MS/SIM Jae-Gun Choi (최재건). Compressive Sensing and Its Application in Wireless
20111092 Sensor Network & Correlated Signal Recovery Method
School of Information and Mechatronics. 2013. Prof. Heung-No Lee

Abstract

In this paper, we consider the application of a new compression technique called compressive sensing (CS) in wireless sensor networks (WSNs). CS is a signal acquisition and compression framework recently developed in the field of signal processing and information theory. We applied this CS technique to WSN which consists of a large number of wireless sensor nodes and a central fusion center (FC). This CS based signal acquisition and compression is done by a simple linear projection at each sensor node. Then, each sensor transmits the compressed samples to the FC. The FC which collects the compressed signals from the sensors jointly reconstructs the signals in polynomial time using a signal recovery algorithm.

The distributed sensors observe similar event in designated region. Therefore, the observed signals have considerable correlation each other. We make some effort in modeling correlation between the signals acquired from the sensors and analyze the component in observed signals. After modeling the correlated signals, we propose POMP (Phased-OMP) which can recover any type of correlated signals stably and effectively. We introduce the idea of our proposed algorithm in detail and then compare the reconstruction performance of POMP with previous algorithms ReMBo, MEM, SOMP, etc.

Contents

1 Introduction

2 Wireless sensor network

2.1 Network structure

2.2 Resource limitations in WSNs

3 Compressive sensing (Literature survey)

3.1 Theoretical background

3.2 System equations

3.3 Unique solution condition of SMV and MMV

4 Compressive sensing and its application in WSN

4.1 The usefulness of CS in WSNs

4.2 Distributed compressive sensing

4.3 Correlated signal models

5 The recovery ideas for correlated signals

5.1 Joint decoding method for MMV (Literature survey)

5.2 Joint vs. separate recovery performance for JSM-1 and JSM-2

5.3 Phased-orthogonal matching pursuit (POMP)

5.4 The properties of POMP algorithm

6 Performance evaluation

7 Conclusion

8 Appendix

8.1 Primal-dual interior point method (PDIP)

8.2 Orthogonal matching pursuit (OMP)

8.3 Simultaneous orthogonal matching pursuit (SOMP)

8.4 Reduce and boost (ReMBo)

9 Reference

List of Figures

Figure 1. Wireless Sensor Network (WSN). 10

Figure 2. Conventional compression and compressive sensing. 13

Figure 3. The summary of compressive sensing. 16

Figure 4. The relationship among SMV, MMV, and IMV 22

Figure 5. Conventional sensor network schme. 28

Figure 6. Conventional sensor network structure 28

Figure 7. CS sensor network scheme 30

Figure 8. CS sensor network structure 30

Figure 9. Intra-sensor correlation scheme 32

Figure 10. Intra/Inter-sensor correlation scheme. 32

Figure 11. The examples of correlated signals 33

Figure 12. The components of correlated signals 34

Figure 13. Joint signal models, JSM-1, JSM-2. 36

Figure 14. Concatenating JSM-1 to a column signal. 38

Figure 15. Graph1. 41

Figure 16. Reduce MMV to SMV in ReMBo 42

Figure 17. The limitation of ReMbo algorithm. 43

Figure 18. Graph2 44

Figure 19. Graph3 45

Figure 20. Total sparsity of MMV equation 47

Figure 21. The movement of POMP algorithm 51

Figure 22. Graph 4 53

Figure 23. Graph 5 54

Figure 24. Correlated signal model (CSM) 58

Figure 25. Graph 6 60

Figure 26. Graph 7 61

Figure 27. Graph 8 62

Figure 28. Graph 9 63

Figure 29. Graph 10 64

Figure 30. Summary of CS application in WSN.. 67

List of Tables

Table 1. L_0 / L_1 Equivalence condition in SMV....................25

Table 2. L_0 / L_1 Equivalence condition in MMV....................27

Table 3. The one-step greedy algorithm (OSGA). 41

Table 4. Inputs and outputs of POMP algorithm. 49

Table 5. POMP algorithm....................50

Table 6. Moore-Penrose pseudo inverse....................52

Table 7. Pseudo-inverse in MMV 55

Table 8. The complexity of POMP algorithm....................57

Table 9. The performance of various algorithms with related to correlated signals. 59

Table 10. Primal-dual interior point method algorithm....................70

Table 11. Inputs and outputs of OMP algorithm....................71

Table 12. OMP algorithm....................71

Table 13. Inputs and outputs of SOMP algorithm....................73

Table 14. SOMP algorithm....................73

Table 15. Inputs and outputs of ReMBo algorithm....................74

Table 16. ReMBo algorithm 74

Acknowledgement

This work has been fully supported by Prof. Heung-No Lee, under his National Research Foundation funded project (include the logistics of Do-Yak project). The work in this project has started in spring 2010 by joining the Compressive Sensing research team in the INFONET laboratory which is run by Prof. Lee. I've also taken Professor Lee's course, Introduction to Compressed Sensing, in spring 2011 [1]. This course not only taught us the basic concepts of Compressive Sensing but also gave an introduction to the basic mathematics of CS theory. While taking the course, I could read many papers about compressive sensing and wireless sensor network. After taking the class successfully, I got a chance to write book chapter with Prof. Heung-No Lee. The name of book chapter is "Chapter 15: Compressive Sensing and Its Application in Wireless Sensor Networks" [2]. It will be published in 2012. 12. 13 with the title of book, "Intelligent Sensor Networks: Across sensing, signal processing, and machine learning". I have learned various knowledge and theory about CS while writing book chapter. The experience for writing book chapter helps to finish my master thesis.

무엇보다도 2년이라는 석사 기간 동안, 많은 관심으로 지도해 주신 이흥노 교수님께 감사 드립니다. 처음 INFONET 연구실에 소속되어서부터 졸업을 앞두고 있는 지금까지, 함께 생활하면서 즐거운 추억을 만들었던 연구실 가족들, 박순철 연구교수님, 변상선 연구교수님, 수정씨, Oliver, Pavel, Evgeni, Nitin, Asif, Zafar, 진택이형, 환철이형, 웅비형, 영학이형, 승찬이형, 상준 선배, 동기 정민이, 그리고 후배 수길이와 형호한테 감사 드리며, 졸업을 한 이후로도 연락을 주고 받으며 친분을 지속해 나갔으면 하는 바람이 있습니다. 연구실의 구성원들로부터 받았던 많은 도움들과 조언에 대해 잊지 못할 것입니다. 마지막으로 대학원 과정을 마치기까지 저를 믿어주시고 지켜봐 주신 가족들께 감사 드리며, 이 기쁨을 함께 나누고 싶습니다.

1. Introduction

In this paper, we discuss the application of a new compression technique called compressive sensing (CS) in wireless sensor networks (WSNs). The objective of a WSN which we assume in this paper is to collect information about events occurring in a region of interest. This WSN consists of a large number of wireless sensor nodes and a central fusion center (FC). The sensor nodes are spatially distributed over the said region to acquire physical signals such as sound, temperature, wind speed, pressure, and seismic vibrations. After sensing, they transmit the measured signals to the FC. In this paper, we focus on the role of the FC which is to recover the transmitted signals in their original waveforms for further processing. By doing so, the FC can produce a global picture that illustrates the event occurring in the sensed region. Each sensor uses its onboard battery for sensing activities and makes reports to FC via wireless transmissions. Thus, limited power at the sensor nodes is the key problem to be resolved in the said WSN.

CS is a signal acquisition and compression framework recently developed in the field of signal processing and information theory [3],[4]. Donoho [3] says that "The Shannon–Nyquist sampling rate may lead to too many samples; probably not all of them are necessary to reconstruct the given signal. Therefore, compression may become necessary prior to storage or transmission." According to Baraniuk [5], CS provides a new method of acquiring compressible signals at a rate significantly below the Nyquist rate. This method employs non-adaptive linear projections that preserve the signal's structure; the compressed signal is then reconstructed from these projections using an optimization process.

We applied this CS technique to WSN. One of our aims in this paper is to determine whether the CS can be used as a useful framework for the aforementioned WSN to compress and acquire signals and save transmittal and computational power at the sensor node. This CS based signal acquisition and compression is done by a simple linear projection at each sensor node. Then, each sensor transmits the compressed samples to the FC; the FC which collects the compressed signals from the sensors jointly reconstructs the signal in polynomial time using a signal recovery algorithm. Illustrating this process in detail throughout this chapter, we check to see if CS can become an effective, efficient strategy to be employed in WSNs, especially for those with low-quality, inexpensive sensors.

The distributed sensors observe similar event in designated region. Therefore, the observed signals have considerable correlation each other. In this paper, as we assume a scenario in which a WSN is used for signal acquisition, we intend to pay some effort in modeling correlation between the signals acquired from the sensors. Then, we divide the correlated signals to three parts for example, common sparsity, innovation sparsity, and total sparsity. Those terminologies give more easy understanding to solve multiple measurement vector (MMV) modeled from WSN structure.

If we will use the correlated information to recover signals transmitted from each sensor, its reconstruction performance will increase over that not using correlated information. We demonstrated this assumption by showing a simulation result. Finally, we proposed advanced algorithm to recover the correlated signals effectively. The proposed algorithm is named as phased advanced orthogonal matching pursuit (POMP). POMP has better performance about reconstruction probability than previous algorithms, for examples, MEM, SOMP, ReMBo in [6],[7],[8],[9] etc. We will introduce the idea of our proposed algorithm in detail and then compare the reconstruction performance of our algorithm with those of the previous algorithms

2. Wireless sensor network

2.1. Network structure

We consider a WSN consisting of a large number of wireless sensor nodes and one FC (**Figure 1**). The wireless sensor nodes are spatially distributed over a region of interest and observe physical changes such as those in sound, temperature, pressure, or seismic vibrations. If a specific event occurs in a region of distributed sensors, each sensor makes local observations of the physical phenomenon as the result of this event taking place. An example of sensor network applications is area monitoring to detect forest fires. A network of sensor nodes can be installed in a forest to detect when a fire breaks out. The nodes can be equipped with sensors to measure temperature, humidity, and the gases produced by fires in trees or vegetation [10]. Other examples include military and security applications. Military applications vary from monitoring soldiers in the field, to tracking vehicles or enemy movement. Sensors attached to soldiers, vehicles and equipment can gather information about their condition and location to help planning activities on the battlefield. Seismic, acoustic and video sensors can be deployed to monitor critical terrain and approach routes; reconnaissance of enemy terrain and forces can be carried out [11].

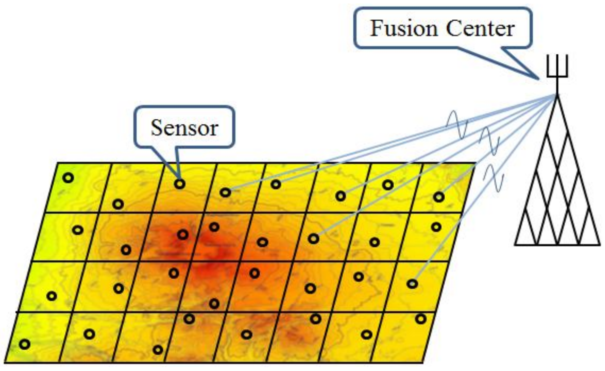


Figure 1. Wireless Sensor Network (WSN)

After sensors observe an event taking place in a distributed region, they convert the sensed information into a digital signal and transmit the digitized signal to the FC. Finally, the FC assembles the data transmitted by all the sensors and decodes the original information. The decoded information at the FC provides a global picture of events occurring in the region of interest. Therefore, we assume that the objective of the sensor network is to determine accurately and rapidly reconstruct transmitted information and reconstruct the original signal.

