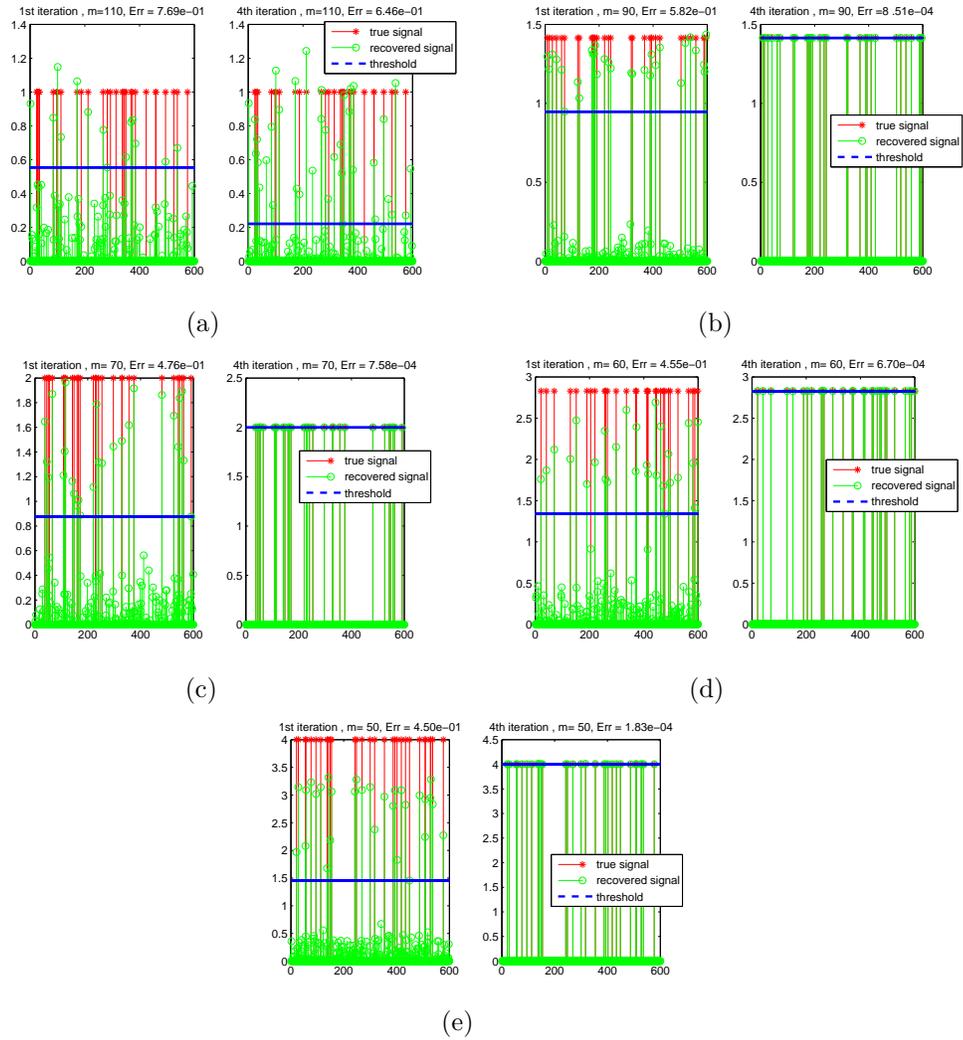


(c)



(d)

Fig 5. Compare relative error of four algorithms in different channels with 0.5% noise for Gaussian signals, (a)L=1, (b)L=2, (c)L=4, (d)L=8.



L-Iteration	Nonzeros				Relative error
	Total true	Detected	Correct	False	
1-1	30	33	27	6	7.69e-01
1-4	30	36	30	6	6.46e-01
2-1	30	32	30	2	5.82e-01
2-4	30	30	30	0	8.51e-04
4-1	30	35	30	5	4.76e-01
4-4	30	30	30	0	7.58e-04
8-1	30	35	30	5	4.56e-01
8-4	30	30	30	0	6.70e-04
16-1	30	32	30	2	4.50e-01
16-4	30	30	30	0	1.83e-04

Fig 6. Compare the true Bernoulli signals and recovered signals obtained by ISDJS in different channels, where the two components in each subplot are the results in the first iteration and the fourth iteration respectively. (a)L=1, m=110, (b)L=2, m=90, (c)L=4, m=70, (d)L=8, m=60, (e)L=16, m=50.