We discuss the resource limitations of WSNs in the next section.

2.2. Resource limitations in WSNs

In this section, we describe the assumptions made in the sensor network we are interested in. We assume that the sensors are distributed and supposed to communicate with the FC through a wireless channel. Because each sensor is important components of WSN which observes event, they should typically be deployed in a large volume over the region of interest. Therefore, they are usually designed to be inexpensive and small. For that reason, each sensor operates on an onboard battery which is not rechargeable at all; thus, for simplicity, the hardware implementation of sensor nodes can provide only limited computational performance, bandwidth, and transmission power. As a result of limitations on the hardware implementation in sensor nodes, the FC has powerful computation performance and plentiful energy which naturally performs most of the complex computations.

Under the limited conditions stated above for a WSN, CS can substantially reduce the data volume to be transmitted at each sensor node. With the new method, it is possible to compress the original signal using only $O\left(k\log\left(n/k\right)\right)$ samples without going through many complex signal processing steps. These signals can be recovered successfully at the FC. All these are done under the CS framework. As the result, the consumption of power for transmission of signal contents at each sensor can be significantly reduced thanks to decreased data volume. Moreover, this data reduction comes without utilizing complex signal processing. Namely, the sensor nodes can compress the signal while not spending any power for running complex compression algorithms onboard.

We discuss the new technique CS in the next section and check how CS can get the advantages like data reduction and simple data compression.

3. Compressive sensing (Literature survey)

In a conventional communication system, an analog-to-digital converter based on the Shannon–Nyquist sampling theorem is used to convert analog signals to digital signals. The theorem says that if a signal is sampled at a rate twice, or higher, the maximum frequency of the signal, the original signal can be exactly recovered from the samples. Once the sampled signals are obtained over a fixed duration of time, a conventional compression scheme can be used to compress them. Because the sampled signals often have substantial redundancy, compression is possible. Several compression schemes follow this approach, e.g., the MP3 and JPEG formats for audio or image data. However, conventional compression in a digital system is sometimes inefficient because it requires unnecessary signal processing stages, for example, retaining all of the sampled signals in one location before data compression. According to Donoho [3], the CS framework, as shown in **Figure 2**, can bypass these intermediate steps, and thus provides a light weight signal acquisition apparatus which is suitable for those sensor nodes in our WSN.

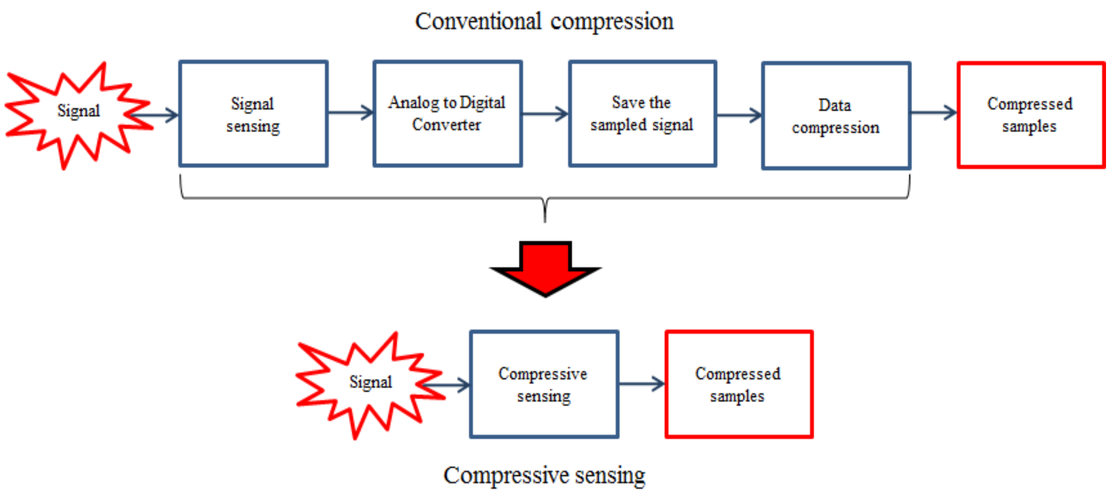


Figure 2. Conventional compression and compressive sensing

The CS provides a direct method which acquires compressed samples without going through the intermediate stages of conventional compression. Thus, CS provides a much simpler signal acquisition solution. In addition, the CS provides several recovery routines which the original signal can be regenerated perfectly from the compressed samples.

3.1 Theoretical background

Let a real-valued column vector $\mathbf{s}$ be a signal to be acquired. Let it be represented by

$$\mathbf{s} = \Psi\mathbf{x} \tag{1}$$

,where $\mathbf{x}$ and $\mathbf{s} \in \mathbf{R}^n$, and $\mathbf{x}$ is also a real-valued column vector. The matrix $\Psi \in \mathbf{R}^{n\times n}$ is an orthonormal basis, i.e., $\Psi^T\Psi = \Psi\Psi^T = I_n$, the identity matrix of size $\mathbf{R}^{n\times n}$. The signal $\mathbf{s}$ is called k-sparse if it can be represented as a linear combination of only k columns of Ψ, i.e., only the k components of the vector $\mathbf{x}$ are nonzero as represented Eq.(2) .

.

$$\mathbf{s} = \sum_{i=1}^{n} x_i \psi_i, \text{ where } \psi_i \text{ is a column vector of } \Psi. \tag{2}$$

A signal is called compressible if it has only a few significant (large in magnitude) components and a greater number of insignificant (close to zero) components. The compressive measurements $\mathbf{y}$ (compressed samples) are obtained via linear projections as follows (**Figure 3**):

$$\mathbf{y} = \Phi\mathbf{s} = \Phi\Psi\mathbf{x} = \mathbf{A}\mathbf{x} \tag{3}$$

where the measurement vector is $\mathbf{y} \in \mathbf{R}^m$, with $m < n$, and the measurement matrix $\mathbf{A} \in \mathbf{R}^{m\times n}$. Our goal is to recover $\mathbf{x}$ from the measurement vector $\mathbf{y}$. We note that Eq. (3) is an underdetermined system because it has fewer equations than unknowns; thus, it does not have a unique solution in general. However, the theory of CS asserts that, if the vector $\mathbf{x}$ is sufficiently sparse, an underdetermined system is guaranteed with high probability to have a unique solution.

In this section, we discuss the basics of CS in more detail.

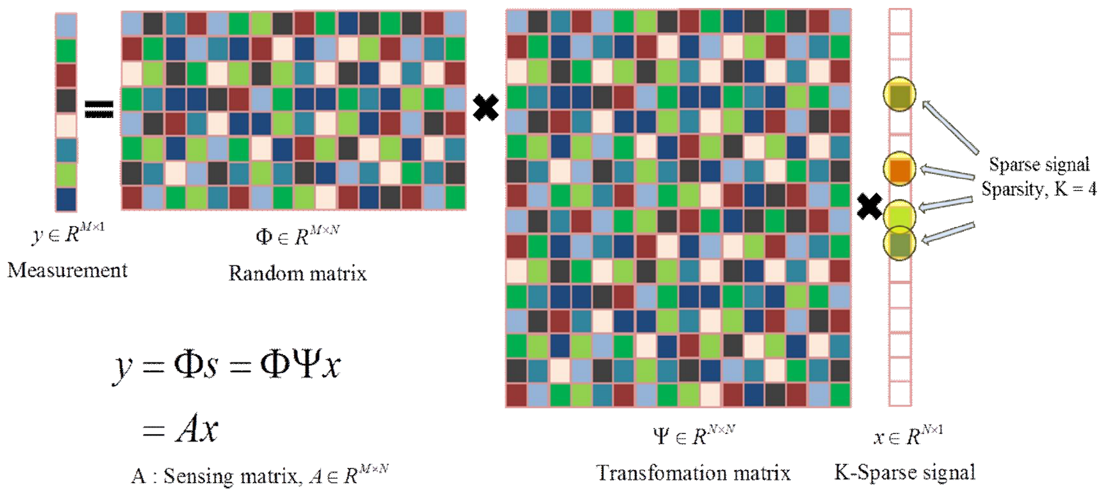


Figure 3. The summary of compressive sensing

i) k-sparse signal $\mathbf{x}$ in orthonormal basis

The k-sparse signal, $\mathbf{s}$ in Eq. (1), has k nonzero components in $\mathbf{x}$. The matrix Ψ is, again, an orthonormal basis, i.e., $\Psi^T\Psi = \Psi\Psi^T = I_n$, the identity matrix of size $\mathbf{R}^{n\times n}$.

ii) Measurement vector $\mathbf{y}$ and underdetermined system

The sensing matrices are Φ and $\mathbf{A}$ in Eq. (3), where its dimension $\mathbf{R}^{m\times n}$, $m < n$. When m is closer to k than n is, sufficient conditions for good signal recovery are satisfied. Then a compression effect exists. Note that Eq. (3) appears to be an ill-conditioned equation. That is, the number of unknowns n is larger than m the number of equations, $m < n$. However, if $\mathbf{x}$ is k-sparse and the locations of the k nonzero elements are known, the problem can be solved provided $m \geq k$. We can form a simplified equation by deleting all those columns and elements corresponding to the zero-elements, as follows:

$$\mathbf{y} = \mathbf{A}_\kappa \mathbf{x}_\kappa \tag{4}$$

where $\kappa \in \{1, 2, \cdots, n\}$ is the support set, which is the collection of indices corresponding to the nonzero elements of x. Note that the support set κ can be any size-k subset of the full index set, $\{1, 2, 3, ..., n\}$. Eq. (4) has the unique solution $\mathbf{x}_\kappa$ if the columns of $\mathbf{A}_\kappa$ are linearly independent. The solution can be found using pseudo inverse easily as Eq. (5)

$$\mathbf{x}_\kappa = \left(\mathbf{A}_\kappa^T \mathbf{A}_\kappa\right)^{-1} \mathbf{A}_\kappa^T \mathbf{y} \tag{5}$$

Thus, if the support set κ can be found, the problem is easy to solve provided the columns are linearly independent.

iii) Incoherence condition

The incoherence condition is that the rows of Φ should be incoherent to the columns of Ψ. If the rows of Φ are coherent to the columns of Ψ, the matrix $\mathbf{A}$ cannot be a good sensing matrix. In the extreme case, we can show a matrix $\mathbf{A}$ having m rows of Φ that are the first m columns of Ψ.

$$\mathbf{A} = \Phi\Psi = \Psi_{(1:m,:)}^T \Psi = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & 0 & \cdots & 0 \end{bmatrix} \tag{6}$$

If $\mathbf{A}$ of Eq. (6) is used as sensing matrix, the compressed measurement vector $\mathbf{y}$ captures only the first m elements of the vector $\mathbf{x}$, and the rest of the information contained in $\mathbf{x}$ is completely lost.

iv) Designing a sensing matrix Φ

One choice for designing a sensing matrix Φ is Gaussian. Under this choice, the sensing matrix Φ is designed as a Gaussian, i.e., matrix elements are independent and identically distributed Gaussian samples. This choice is deemed good since a Gaussian sensing matrix satisfies the incoherence condition with high probability for any choice of orthonormal basis Ψ. This randomly generated matrix acts as a random projection operator on the signal vector $\mathbf{x}$. Such a random projection matrix needs not depend on specific knowledge about the source signals. Moreover, random projections have the following advantages in the application to sensor networks [7].

1) Universal incoherence: Random matrices Φ can be combined with all conventional sparsity basis Ψ, and with high probability sparse signals can be recovered by an L_1 minimum algorithms from the measurements $\mathbf{y}$.

2) Data independence: The construction of a random matrix does not depend on any prior knowledge of the data. Therefore, given an explicit random number generator, only the sensors and the fusion center are required to agree on a single random seed for generating the same random matrices of any dimension.

3) Robustness: Transmission of randomly projected coefficients is robust to packet loss in the network. Even if part of the elements in measurement $\mathbf{y}$ is lost, the receiver can still recover the sparse signal, at the cost of lower accuracy.

3.2 System equations

We knew the method how to find a unique solution of CS problem in previous section. In this section, we discuss various equations which are handled in CS theory as single measurement vector (SMV) and multiple measurement vector (MMV). The SMV is a basic equation for CS. It is expressed as Eq. (7). The research problems of many CS paper are based on this SMV equation.

$$\mathbf{y} = \mathbf{A}\mathbf{x}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ \\ y_{n-1} \\ y_n \end{bmatrix} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & & a_{2,n} \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ a_{m,1} & a_{m,2} & \cdots & \cdots & a_{m,n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ \\ \vdots \\ x_n \end{bmatrix} \tag{7}$$

Otherwise, the MMV has multiple measurement vectors and sparse matrix as Eq. (8). The sparse vector in each SMV results in MMV. It has much unknowns compared with SMV. The many number of unknowns may make the MMV to be solved hard. To solve this equation effectively, some algorithms are proposed as SOMP, ReMBo, M-FOCUSS. If each column of sparse matrix **X** has similar support set, the priori information about support location can be used to get exact solution easily.

$$\mathbf{Y} = \mathbf{A}\mathbf{X}$$

$$\begin{bmatrix} y_{1,1} & y_{1,2} & \cdots & y_{1,J} \\ y_{2,1} & y_{2,2} & \cdots & y_{2,J} \\ \\ y_{m,1} & y_{m,2} & \cdots & y_{m,J} \end{bmatrix} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & & a_{2,n} \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ a_{m,1} & a_{m,2} & \cdots & \cdots & a_{m,n} \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,J} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,J} \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,J} \end{bmatrix} \tag{8}$$

The MMV equation can have the more number of equations by transforming Eq. (8). It means that the MMV equation has more information to solve underdetermined equation. The modified MMV equation is expressed as below Eq. (9). Furthermore, infinite measurement vector (IMV) consists of an infinite set of jointly sparse vectors.

$$\mathbf{y}_j = \mathbf{A}_j \mathbf{x}_j, \text{ where } j \in \{1, 2, ..., J\}$$

$$\begin{bmatrix} y_{1,1} \\ y_{2,1} \\ \vdots \\ \\ y_{m-1,1} \\ y_{m,1} \end{bmatrix}_1 = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & & a_{2,n} \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ a_{m,1} & a_{m,2} & \cdots & \cdots & a_{m,n} \end{bmatrix}_1 \begin{bmatrix} x_{1,1} \\ x_{2,1} \\ \vdots \\ \vdots \\ \\ \vdots \\ x_{n,1} \end{bmatrix}_1 ,\cdots, \begin{bmatrix} y_{1,J} \\ y_{2,J} \\ \vdots \\ \\ y_{m-1,J} \\ y_{m,J} \end{bmatrix}_J = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & & a_{2,n} \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ a_{m,1} & a_{m,2} & \cdots & \cdots & a_{m,n} \end{bmatrix}_J \begin{bmatrix} x_{1,J} \\ x_{2,J} \\ \vdots \\ \vdots \\ \\ \vdots \\ x_{n,J} \end{bmatrix}_J$$

$$\mathbf{Y} = \mathbf{A}_j \mathbf{X}, \text{ where } j \in \{1, 2, ..., J\}$$

$$\Rightarrow \quad \begin{bmatrix} y_{1,1} & y_{1,2} & \cdots & y_{1,J} \\ y_{2,1} & y_{2,2} & \cdots & y_{2,J} \\ \\ \\ y_{m,1} & y_{m,2} & \cdots & y_{m,J} \end{bmatrix} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & & a_{2,n} \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ a_{m,1} & a_{m,2} & \cdots & \cdots & a_{m,n} \end{bmatrix}_j \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,J} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,J} \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ \\ \vdots & \vdots & & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,J} \end{bmatrix} \tag{9}$$

Therefore, we can draw the relationship among the SMV, MMV and IMV as **Figure 4.** As the **Figure 4** shows, the MMV includes all of the SMV. It means that the MMV has all the information of the SMV. Therefore, if we solve the MMV equation exactly, it results the solution of each SMV also.

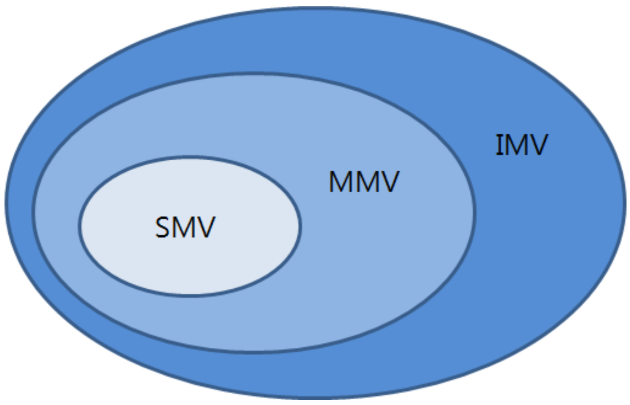


Figure 4. The relationship among SMV, MMV and IMV

3.3. Unique solution condition of SMV and MMV

In CS, a core problem is to find a unique solution for an underdetermined equation. This problem is related to the signal reconstruction algorithm, which takes the measurement vector $\mathbf{y}$ as an input and the k-sparse vector $\mathbf{x}$ as an output. To solve an underdetermined problem, we consider minimization criteria using different norms such as the L_2, L_1, and L_0 norms. The L_p norm of a vector $\mathbf{x}$ of length n is defined as

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}}, \ p > 0 \tag{10}$$

.

Although we can define the L_2 and L_1 norms as $\|\mathbf{x}\|_2 = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{\frac{1}{2}}$ and $\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|$, respectively, using the definition of L_p norm, L_0 norm cannot be defined this way. The L_0 norm is a pseudo-norm that counts the number of nonzero components in a vector as defined by Donoho and Elad [3],[12]. Using this definition of norms, we will discuss the minimization problem to get solution $\mathbf{x}$.

i) The minimization problem in SMV

1) L_2 norm minimization in SMV

$$\begin{aligned}(L_2)\ \hat{\mathbf{x}} &= \arg\min \|\mathbf{x}\|_2 \ \text{ subject to } \mathbf{y} = \mathbf{A}\mathbf{x}, \text{ where } \mathbf{A} \in \mathrm{R}^{m\times n},\ rank(\mathbf{A}) = m \\ &= \mathbf{A}^T \left(\mathbf{A}\mathbf{A}^T \right)^{-1} \mathbf{y}\end{aligned} \tag{11}$$

However, this conventional solution yields a non-sparse solution, so it is not appropriate as a solution to the CS problem. Thus, we do not consider this method for finding solution.

2) L_0 norm minimization in SMV

$$(L_0)\ \text{Minimize } \|\mathbf{x}\|_0 \ \text{ subject to } \mathbf{y}=\mathbf{A}\mathbf{x}, \text{ where } \mathbf{A}\in \mathrm{R}^{m\times n},\ rank(\mathbf{A})=m \tag{12}$$

The L_0 norm of a vector is, by definition, the number of nonzero elements in the vector. In the CS literature, it is known that the L_0 norm problem can be solved by examining all the possible cases. Since this process involves a combinatorial search for all possible $\binom{n}{k}$ support sets, it is an NP-complete problem. Thus, we cannot solve it within polynomial time. Therefore, we consider L_1 norm minimization as an alternative. In literature, the unique solution of the L_0 minimization is known as following,

$$k<\frac{\operatorname{spark}(\mathbf{A})}{2} \tag{13}$$

The $\operatorname{spark}(\mathbf{A})$ is the smallest number n such that there exists a set of n columns in $\mathbf{A}$ which are linearly dependent [12]. In summary, if the above equation is satisfied, then the unique solution of the Eq. (12) is guaranteed.

3) L_1 norm minimization in SMV

$$(L_1)\ \text{Minimize } \|\mathbf{x}\|_1 \ \text{ subject to } \mathbf{y}=\mathbf{A}\mathbf{x}, \text{where } \mathbf{A}\in \mathrm{R}^{m\times n},\ rank(\mathbf{A})=m \tag{14}$$

This L_1 norm minimization can be considered as a relaxed version of the L_0 problem. Fortunately, the L_1 problem is a convex optimization problem and in fact can be recast as a linear programming problem.

For example, it can be solved by an interior point method. Many effective algorithms have been developed to solve the minimum L_1 problem, and it will be considered later in this chapter. Here, we aim to study the sufficient conditions under which Eq. (12) and (14) have unique solutions. We provide a theorem related to this issue.

L_0 / L_1 **equivalence condition in SMV:**
Let $\mathbf{A} \in \mathbf{R}^{m\times n}$ be a matrix with a maximum correlation definition μ , $\mu(\mathbf{A}) = \max_{i\neq j}\left|\left\langle \mathbf{a}_i, \mathbf{a}_j \right\rangle\right|$, where $\mathbf{a}_i$ is the i th column vector of $\mathbf{A}$ with $i = 1,2,...,n$, and $\mathbf{x}$ is a k -sparse signal. Then, if $k < \frac{1}{2}\left(1+\frac{1}{\mu}\right)$ is satisfied, then the solution of L_1 coincides with that of L_0 [13] .