4.1.2. Test 2: Compressive recovery of joint sparse Bernoulli signals

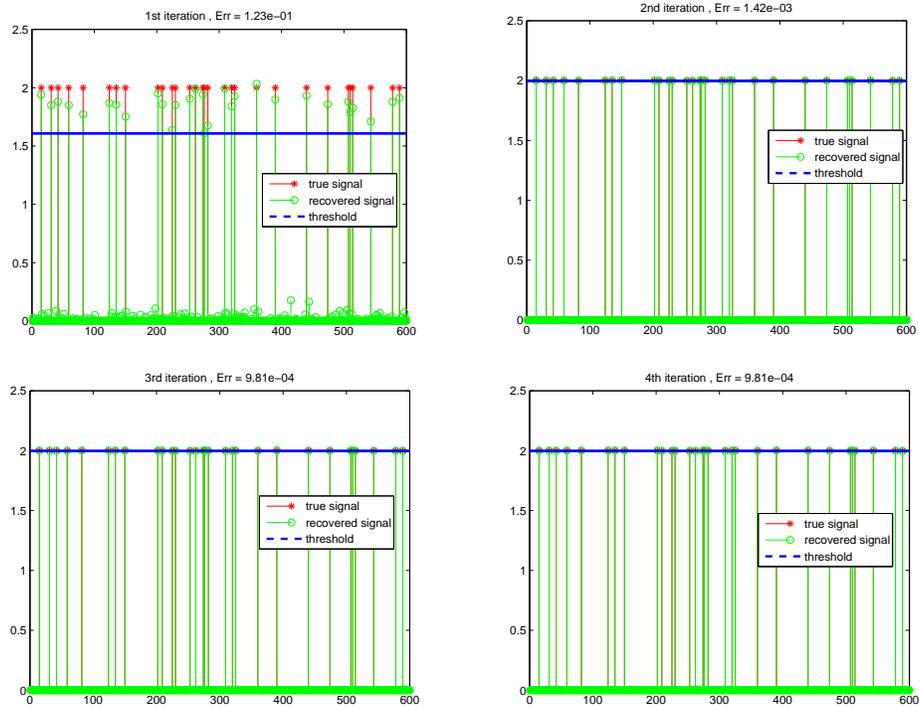
We show a surprising performance of ISD with joint sparsity. In [34] for the single channel signal recovery, the *threshold*-ISD works well for signals with a fast decaying property of nonzero entries such as sparse Gaussian signals and certain power-law decaying signals. However, it does not work for signals that decay slowly or have no decay at all such as sparse Bernoulli signals, since the threshold based support detection fail to accurately distinguish true nonzero components according to the intermediate recovery results.

As Fig 6 (a) presented, the support detection is poor and fails to correctly detect the true nonzero components in the single channel signal recovery. Nevertheless, Fig 6 (b) shows the threshold based support detection can accurately find some true nonzero components, even just for $L = 2$. Then, in Figs 6 (c), (d) and (e), threshold based support detection works well. Finally, the ISDJS achieves quite good recovery performance, which suggests that ISD is able to achieve relatively accurate support detection even for Bernoulli signals by incorporating the joint sparsity structure, as briefly explained in Section 2.3. The table below Fig 6 displays that the support detection works well as the iteration proceeds.

In Fig 7, we exhibit the performance of ISDJS for sparse Bernoulli signals of each outer iteration by taking $L = 4$ and $m = 70$ as an example. It is possible to add more iterations but four iterations are enough for ISDJS to return an accurate solution. In Fig 8, we show the recovery rates of ISDJS with different channel number settings under four noise levels. The ISDJS consistently performs much better in multichannel cases rather than the single channel situation. Moreover, the ISDJS keeps robust in different noise levels.

We plot the recovery rate of ISDJS in comparison with other three algorithms for noiseless sparse Bernoulli signals in Fig 9. Obviously, Fig 9 (a) shows that all test algorithms perform poor on single channel Bernoulli signals, since the joint structure prior of signals do not exist here. Surprisingly, the recoverability of ISDJS is dramatically improved as the channel numbers increases in Figs 9 (b), (c), (d) and (e). Similarly, Fig 10 exhibits the relative error of all test algorithms with 0.5% noise in different channel number settings.

All above numerical experiments attest that ISDJS can make significant improvement for multichannel sparse signal recovery even without the fast decaying property, by incorporating joint sparsity property into the imple-



Iteration	Nonzeros				Relative error
	Total true	Detected	Correct	False	
1	30	28	26	2	1.23e-01
2	30	29	29	0	1.42e-03
3	30	30	30	0	9.81e-04
4	30	30	30	0	9.81e-04

Fig 7. Compare the true Bernoulli signals and recovered signals from ISDJS in each iteration with the $L=4$ channels.

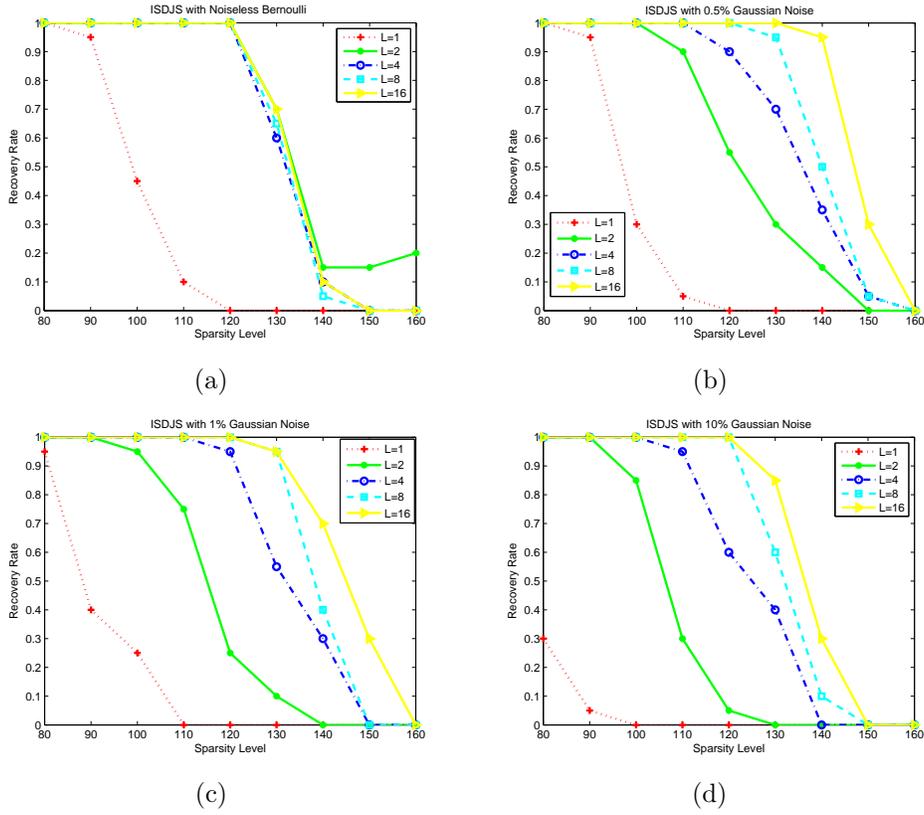


Fig 8. Compare the recovery rate of ISDJS with $L=1, 2, 4, 8, 16$ in different noise levels for Bernoulli signals, (a) noiseless, (b) 0.5% noise, (c) 1% noise, (d) 10% noise.

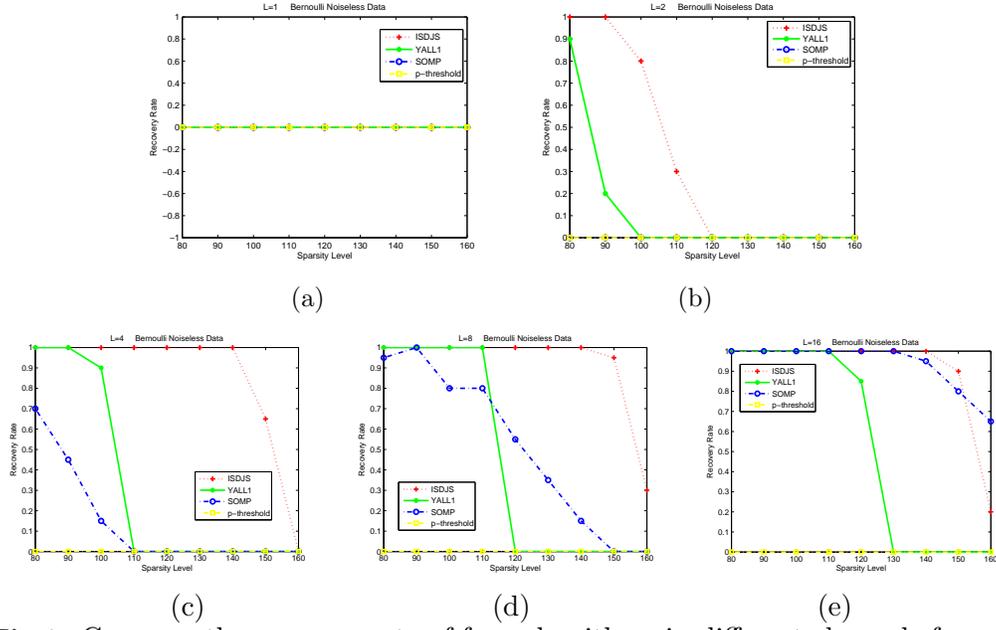


Fig 9. Compare the recovery rate of four algorithms in different channels for noiseless Bernoulli signals, (a)L=1, (b)L=2, (c)L=4, (d)L=8, (e)L=16.