Table 1. L_0 / L_1 Equivalence condition in SMV.

ii) The minimization problem in MMV

To get the unique solution of MMV, it can be considered similar method with that of SMV. We introduce theorems from references [14]. To explain the uniqueness condition for MMV, we introduce the following definitions $R(\mathbf{X})$ and $relax(\mathbf{X})$.

$$R(\mathbf{X}) = \left\| m(\mathbf{x}_i)_{n\times 1} \right\|_0 \qquad (15)$$

where $x_i \in R^L$ is the transpose of the i th row of matrix $\mathbf{X}$, ie $\mathbf{X} = [x_1, x_2, ..., x_n]^T$, $m(\cdot)$ is any vector norm in R^L. Therefore, $R(\mathbf{X})$ is the number of rows which have nonzero element in matrix $\mathbf{X}$. When norm of $m(\mathbf{x}_i)_{n\times 1}$ is one, then it is defined as $relax(\mathbf{X})$.

$$relax(\mathbf{X}) = \left\| \left(m(\mathbf{x}_i)\right)_{n\times 1} \right\|_1 \qquad (16)$$

1) L_0 norm minimization in MMV

$$(L_0) \text{ Minimize } R(\mathbf{X}) = \left\| m(\mathbf{x}_i)_{n\times 1} \right\|_0 \text{ subject to } \mathbf{Y} = \mathbf{AX}, \text{ where } \mathbf{A} \in \mathrm{R}^{m\times n},\ rank(\mathbf{A}) = m \tag{17}$$

In literature [9], [14], the MMV unique solution of the L_0 minimization is known as following,

$$R(\mathbf{X}) < \frac{\text{spark}(\mathbf{A}) - 1 + rank(Cols(\mathbf{Y}))}{2} \tag{18}$$

The $rank(Cols(\mathbf{Y}))$ is the column rank of matrix $\mathbf{Y}$. If the above equation is satisfied, then the unique solution of the Eq. (8) is guaranteed.

2) L_1 norm minimization in MMV

$$(L_1) \text{ Minimize } relax(\mathbf{X}) = \left\| m(\mathbf{x}_i)_{n\times 1} \right\|_1 \text{ subject to } \mathbf{Y} = \mathbf{AX}, \text{ where } \mathbf{A} \in \mathrm{R}^{m\times n},\ rank(\mathbf{A}) = m \tag{19}$$

A sufficient condition to be the unique solution to 2) of MMV is that

$$\left\| \mathbf{A}_S^{\dagger} \mathbf{A}_j \right\|_1 < 1,\ \forall j \notin S \tag{20}$$

.

$\mathbf{A}_S$ is reduced matrix of $\mathbf{A}$ corresponding to indices from support location of $R(\mathbf{X})$. So, we can write $\mathbf{Y} = \mathbf{A}_S \mathbf{X}_S$, where matrix $\mathbf{X}_S$ is made by nonzero rows of $\mathbf{X}$. $\mathbf{A}_S$ is of full column rank. $\mathbf{A}_S^{\dagger}$ is pseudo-inverse which is defined by $\mathbf{A}_S^{\dagger} = \left(\mathbf{A}_S^T \mathbf{A}_S\right)^{-1} \mathbf{A}_S^T$. Because $\mathbf{A}_S$ is of full column rank, the generalized inverse is well defined. The above is the Exact Recovery Condition (ERC) in Tropp's "Greed is good: Algorithmic results for sparse approximation"

L_0 / L_1 **equivalence condition in MMV:**
If $R(\mathbf{X}) < \frac{spark(\mathbf{A})}{2}$ is satisfied, then the solution of L_1 in MMV coincides with that of L_0 [14] .

Table 2. L_0 / L_1 Equivalence condition in MMV.

4. Compressive sensing and its application in WSN

4.1 The usefulness of CS in WSNs

In this section, we provide a brief comparison of using CS and using the conventional compression in a WSN. This comparison illustrates why CS could be a useful solution for WSNs.

i) Sensor network scheme with conventional compression

For a conventional sensor system, the distributed sensors observe physical changes in designed area. Since each sensor observes similar physical changes, the signals observed from each sensor have much correlation. The correlated signal can be compressed for reducing data. The conventional compression for WSN requires exchanging information between distributed sensors in order to exploit inter-sensor correlation. Such a transmission strategy makes the network system complex below **Figure 5.**

The conventional compression needs to get together redundant data for compression as **Figure 6**. At the collection point, joint compression can be made and compressed information can be sent to the FC. This option has a couple drawbacks. First, gathering the samples from all the sensors and jointly compressing them cause a transmission delay. Second, a lot of onboard power should be spent at the collaboration point. Third, each sensor should be collocated so that the transmitted information can be gathered at collaboration location.

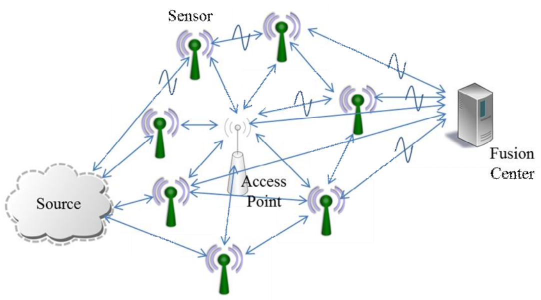


Figure 5. Conventional sensor network scheme

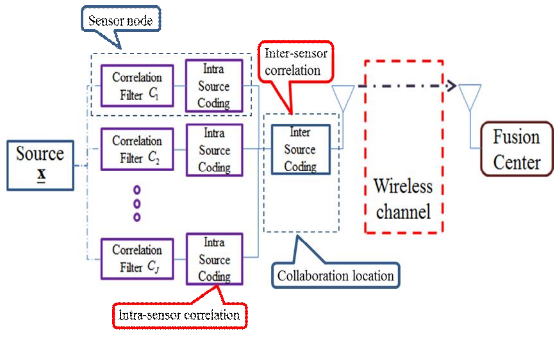


Figure 6. Conventional sensor network structure

Now, we may suppose that the joint compression is not aimed at and each sensor compresses the signal on its own. First, the data reduction effect with this approach will be limited because inter-sensor correlation is not exploited at all. The total volume of the independently compressed data is much larger than that of jointly compressed data. This may produce a large traffic volume in the WSN and a large amount of transmission power will be wasted from the sensor nodes which transmit essentially the same information to the FC. Thus, this is an inefficient strategy as well.

ii) Sensor network scheme with compressive sensing

In contrast to the conventional schemes considered in the previous paragraph, the CS method aims to acquire compressed samples directly. If a high-dimensional observation vector $\mathbf{x}$ exhibits sparsity in a certain domain (by exploiting intra-sensor correlation), CS provides the *direct method* for signal compression as discussed in **Figure 2**. To compress the high-dimensional signal $\mathbf{x}$ into a low-dimensional signal $\mathbf{y}$, as Eq. (3), it uses a simple matrix multiplication with an $m \times n$ projection matrix $\mathbf{A}_j, j \in \{1, 2, ...J\}$, where j is the sensor index, as depicted in **Figure 6**.

In the CS-based sensor network scheme, each sensor compresses the observed signals using a simple linear projection and transmits the compressed samples to the FC. Then, the FC can jointly reconstruct the received signals (by exploiting inter-sensor correlation) using one of the CS algorithms. Therefore, each sensor does not need to communicate with its neighboring sensors for joint compression. Our method is distributed compression without having the sensors to talk to each other; only the joint recovery at the FC is needed. Thus, no intermediate stages are required which are to gather all of the samples at a single location and carry out compression aiming to exploiting inter-sensor correlation as **Figure 8**. This free of intermediate stages allow us to reduce time delay significantly as well. Therefore, if the original data are compressed by CS, each sensor node produces much smaller traffic volume which can be transmitted to the FC at a much lower transmission power and with a smaller time delay. The CS sensor network structure applied for WSN is as below (**Figure 7**). You can check the simplicity of transmission strategy of CS based WSN compared with conventional network.

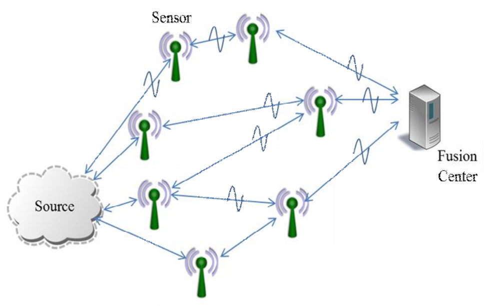


Figure 7. CS sensor network scheme

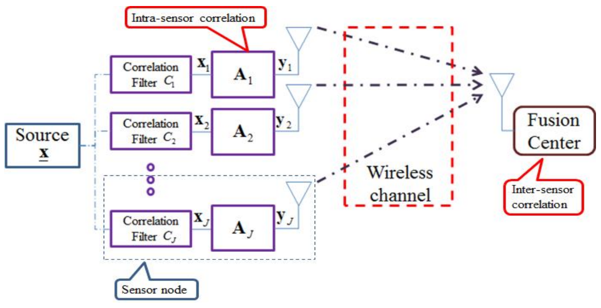


Figure 8. CS sensor network structure

4.2 Distributed compressive sensing

Each sensor can observe only the local part of an entire physical phenomenon, and a certain event of interest is measured by one or more sensors. Therefore, the sensed signals are often partially correlated. These measured signals have two distinct correlations: intra-sensor correlation and inter-sensor correlation. Intra-sensor correlation exists in the signals observed by each sensor. Once a high-dimensional sensed signal has a sparse representation in a certain domain, we can reduce its size by using CS. This process exploits the intra-sensor correlation. In contrast, inter-sensor correlation exists between the signals sensed by different sensors. By exploiting inter-sensor correlation, further reduction in transmitted signals can be made.

These two correlations can be exploited to improve the system performance. As the number of sensors in a region becomes dense, each sensor has a strongly correlated signal that is similar to that of neighboring sensors. In contrast, if we decrease the density of sensors distributed in a given region, the sensed signals will obviously be more weakly correlated with each other. In this section, we discuss two strategies for transmitting signals in a multi-sensor CS-based system. One strategy uses only intra-sensor correlation, and the other uses both types of correlation. We illustrate that CS-based system in WSN exploits the inter-sensor correlation more effectively and simply than that of conventional sensor network.

i) Exploiting only intra-sensor correlation

In **Figure 9**, each sensor observes the source signal and independently compresses it to a low-dimensional signal. After compression, each sensor transmits the compressed signal to the FC. Without exploiting inter-sensor correlation between transmitted signals, the FC recovers these signals separately. In this case, even if there exists correlation among the sensed signals, because only intra-sensor correlation is exploited, we cannot gain any advantages from joint recovery. This method has the following characteristics:

1) Independent compression and transmission at each sensor

2) Signal recovery by exploiting only intra-sensor correlation at the FC

ii) Exploiting both intra- and inter-sensor correlation

Figure 10 shows the same process as in situation i) above, except that the FC exploits the inter-sensor correlation among sensed signals at signal reconstruction stage. In conventional sensor network system as shown in **Figure 6**, the sensor nodes communicate with their neighboring sensors to take advantage of joint compression by exploiting inter-sensor correlation. However, in the CS-based system, a stage for exploiting inter-sensor correlation is achieved at FC. It means that if inter-sensor correlation exists within the sensed signals, and the FC can exploit it. This is done with sensors communicating with the FC but not among the sensors themselves. We refer to this communication strategy as the Distributed Compressive Sensing (DCS). Exploitation of inter-sensor correlation should be manifested with the reduction of the measurement size m of matrix $\mathbf{A} \in \mathbf{R}^{m \times n}$, where $\mathbf{y} = \mathbf{A}\mathbf{x}$, required for good single recovery. The characteristics of our DCS sensor network are:

1) Independent compression and transmission at each sensor

2) Exploitation of inter-sensor signal correlation with the joint recovery scheme at the FC

3) Variation of the per sensor CS measurements to manipulate the level of signal correlation

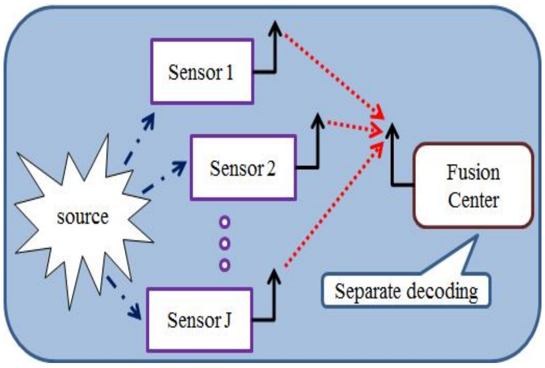


Figure 9. Intra-sensor correlation scheme

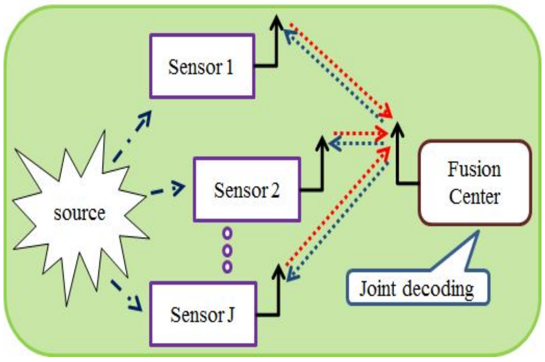


Figure 10. Intra/Inter-sensor correlation scheme

4.3. Correlated signal models

We assumed the WSN which consists of large number of sensor with a built-in CS and one fusion center. Because of the feature of considered WSN, the observed signals have inter-sensor correlation. We can model this WSN as Eq. (8) or Eq. (9). Those two equations have sparse signal matrix $\mathbf{X}$ which consists of signals transmitted from each sensor.

In this section, we introduce how the signal matrix with different degrees of correlation can be generated as sparse signal models. The sparse signal matrix in WSN has correlated properties. The degree of sparseness which is called the sparsity, is proportional to the amount of correlation. More correlated signal means sparser in terms of intra-sensor correlation. In addition, inter-sensor signal correlation can be modeled $i)$ by the degree of overlaps in the support sets of any two sparse signals, and $ii)$ by the correlation of non-zero signal values. By using those two properties, we can model correlated sparse signal matrix $\mathbf{X}$ as below examples **Figure 11**.

$$
\begin{bmatrix} &&&& \\ &&&& \\ &&&& \\ &&&& \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ &&&& \\ &&&& \\ &&&& \\ &&&& \end{bmatrix}
\begin{bmatrix} &&&& \\ &&&& \\ &&&& \\ * & * & & & * \\ &&&& \\ * & & * & * & \\ * & & & * & * \\ &&&& \\ &&&& \\ &&&& \\ &&&& \end{bmatrix}
\begin{bmatrix} * &&&& \\ * &&&& \\ & * &&& \\ & * &&& \\ && * && \\ && * && \\ &&& * & \\ &&& * & \\ &&&& * \\ &&&& * \\ &&&& * \end{bmatrix}
\begin{bmatrix} * &&&& \\ * &&&& \\ * & * &&& \\ & * &&& \\ & * & * && \\ && * && \\ && * & * & \\ &&& * & \\ &&& * & * \\ &&&& * \\ &&&& * \end{bmatrix}
\begin{bmatrix} &&& * & \\ && * && \\ * &&&& \\ & * &&& \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ & * &&& \\ && * && \\ * &&&& \\ &&&& * \end{bmatrix}
$$

$*$: Unknown nonzero value

Figure 11. The examples of correlated signals

We can divide those correlated signals in **Figure 11** as three components; common sparsity part, innovation sparsity part, and total sparsity part. The common sparsity part has one more nonzero value in the row of sparse signal matrix $\mathbf{X}$. The Innovation sparsity part has only one nonzero value in the row of signal matrix. Lastly, the total sparsity part is the total number of rows which have nonzero elements. The common sparsity is a correlated part. Therefore, if we find the location of common part, we can also use it to solve another SMV. The innovation sparsity is a uncorrelated part. Even if we find the location of innovation part, we cannot use it to solve other SMV equations. Finally, the total sparsity is related with the degree of correlation among observed signals. We re-expressed the correlated signals as following **Figure 12** by using three terminologies mentioned.

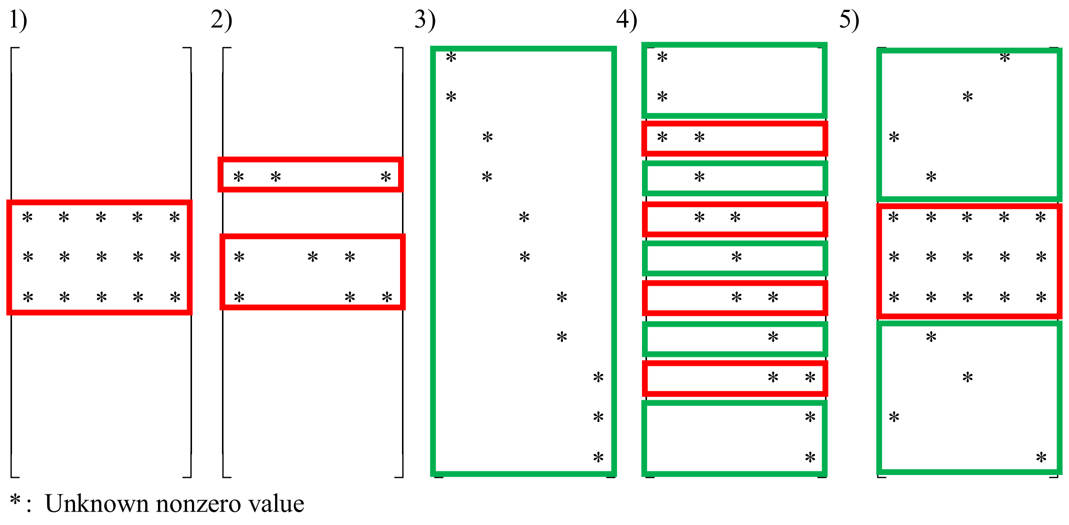


Figure 12. The components of correalted signals

The first and second correlated signals of **Figure 12** have only common sparsity part. The third signal consists of only innovation sparsity. Therefore, there is no common part which has one more nonzero element in same row. The 4), 5) signals have both common part and innovation part. If we know the prior information of the heuristic signal $\mathbf{X}$ about support location, we can use it to find solution effectively. We can use those correlated properties to recover signals transmitted from each sensor, and its reconstruction performance will increase over that not using correlated information.

We discuss the ideas for recovering those correlated signals in next section.

5. The recovery ideas for correlated signals

5.1 Joint decoding method for MMV (Literature survey)

We discussed the correlated signals which consist of common, innovation, and total parts in the previous section. The understanding of various correlated signal patterns gives more clues to get exact solution. In this section, we argue ideas using correlated information to get solution effectively.

Some specific correlated signals also are handled in [6],[7],[15]. In those references, the correlation signals are referred to as JSM-1 (joint signal model) or JSM-2 depending on the correlation type. In JSM-1, all of the signals share exactly the same common nonzero components that have the same values, whereas each signal also independently has different nonzero components, which is called innovation. In JSM-2, it shares same support location that has different value. Those two signals is expressed below, **Figure 13**. In [6],[7],[15], they proposed methods which find the solution of the correlated signals consisting of those specific pattern in Eq. (9).

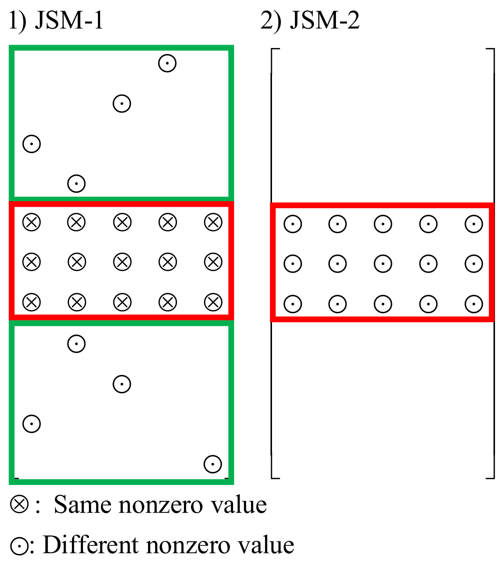


Figure 13. Joint signal models, JSM-1, JSM-2

The JSM-1 is expressed as

$$\mathbf{x}_j = \mathbf{z}_c + \mathbf{z}_j,\ j \in \{1, 2, ..., J\},\ j \text{ is the index of the sensors} \quad (21)$$

where $\|\mathbf{z}_c\|_0 = k_c$ (Common part), and $\|\mathbf{z}_j\|_0 = k_j$ (Innovation part) in each sensed signal. Obviously, $\mathbf{z}_c$ appears in all the columns of the correlated signals. It can be recognized as the inter-sensor correlation. We note that the intra-sensor correlation is that all of the signals are sparse. The j th sensor transmits $\mathbf{y}_j = \mathbf{A}_j\mathbf{x}_j$ to the FC. After all the sensed signals are transmitted to the FC, the FC aims to recover all the signals. Because inter- sensor correlation exists in the sensed signals, we can obtain several benefits by using the correlated information in the transmitted signals. For ease of explanation, suppose that the WSN contains J sensors, and its sensed signal follows JSM-1 pattern. Then, the FC can exploit both intra- and inter- sensor correlation by solving Eq. (22) as described below.

i) Modified equation method (MEM)

1) Joint decoding method for JSM-1

The sensed signals from j sensors can be expressed as follows.

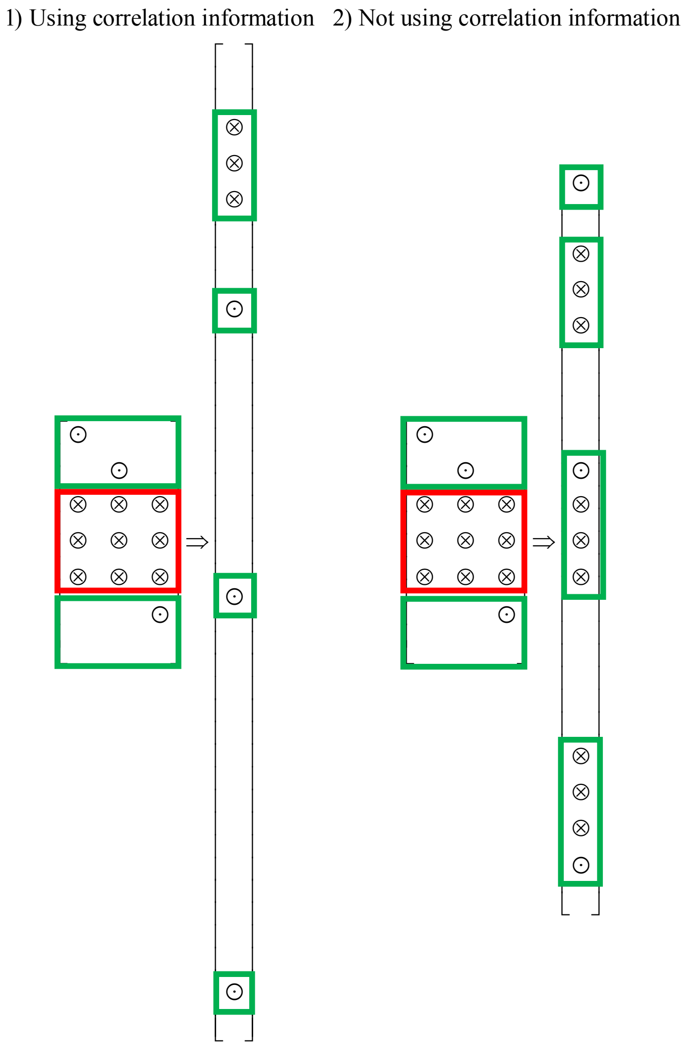
$$\begin{aligned} \mathbf{x}_1 &= \mathbf{z}_c + \mathbf{z}_1 \in \mathbf{R}^n \\ \mathbf{x}_2 &= \mathbf{z}_c + \mathbf{z}_2 \in \mathbf{R}^n \\ &\vdots \\ \mathbf{x}_J &= \mathbf{z}_c + \mathbf{z}_J \in \mathbf{R}^n \end{aligned},$$

where the sparsity of vectors $\mathbf{z}_c$ and $\mathbf{z}_j$ are k_c and k_j, respectively and each sensor has same spasity $k = k_c + k_j$. Then, the transmitted signal $\mathbf{y}_j$ can be divided into two parts as follows.