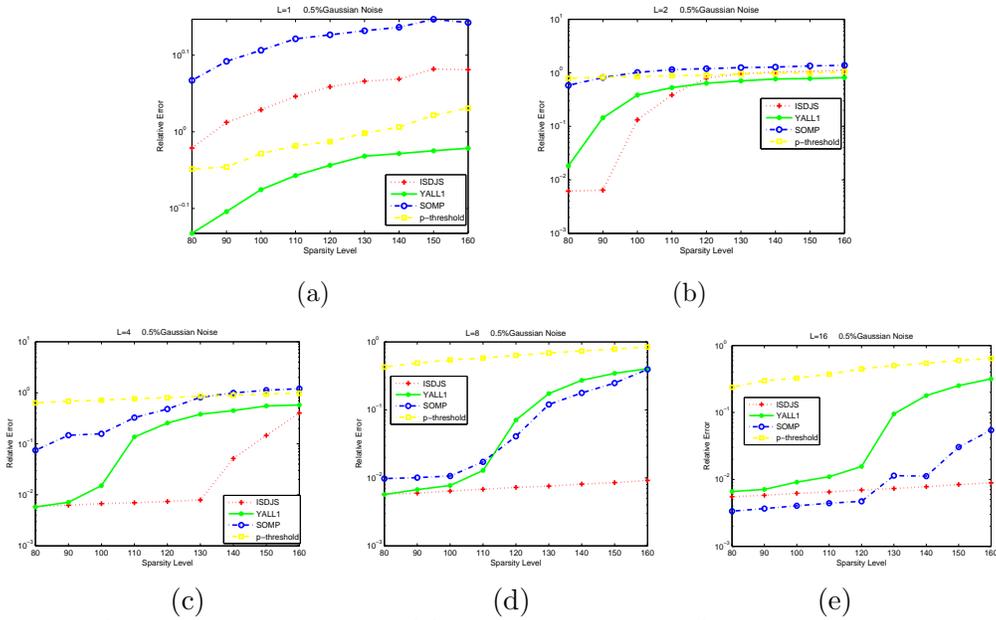


Fig 10. Compare relative error of four algorithms in different channels with 0.5% noise for Bernoulli signals, (a)L=1, (b)L=2, (c)L=4, (d)L=8, (e)L=16.

mentation of threshold based support detection.

4.2. An Example from Collaborative Spectrum Sensing

Now we consider a compressive spectrum sensing scheme for cognitive radio networks [27, 4]. Spectrum sensing aim to detect spectrum holes (i.e., channels not used by any primary users). The cognitive radio (CR) nodes must constantly sense the spectrum in order to detect the presence of the primary radio (PR) nodes and use the spectrum holes without causing harmful interference to the PRs. In practice, improving the ability of detecting complete spectrum usage in collaborative spectrum sensing is an important topic but also a major challenge in CR networks.

We view a l -node cognitive radio network within a 500×500 meter square area centered at the fusion center. The l CR nodes are uniformly randomly located. These cognitive radio nodes collaboratively sense the existence of primary users within a 1000×1000 meter square area on n channels, which are centered also at the fusion center. A channel is either occupied by a PR or unoccupied, corresponding to the states 1 and 0, respectively. Let an $n \times n$ diagonal matrix H represent the states of all the channel sources using 0 and 1 as diagonal entries, indicating the unoccupied or occupied states, respectively. Channel gains are characterized by an $l \times n$ channel gain matrix G . Then, the collaborative spectrum sensing model can be formulated as follows [27]:

$$X_{n \times l} = H_{n \times n} (G_{l \times n})^T. \quad (31)$$

For X , the j -th column of X corresponds to the channel occupancy status received by the j -th CR, and the i -th row of X corresponds to the occupancy status of the i -th channel. A row has a positive value if and only if the i -th channel is used. Since there are only a small number of used channels, X is sparse in terms of the number of nonzero rows. In this example, we set $n = 25$ and $l = 1, 2, 4, 8, 16$. We apply the ISDJS to solve above collaborative spectrum sensing model. Fig 11 presents the results of ISDJS compared with YALL1 group, SOMP and p-threshold algorithms in different settings of l . With the sparsity level (nonzero rows) of X increasing, the advantage of ISDJS is notable.