$$\mathbf{y}_j = \mathbf{A}_j(\mathbf{z}_c + \mathbf{z}_j) = \mathbf{A}_j\mathbf{z}_c + \mathbf{A}_j\mathbf{z}_j$$

If the FC received all the signals transmitted from J sensors, it then concatenates the used sensing matrix and received signal using Eq. (22). Therefore, the sensed signal in JSM-1 is transformed into **Figure 14** This idea is handled in [6],[7],[15].

$$\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \\ \vdots \\ \mathbf{y}_J \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{A}_2 & \mathbf{0} & \mathbf{A}_2 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{A}_3 & \mathbf{0} & \mathbf{0} & \mathbf{A}_3 & \mathbf{0} & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \mathbf{0} \\ \mathbf{A}_J & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_J \end{bmatrix} \begin{bmatrix} \mathbf{z}_c \\ \mathbf{z}_1 \\ \mathbf{z}_2 \\ \mathbf{z}_3 \\ \vdots \\ \mathbf{z}_J \end{bmatrix} \tag{22}$$



1) Using correlation information
2) Not using correlation information

Figure 14. Concatenating JSM-1 to a column signal

Because JSM-1 shares common part $\mathbf{z}_c$ in the equation, we can reduce the number of nonzero value as 1) of **Figure 14**. In conclusion, the total number of nonzero in matrix $\mathbf{X}$ is 12, but in transforming equation, it is 6 only. Thus, the total number of nonzero, s is reduced from $J \times \left(k_c + k_j\right)$ to $k_c + \left(J \times k_j\right)$. The total number of sparsity affects the probability of exact reconstruction. By solving this equation, the FC can take advantage of exploiting inter-sensor correlation. However, if the FC recovers the received signals independently without using any correlation information, separate recovery is done. Even if the sensed signals are correlated, separate recovery offers no advantages for signal reconstruction because it does not exploit inter- sensor correlation.

2) Separate decoding method for JSM-1

Even if a common correlated element exists in the sensed signals, separate recovery does not use that correlation information as before example. Therefore, the received signals are recovered as follows and its concatenated signal is express as 2) of **Figure 14**.

$$\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_J \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2 & \mathbf{0} & \vdots \\ \vdots & \vdots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_J \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_J \end{bmatrix} \tag{23}$$

To solve Eq. (22) and (23), we use the primal-dual interior point method (PDIP) in **Appendix 7.1**, which is an L_1 minimization algorithm, and compare the results of the two types of recovery, joint decoding and separate decoding respectively. Using the comparison results, we can confirm that the measurement size required for perfect reconstruction is smaller for joint recovery than for separate recovery.

Now, we introduce JSM-2, which is simpler than JSM-1. All the signal coefficients are different, but their indices for nonzero components are the same. Suppose that there exist two signals, $\mathbf{x}_1$ and $\mathbf{x}_2$. The i th coefficient for $\mathbf{x}_1$ is nonzero if and only if the i th coefficient for $\mathbf{x}_2$ is nonzero. This property represents inter-sensor correlation, because if we know the support set for $\mathbf{x}_1$, then we automatically know the support set for $\mathbf{x}_2$.

ii) Simultaneous orthogonal matching pursuit (SOMP)

The prior inter-correlation becomes relevant when the number of sensors is more than two. To get the advantages of exploiting inter-sensor correlation about JSM-2, we should solve the Eq. (8) and Eq. (9) jointly. Like the FC in JSM-1, the FC in JSM-2 can exploit the fact that the support set is shared. By solving the MMV jointly, we obtain several benefits as high reconstruction probability on same number of measurement. If we solve those two equations separately, but not jointly, it is separate recovery. As an algorithm for solving the equation of the JSM-2 signal, we use a simultaneous OMP (SOMP) modified from an OMP algorithm for joint decoding and apply OMP for separate decoding. These algorithms are introduced in **Appendix 7.2** and **7.3** correspondingly.

iii) One-step greedy algorithm (OSGA)

Figure 17 plots the probability of success in recovering the support set by using the one-step greedy algorithm (OSGA). OSGA finds common support location by using method described in **Table 3**, for the JSM-2 signal. In comparison with other greedy algorithms, it finds all the nonzero location at once so its performance is lower than that of SOMP.

The result of **Figure 17** suggests that the number of required measurements decreases for the same probability of exact reconstruction as the number of sensors increases. The OSGA works for a small number of measurements m if the number of sensors is sufficiently large. Therefore, if many distributed sensors observe a correlated signal, each sensor is enough to send only a small number of compressed signals to achieve perfect reconstruction probability. Consequently, the transmission power of each sensor can be reduced because only the traffic volume required for exact reconstruction, which decreases significantly, must be transmitted. However, OSGA works poorly when there are fewer sensors, so it is not good method finding correlated signals. The OSGA is described in more details in [15]

The one-step greedy algorithm (OSGA):
1. Make greedy choice: Given all of the measurements, compute the test statistics $\varepsilon_n = \frac{1}{J}\sum_{j=1}^{J}\left\langle y_j, \phi_{j,n} \right\rangle^2$ for $n \in \{1,2,...,N\}$ and estimate the common coefficient support set by $\hat{\Omega} = \{n$ having one of the k largest $\varepsilon_n\}$.

Table 3. The one-step greedy algorithm (OSGA).

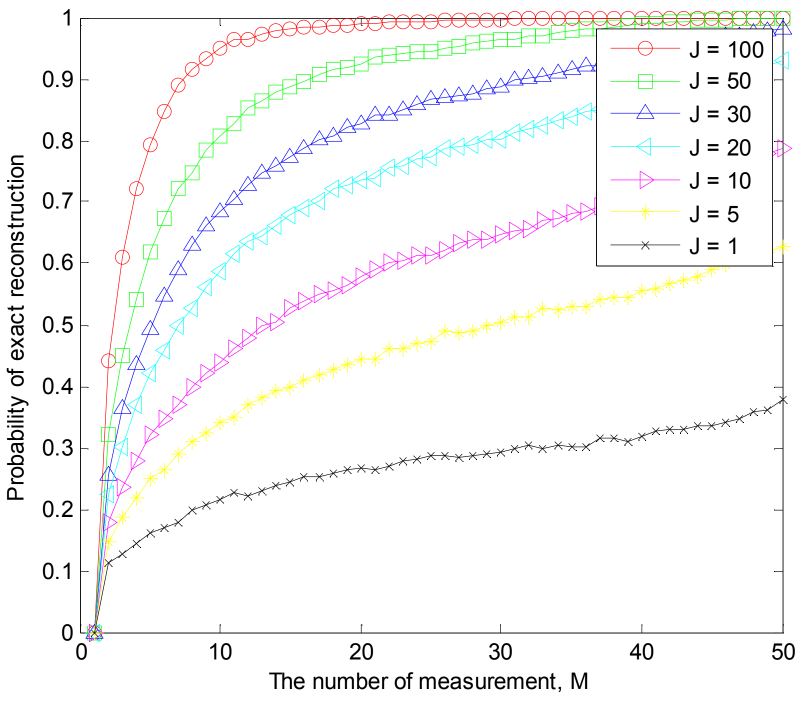


Figure 15. Reconstruction using OSGA for **JSM-2**. Experimental probability of success in recovering the support set in **JSM-2**. Signal length and sparsity are $n = 50$ and $k = 5$, respectively.

iv) Reduced MMV and boosting algorithm (ReMBo)

The Reduced and boost algorithm (ReMBo) is introduced in [9]. They reduce the correlated signal matrix $\mathbf{X}$ to one column signal and then solve reduced SMV problem by using greedy or gradient algorithm. The algorithm is summarized in **Appendix 7.4**. To get solution of MMV, ReMBo makes $\mathbf{Y} = \mathbf{AX}$ to $\mathbf{y} = \mathbf{Ax}$ by multiplying randomly generated vector $\mathbf{a}$ as **Figure 18**. The reduced SMV problem preserves the support location of the total sparsity

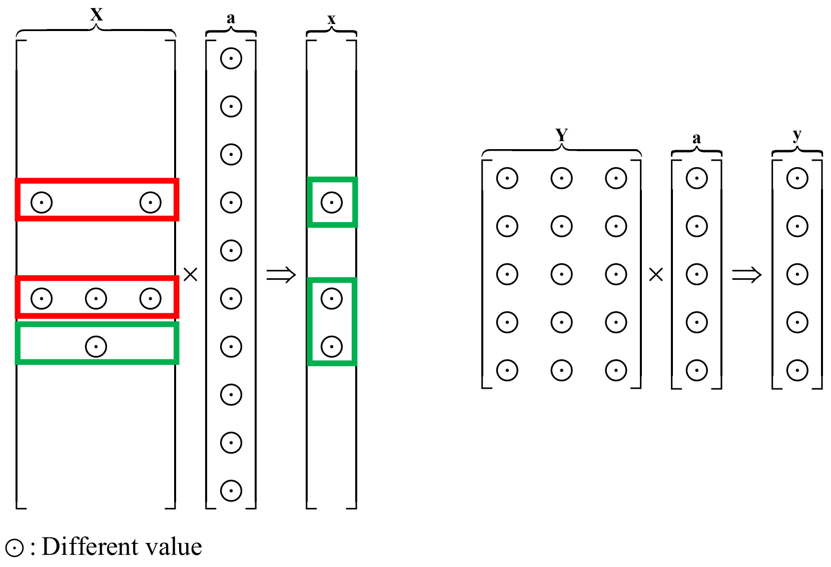
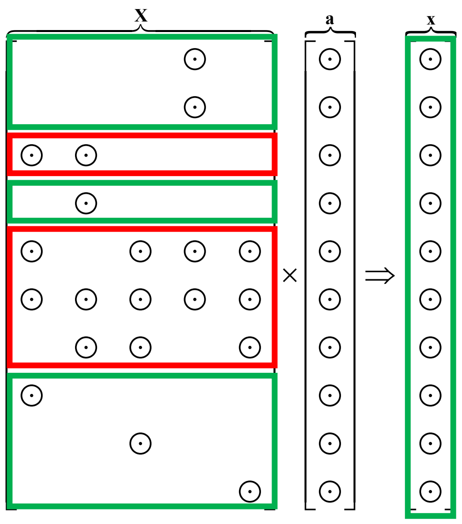


Figure 16. Reduce MMV to SMV in ReMBo

The reduced SMV can be solved by using any one of SMV algorithm and then ReMBo algorithm saves the support location of SMV solution. From the information of support set, they can get the exact solution of matrix $\mathbf{X}$. This algorithm has easy stage for understanding and the speed of this algorithm is fast and effective. However, it is possible to apply only Eq. (8) not (9), and if the original matrix $\mathbf{X}$ has much number of distributed innovation part as **Figure 19**, it cannot find solution. In the case of **Figure 19**, it makes the reduced signal $\mathbf{x}$ which is not sparse. The transformed equation $\mathbf{y} = \mathbf{Ax}$ in the example of **Figure 19** cannot be solved. Therefore, ReMBo has limitations to apply various correlated signals.



⊙ : Different value

Figure 17. The limition of ReMBo algorithm

5.2 Joint vs. separate recovery performance for JSM-1 and JSM-2

Now, we compare the results of joint recovery and separate recovery. In joint recovery, if a correlation exists between the signals observed from the distributed sensors, the FC can use the correlated information to recover the transmitted signals. In separate recovery, correlated information is not used regardless of whether a correlation pattern exists between the observed signals. In **Figure 15**, solid lines were obtained from joint reconstructions, whereas dotted lines are the results of separate reconstructions.

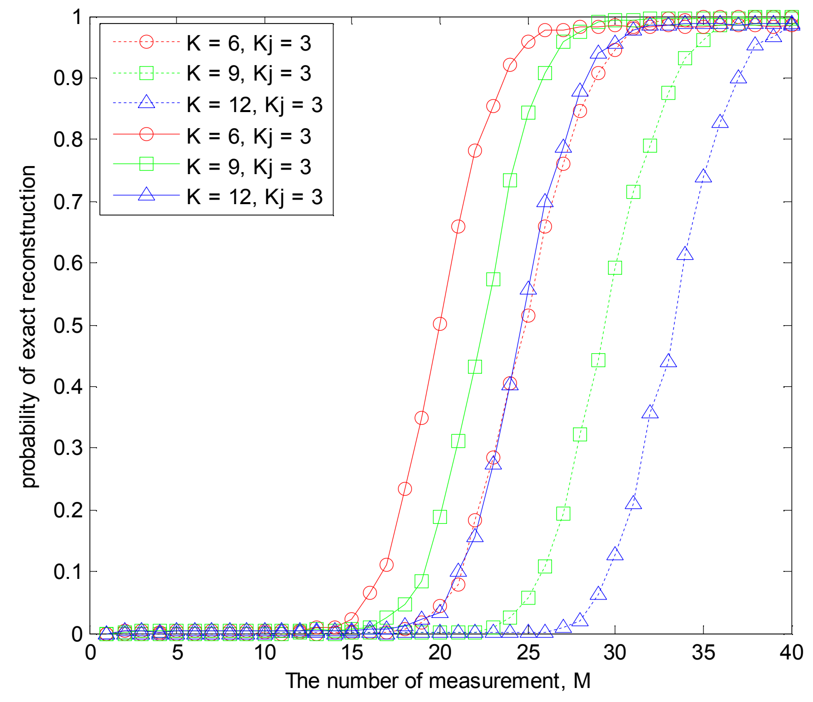


Figure 18. Joint (solid line) and separate (dotted line) reconstruction using PDIP algorithm for JSM-1. System parameters: $N = 50$, $J = 2$. The benefits of joint reconstruction depend on the common sparsity k_c.

When we use separate reconstruction, we cannot obtain any benefits from correlated information. However, when we use joint reconstruction, we can reduce the measurement size. For example, in **Figure 16**, the required number of measurements is almost 40 (dashed line and circles, $k = 6$) for perfect reconstruction when we use separate reconstruction. On the other hand, when we use joint reconstruction, it decreases to around 30 (solid line and circles, $k = 6$). Furthermore, as the common sparsity increases, the performance gap increases. For example, when the common sparsity is 9, joint reconstruction has a 90% probability of recovering all the signals at $m = 30$. However, the probability that separate reconstruction can recover all the signals is only 70%. **Figure 15** also shows that joint reconstruction is superior to separate reconstruction. For example, we need at least 30 measurements for reliable recovery using separate reconstruction when k is 6. However, we merely need at least 25 measurements for reliable recovery using joint reconstruction.

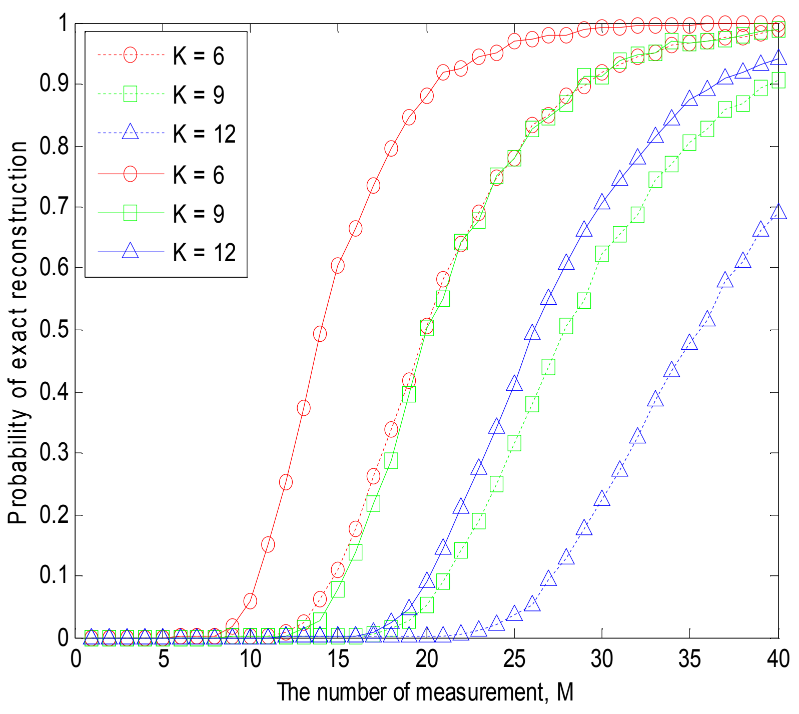


Figure 19. Joint (solid line) and separate (dotted line) reconstruction using SOMP for **JSM-2**. System parameters: $N = 50$, $J = 2$. Joint reconstruction has a higher probability of success than separate reconstruction.

5.3 Phased-Orthogonal matching pursuit (POMP)

In previous section, we discussed joint decoding methods for recovering correlated signals such as JSM-1 and JSM-2. The joint decoding method, MEM, for JSM-1 cannot be applied for signals of JSM-2 which shares same support location only, since JSM-2 does not have the same nonzero value in common part. Therefore, if we use same recovery idea for JSM-2, we cannot get the advantages of exploiting inter-sensor correlation. On the contrary, SOMP, a joint decoding algorithm for JSM-2, cannot be applied for JSM-1 which has a large number of innovation sparsity. If SOMP algorithm is used for JSM-1, it may not find solution exactly. In addition, OSGA algorithms for JSM-2 works poorly when there are fewer sensors, so it is not good method finding correlated signals. When the number of innovation sparsity is large, ReMBo algorithm may not work well.

In summary, those methods in previous section cannot be applied all of the correlated signals which have various correlated patterns. To get the exact solution of various correlated signals, we proposed joint decoding algorithm. The proposed algorithm is named to phased orthogonal matching pursuit (POMP). POMP has better performance in terms of the exact reconstruction probability of the correlated signals than previous algorithms do, for examples, PDIP, SOMP, ReMBo, etc. In this section, we explain how the proposed algorithm works to recover various correlated signals in detail and then compare the reconstruction performance of our algorithms with previous algorithms.

1) Basic idea of correlated signal recovery

We already talked about the unique solution of SMV problem before. The condition satisfied for solving SMV equation is $k < \frac{spark(\mathbf{A})}{2}$. It is proved in [12]. We used this proof as an idea for making our proposed algorithm. If the total sparsity made from matrix X satisfies $T < \frac{spark(\mathbf{A})}{2}$, it guarantees each column has unique solution. Otherwise, even though each column satisfies $k < \frac{spark(\mathbf{A})}{2}$, it doesn't mean that $T < \frac{spark(\mathbf{A})}{2}$ is satisfied due to distributed innovation part

If each column in MMV satisfies $k < \frac{spark(\mathbf{A})}{2}$, and then we can get unique solution and its support location by using separate decoding. From the support set information of first SMV problem, we can also solve next SMV problem easily by using pseudo-inverse, if the next support set is only consist of common part. Therefore, it is important to know common part information since exact common part can reduce calculation time and complexity.

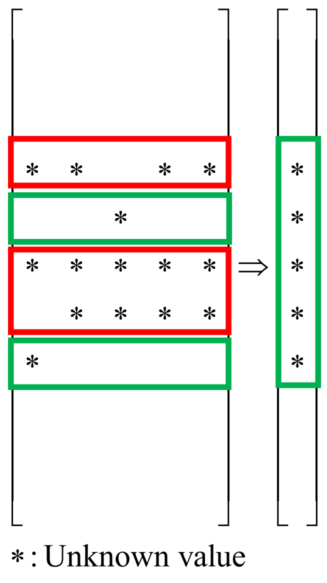


Figure 20. Total sparsity of MMV equation

The idea for proposed algorithm follows. Common sparsity is a correlated part. Therefore, if we find the location of common part, we can also use it to solve another SMV. Innovation sparsity is an uncorrelated part. Therefore, even if we know the location of innovation part, we cannot use it to recover the innovation of other signals. In SMV problem, if we want to find the unique solution, each

SMV should satisfy the unique condition. However, if we use correlated sparsity information in MMV equation, each SMV problem can get more guarantee for exact solution by using correlated information.

To use correlated information in POMP, we will find the common support location at first by using joint decoding. Then, by using separate decoding, we will find the remaining support location for each SMV problem. Therefore, we use the specific characters of both the joint decoding and separate decoding for effective reconstruction. Although the proposed algorithm works with easily understanding, its performance is better than previous methods (Modified equation method, SOMP, RemBo [6],[7],[8],[9]) for correlated signal reconstruction. In addition, it doesn't be related with the number of total sparsity.