4.3. An Example for Multi-task Feature Learning

We provide an example to demonstrate the performance of ISDJS for multi-task feature learning. A real-world data set, i.e. Letter [25, 28] is employed in this experiment. The Letter data set was collected by the MIT

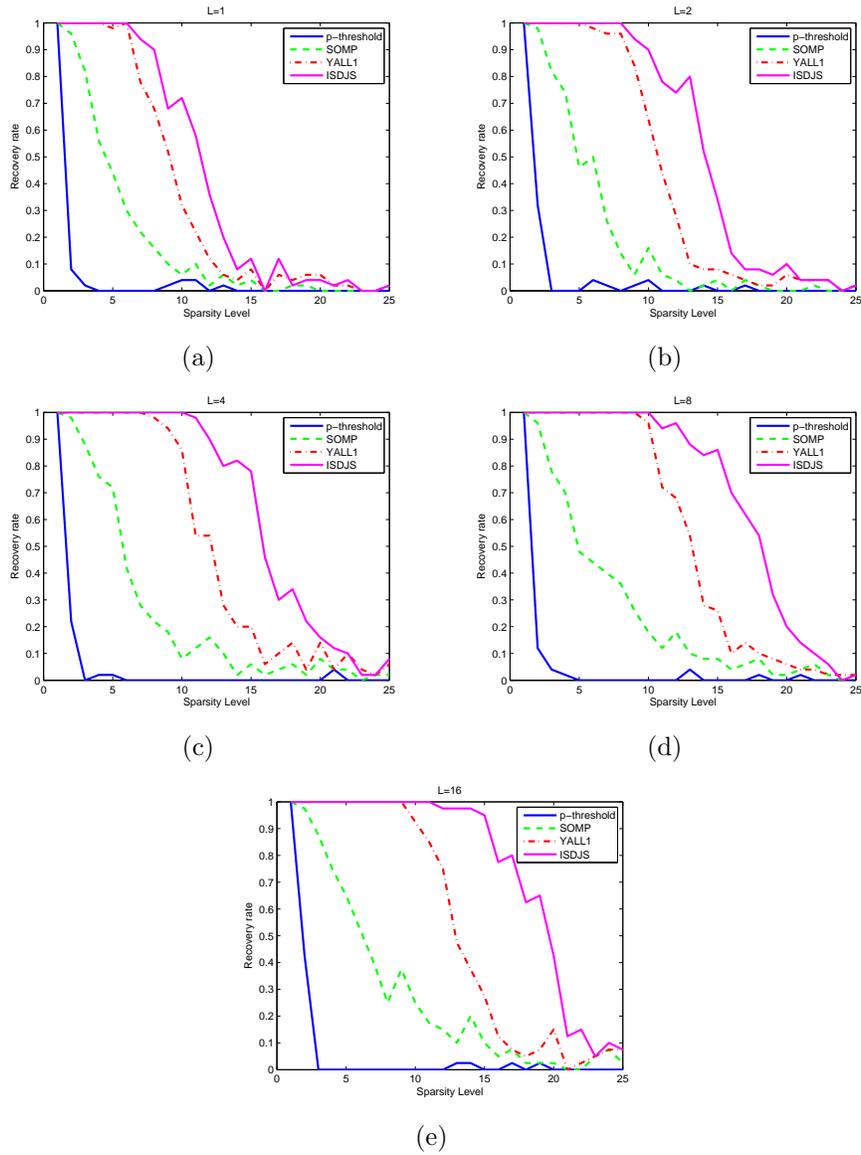


Fig 11. Compare the recovery rate of four algorithms in different channel numbers in spectrum sensing, (a)L=1, (b)L=2, (c)L=4, (d)L=8, (e)L=16.

Spoken Language Systems Group ¹. It contains 8 default tasks for the hand-written letters. The writings are collected from over 180 different writers and there are 45,679 samples, where the letters are represented by 8×16 binary pixel images. This is a typical multi-task feature learning problem.

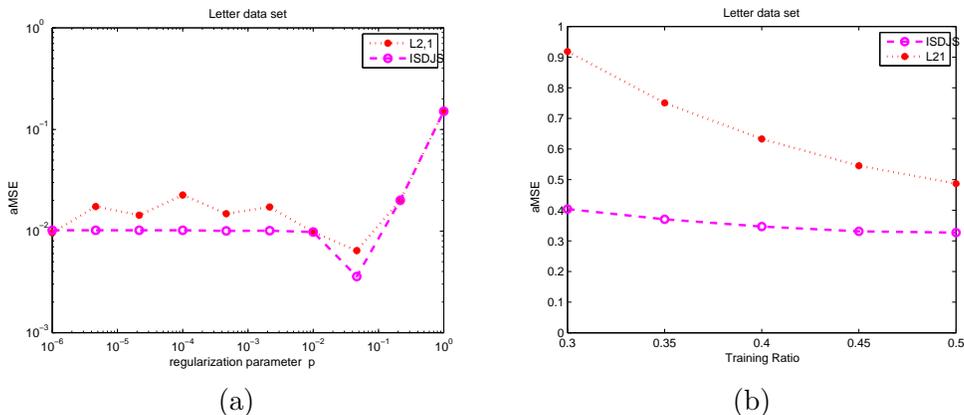


Fig 12. Compare the algorithm in [25] with ISDJS on Letter data set. (a) aMSE vs. regularization parameter ρ , (b) aMSE vs. training ratio.

The baseline algorithm for solving the common $\ell_{2,1}$ regularized model proposed in [25] is used to compare with ISDJS. Here we do not evaluate the performance according to the estimation error of the weight matrix X , whose true values are unknown in practice. Instead, we use the averaged means squared error (aMSE) and normalized mean squared error (nMSE):

$$\text{aMSE} = \frac{\|\hat{b} - \bar{b}\|_F}{\|\bar{b}\|_F},$$

$$\text{nMSE} = N \frac{\|\hat{b} - \bar{b}\|_F^2}{\|\hat{b}\|_1 \cdot \|\bar{b}\|_1},$$

where \hat{b} is the predictive value of the trained model for the test set, \bar{b} is the known reference true value and N is the number of the test sample. Both nMSE and aMSE are commonly used in multi-task learning problems [15].

It is well known that an appropriate regularization parameter is vital for the great performance of the algorithm. As depicted in Fig 12 (a), ISDJS

¹<http://www.seas.upenn.edu/~taskar/ocr/>

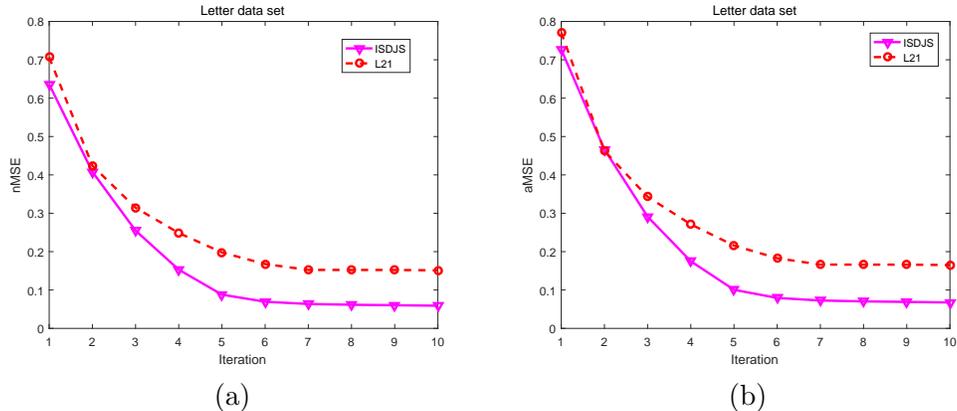


Fig 13. Compare the algorithm in [25] with ISDJS on Letter data set. (a) nMSE vs. iteration, (b) aMSE vs. iteration.

achieves a small error when the regularization parameter ρ is around 4.64×10^{-2} . Then, we randomly extract the training samples from each task with different training ratios (30%, 35%, 40%, 45% and 50%) and exploit the rest of samples to form a test set. In Fig 12 (b), ISDJS performs much better even with a small training ratio. It is easy to observe that ISDJS is convergent after a few iterations in Fig 13, which is consistent with the **Theorem 1** in Section 3. This real-world experimental results for the multi-task feature learning further support the effectiveness of ISDJS.

5. Conclusion

In this paper, we have proposed a truncated joint sparsity model and developed an efficient algorithm named ISDJS to enhance the sparse estimation performance. They are applied in the fields of compressive sensing and feature learning. The proposed method is an extension of self-learning based iterative support detection (ISD) from common sparsity to joint sparsity. The joint sparsity structure is naturally incorporated into the implementation of threshold based support detection and in this way the fast decaying property is no longer required. Then, we have elaborated some preliminary results of the convergence analysis and a sufficient recovery condition for the proposed method. Both synthetic and practical experiments demonstrate the better performance of ISDJS compared with several state-of-the-art alternatives. In the future, we will explore more applications such as image inpainting, image classification and feature selection to employ the ISDJS

method, and design specific implementations of support detection to achieve the outstanding performance for different applications.

Acknowledgment

The authors would like to thank Prof. Wenxing Zhang for valuable discussion about the theoretical analysis of the proposed method, and the editor and referees for their valuable suggestions and comments. This work is supported by the 973 project (No. 2015CB856000), the Natural Science Foundation of China (91330201) and the Fundamental Research Funds for the Central Universities (ZYGX2013Z005).

References

References

- [1] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.
- [2] E.J. Candes, M.B. Wakin, and S.P. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5):877–905, 2008.
- [3] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):679–698, 1986.
- [4] W.-L. Chang, D. Zeng, R.-C. Chen, and S. Guo. An artificial bee colony algorithm for data collection path planning in sparse wireless sensor networks. *International Journal of Machine Learning and Cybernetics*, 6(3):375–383, 2015.
- [5] R. Chartrand. Exact reconstruction of sparse signals via nonconvex minimization. *IEEE Signal Processing Letters*, 14(10):707–710, 2007.
- [6] R. Chartrand and W. Yin. Iteratively reweighted algorithms for compressive sensing. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3869–3872. IEEE, 2008.

- [7] S.F. Cotter, B.D. Rao, K. Engan, and K. Kreutz-Delgado. Sparse solutions to linear inverse problems with multiple measurement vectors. *IEEE Transactions on Signal Processing*, 53(7):2477–2488, 2005.
- [8] W. Deng, W. Yin, and Y. Zhang. Group sparse optimization by alternating direction method. In *SPIE Optical Engineering + Applications*, pages 88580R–88580R. International Society for Optics and Photonics, 2013.
- [9] M.F. Duarte, S. Sarvothamand, M.B. Wakin, D. Baron, and R.G. Baraniuk. Joint sparsity models for distributed compressed sensing. In *Proceedings of the Workshop on Signal Processing with Adaptative Sparse Structured Representations*. IEEE, 2005.
- [10] Y.C. Eldar and H. Rauhut. Average case analysis of multichannel sparse recovery using convex relaxation. *IEEE Transactions on Information Theory*, 56(1):505–519, 2010.
- [11] Y.-R. Fan, T.-Z. Huang, J. Liu, and X.-L. Zhao. Compressive sensing via nonlocal smoothed rank function. *PloS One*, 11(9):e0162041, 2016.
- [12] Y.-R. Fan, T.-Z. Huang, T.-H. Ma, and X.-L. Zhao. Cartoon–texture image decomposition via non-convex low-rank texture regularization. *Journal of the Franklin Institute*, 354(7):3170–3187, 2017.
- [13] Y. Fang and T. Gui-fa. Visual music score detection with unsupervised feature learning method based on K-means. *International Journal of Machine Learning and Cybernetics*, 6(2):277–287, 2015.
- [14] A. Gonçalves, P. Das, S. Chatterjee, V. Sivakumar, F.J. Von Zuben, and A. Banerjee. Multi-task sparse structure learning. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 451–460. ACM, 2014.
- [15] P. Gong, J. Ye, and C. Zhang. Multi-stage multi-task feature learning. In *Advances in Neural Information Processing Systems*, pages 1988–1996, 2012.
- [16] R. Gribonval, B. Mailhe, H. Rauhut, K. Schnass, and P. Vandergheynst. Average case analysis of multichannel thresholding. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages II–853. IEEE, 2007.

- [17] R. Heckel and H. Bolcskei. Joint sparsity with different measurement matrices. In *50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 698–702. IEEE, 2012.
- [18] Y. Hu, J. Liu, C. Leng, Y. An, S. Zhang, and K. Wang. L_p regularization for bioluminescence tomography based on the split bregman method. *Molecular Imaging and Biology*, 18(6):830–837, 2016.
- [19] M.M. Hyder and K. Mahata. A robust algorithm for joint-sparse recovery. *IEEE Signal Processing Letters*, 16(12):1091–1094, 2009.
- [20] T. Ince, A. Nacaroglu, and N. Watsuji. Nonconvex compressed sensing with partially known signal support. *Signal Processing*, 93(1):338–344, 2013.
- [21] B. Jiang, Y.-F. Liu, and Z. Wen. L_p -norm regularization algorithms for optimization over permutation matrices. *SIAM Journal on Optimization*, 26(4):2284–2313, 2016.
- [22] J.M. Kim, O.K. Lee, and J.C. Ye. Improving noise robustness in subspace-based joint sparse recovery. *IEEE Transactions on Signal Processing*, 60(11):5799–5809, 2012.
- [23] M.-J. Lai and Y. Liu. The null space property for sparse recovery from multiple measurement vectors. *Applied and Computational Harmonic Analysis*, 30(3):402–406, 2011.
- [24] K. Lee, Y. Bresler, and M. Junge. Subspace methods for joint sparse recovery. *IEEE Transactions on Information Theory*, 58(6):3613–3641, 2012.
- [25] J. Liu, S. Ji, and J. Ye. Multi-task feature learning via efficient $\ell_{2,1}$ -norm minimization. In *Proceedings of the Twenty-fifth Conference on Uncertainty in Artificial Intelligence*, pages 339–348. AUAI Press, 2009.
- [26] H. Lu, X. Long, and J. Lv. A fast algorithm for recovery of jointly sparse vectors based on the alternating direction methods. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 461–469, 2011.

- [27] J.J. Meng, W. Yin, H. Li, E. Hossain, and Z. Han. Collaborative spectrum sensing from sparse observations in cognitive radio networks. *IEEE Journal on Selected Areas in Communications*, 29(2):327–337, 2011.
- [28] G. Obozinski, B. Taskar, and M. Jordan. Joint covariate selection for grouped classification. Technical report, Technical report, Statistics Department, UC Berkeley, 2007.
- [29] Z. Qin, K. Scheinberg, and D. Goldfarb. Efficient block-coordinate descent algorithms for the group lasso. *Mathematical Programming Computation*, 5(2):143–169, 2013.
- [30] A.A. Saleh, F. Alajaji, and W.-Y. Chan. Compressed sensing with non-Gaussian noise and partial support information. *IEEE Signal Processing Letters*, 22(10):1703–1707, 2015.
- [31] T. Sanders, A. Gelb, and R.B. Platte. Composite SAR imaging using sequential joint sparsity. *Journal of Computational Physics*, 338:357–370, 2017.
- [32] A. Singh and S. Dandapat. Block sparsity-based joint compressed sensing recovery of multi-channel ecg signals. *Healthcare Technology Letters*, 4(2):50–56, 2017.
- [33] J.A. Tropp, A.C. Gilbert, and M.J. Strauss. Algorithms for simultaneous sparse approximation. Part I: Greedy pursuit. *Signal Processing*, 86(3):572–588, 2006.
- [34] Y. Wang and W. Yin. Sparse signal reconstruction via iterative support detection. *SIAM Journal on Imaging Sciences*, 3(3):462–491, 2010.
- [35] Z. Wen, B. Hou, and L. Jiao. Joint sparse recovery with semisupervised MUSIC. *IEEE Signal Processing Letters*, 24(5):629–633, 2017.
- [36] T. Wimalajeewa and P.K. Varshney. OMP based joint sparsity pattern recovery under communication constraints. *IEEE Transactions on Signal Processing*, 62(19):5059–5072, 2014.
- [37] J. Yang and Y. Zhang. Alternating direction algorithms for ℓ_1 -problems in compressive sensing. *SIAM journal on scientific computing*, 33(1):250–278, 2011.

- [38] X. Yang, S. Kim, and E.P. Xing. Heterogeneous multitask learning with joint sparsity constraints. In *Advances in Neural Information Processing Systems*, pages 2151–2159, 2009.
- [39] Y. Yang, C. Deng, S. Gao, W. Liu, D. Tao, and X. Gao. Discriminative multi-instance multitask learning for 3D action recognition. *IEEE Transactions on Multimedia*, 19(3):519–529, 2017.
- [40] Z. Yang and L. Xie. Exact joint sparse frequency recovery via optimization methods. *IEEE Transactions on Signal Processing*, 64(19):5145–5157, 2016.
- [41] X. Zhang and C. Liu. A one-dimensional slope detection approach. *SpringerPlus*, 2(1):474, 2013.
- [42] J. Zhou, J. Chen, and J. Ye. Malsar: Multi-task learning via structural regularization. *Arizona State University*, 21, 2011.
- [43] X.X. Zhu, C. Grohnfeldt, and R. Bamler. Exploiting joint sparsity for pansharpening: The J-sparseFI algorithm. *IEEE Transactions on Geoscience and Remote Sensing*, 54(5):2664–2681, 2016.