2) Pseudo-code of POMP

POMP uses the method which finds support set at every iteration. It is similar with the OMP method about finding support set, but we applied many ideas different with original OMP. We illustrate the pseudo code of POMP algorithm as following **Table 4**

Input	Output
A $m \times n$ measurement matrix $\mathbf{A}_j$ A $m-$dimensional data vector $\mathbf{y}_j$ The sparsity level k of the ideal signal The estimate number of common sparisty C Stop conditon ε	An estimate $\hat{\mathbf{x}}_j$ in $\mathbf{R}^n$ for the ideal signal. A set $\Lambda_{j,k}$ containing k elements from $\{1,...,n\}$ An $m-$dimensional approximation $\hat{\mathbf{y}}_{j,k}$ of the data $\mathbf{y}_j$ An $m-$dimensional residual $\mathbf{r}_{j,k} = \mathbf{y}_j - \hat{\mathbf{y}}_{j,k}$

Table 4. Inputs and outputs of POMP algorithm.

```
The POMP algorithm:
Phase 1: For find common sparsity
1. Initialize:
Let the residual matrix be $\mathbf{r}_{j,0} = \mathbf{y}_{j,0}$. The sparse set $\Lambda_{j,0} = \{\}$, and iteration number $t = 1$.

2. Find the common sparsity index $\lambda_{j,t}$ for each $j$:
$\lambda_{j,t} = \arg\max_{i=1,...,n} \sum_{j=1}^{J} \left| \langle \mathbf{r}_{j,t-1}, \mathbf{a}_{j,i} \rangle \right|$. The $\mathbf{a}_{j,i}$ is the $i$ th column vector of matrix $\mathbf{A}_j$.

3. Update set:
$\Lambda_{j,t} = \Lambda_{j,t-1} \cup \{\lambda_{j,t}\}$.

4. Signal estimate:
$\mathbf{x}_{j,t}(\Lambda_{j,t}) = \mathbf{A}^{\dagger}_{j,\Lambda_t}\mathbf{y}_j$ and $\mathbf{x}_{j,t}(\Lambda^C_{j,t}) = \mathbf{0}$, where $\mathbf{x}_{j,t}(\Lambda_{j,t})$ is the set of elements whose indices are corresponding to the sparse set.

5. Get new residual:
$\hat{\mathbf{y}}_{j,t} = \mathbf{A}_{j,t}\mathbf{x}_{j,t}$, $\mathbf{r}_{j,t} = \mathbf{y}_j - \hat{\mathbf{y}}_{j,t}$.

6. Increment $t$:
Increase iteration number $t = t + 1$, and
if $t < C$ return to Step 2 of Phase 1
otherwise, $t > C$ go to Phase 2
```

```
Phase 2: For find innovation sparsity for each $j$
7. Find the index $\lambda_{j,t}$ for each $j$:
    $\lambda_{j,t} = \arg\max_{i=1,\dots,n} \left|\langle \mathbf{r}_{j,t-1}, \mathbf{a}_{j,i} \rangle\right|$ for every $j$. The $\mathbf{a}_{j,i}$ is the $i$ th column vector of matrix $\mathbf{A}_j$.
8. Update set:
    $\Lambda_{j,t} = \Lambda_{j,t-1} \cup \{\lambda_{j,t}\}$ for each $j$.
9. Signal estimate:
    $\mathbf{x}_{j,t}(\Lambda_{j,t}) = \mathbf{A}^{\dagger}_{j,\Lambda_t}\mathbf{y}_j$ and $\mathbf{x}_{j,t}(\Lambda^C_{j,t}) = \mathbf{0}$, where $\mathbf{x}_{j,t}(\Lambda_{j,t})$ is the set of elements whose indices are corresponding to the sparse set.
10. Get new residual:
    $\hat{\mathbf{y}}_{j,t} = \mathbf{A}_{j,t}\mathbf{x}_{j,t}$, $\mathbf{r}_{j,t} = \mathbf{y}_j - \hat{\mathbf{y}}_{j,t}$.
11. Increment $t$:
    Increase iteration number $t = t+1$, and
    return to Step 7 of phase 2 if $\sum_{j=1}^{J} \left\|\mathbf{y}_j - \mathbf{A}_j\mathbf{x}_j\right\|_2 > \varepsilon$
    otherwise stop the algorithm.
```

Table 5. POMP algorithm.

For **Phase 1**, it is a stage for finding common sparsity Because the common sparsity is the correlated part of the signal matrix $\mathbf{X}$, we use joint decoding method for finding the location of common part. The joint decoding method is able to find support location successfully. We already knew the advantages of joint decoding from the comparison of joint decoding and separate decoding in **Section 5.1**. Therefore, if it is possible to use joint decoding for MMV equation, we should use it for advantages about signal reconstruction. It results in better performance for solving MMV equation. According to pseudo-code of POMP, it finds the location of common part in **Phase 1** and memorizes the index as **Figure 21**. After the stage of **Phase 1** is finished, POMP algorithm tries to find the remaining support set by separate decoding. Due to separate decoding of **Phase 2** for remaining support set, POMP can find the missed common sparsity in previous stage.

We draw **Figure 21** which expresses the movement of POMP algorithm. The nonzero values in red box are common sparsity which is exploited in **Phase 1** and then the remaining nonzero values in green box can be exploited in **Phase 2**. The index of row having nonzero values is added to Λ at every iteration until the criteria $\sum_{j=1}^{J} \left\| \mathbf{y}_j - \mathbf{A}_j \mathbf{x}_j \right\|_2 \le \varepsilon$ is satisfied. After finishing the movement of POMP, we can get the original solution by using pseudo-inverse based on the estimated support set.

Support set in **Phase 1**:
j = 1, {3, 5, 6, 7}
j = 2, {3, 5, 6, 7}
j = 3, {3, 5, 6, 7}
j = 4, {3, 5, 6, 7}
j = 5, {3, 5, 6, 7}

Support set in **Phase 2**:
j = 1, {3, 5, 6, 7, 8}
j = 2, {3, 4, 5, 6, 7}
j = 3, {3, 5, 6, 7, 9}
j = 4, {1, 2, 3, 5, 6, 7}
j = 5, {3, 5, 6, 7, 10}

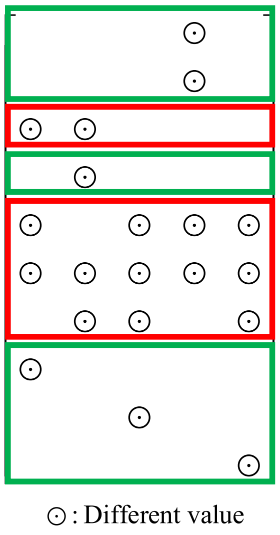


Figure 21. The movement of POMP algorithm

Moore-Penrose pseudo inverse:

If we define that Λ_j is the support set of the j th column in matrix $\mathbf{X}$, we can reduce the sensing matrix $\mathbf{A}_j$ to $\mathbf{A}_{\Lambda_j}$ corresponding to the nonzero elements of $\mathbf{x}_j$. If the columns of the reduced matrix $\mathbf{A}_{\Lambda_j}$ are linearly independent, Moore-Penrose pseudo inverse equation is accepted.

$$\left(\mathbf{A}_{\Lambda_j}\right)^{\dagger} \mathbf{A}_{\Lambda_j} = \mathbf{I}, \text{ where } \mathbf{A}_{\Lambda_j}^{\dagger} = \left(\mathbf{A}_{\Lambda_j}^{T} \mathbf{A}_{\Lambda_j}\right)^{-1} \mathbf{A}_{\Lambda_j}^{T}$$

Therefore, if we know the support set and the reduced matrix $\mathbf{A}_{\Lambda_j}$ are linearly independent, then the original signal $\mathbf{x}_j$ can be found by using pseudo-inverse.

$$\mathbf{x}_{\Lambda_j} = \left(\mathbf{A}_{\Lambda_j}^T \mathbf{A}_{\Lambda_j}\right)^{-1} \mathbf{A}_{\Lambda_j}^T \mathbf{y}$$

Table 6. Moore-Penrose pseudo inverse.

5.4 The properties of POMP algorithm

i) The advantages of using prior correlation information

If we know the prior information of correlated signal like the number of common sparstiy, innovation sparsity, or the distribution of support location, we can use that information for signal recovery. If we know the number of common sparsity as prior information, we can choice parameter C as the number of iteration used for finding common part exactly. To select the number of iteration exactly in POMP affects the reconstruction performance as **Figure 22**. The parameters of simulation are $N = 150$, $C = 10$, $I = 10$. Even though the value of estimated C is not exact correct as red and or blue, its performance is stable. However, it requires much number of measurements for perfect signal recovery.

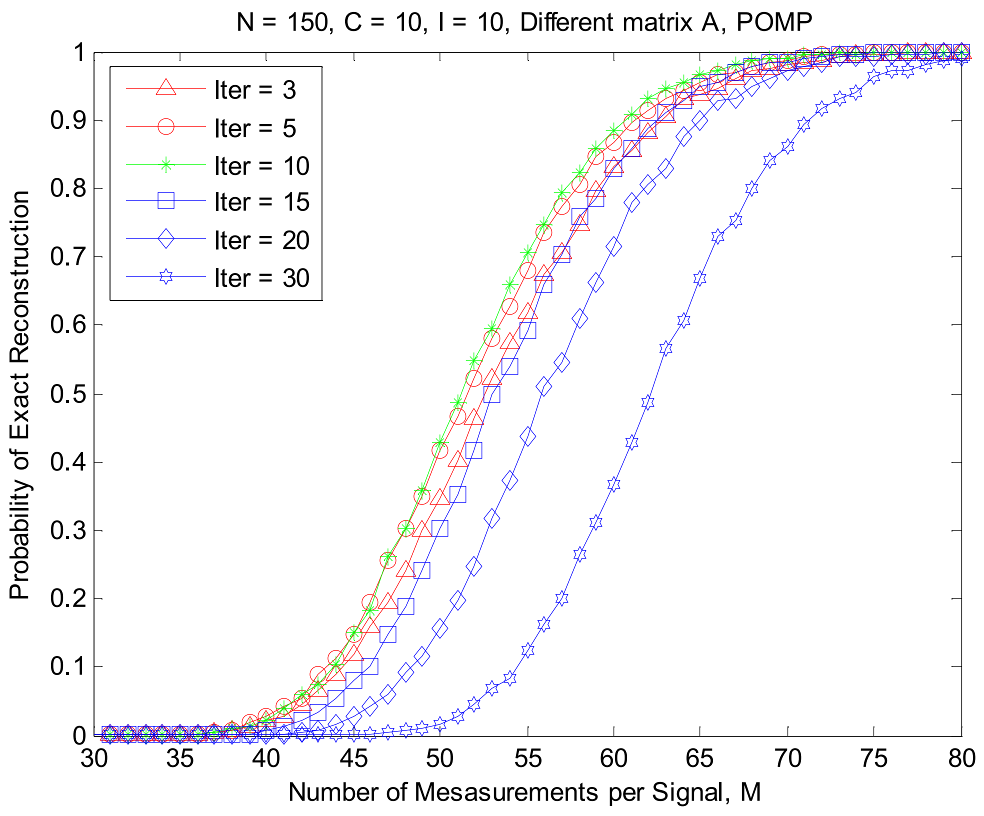


Figure 22. The advantages of using prior information

ii) The stable performance for various correlated signal model.

POMP algorithm has two stages for finding the support location of every sensor. By using those two stages, POMP can find exact solution, if the given number of measurements is enough as the graphs of **Figure 23**. We generated various correlated signals which have different number of common sparsity C and innovation sparsity I. The first stage of POMP uses joint decoding method for finding correlated support set. Therefore, if the correlated signals include more common sparsity, its reconstruction performance of POMP increases as using joint decoding method. The second stage of POMP uses separate decoding method for finding uncorrelated support set. Although it cannot get any advantages about using inter-sensor correlation, it guarantees the problem to be solved perfectly under the enough number of measurements.

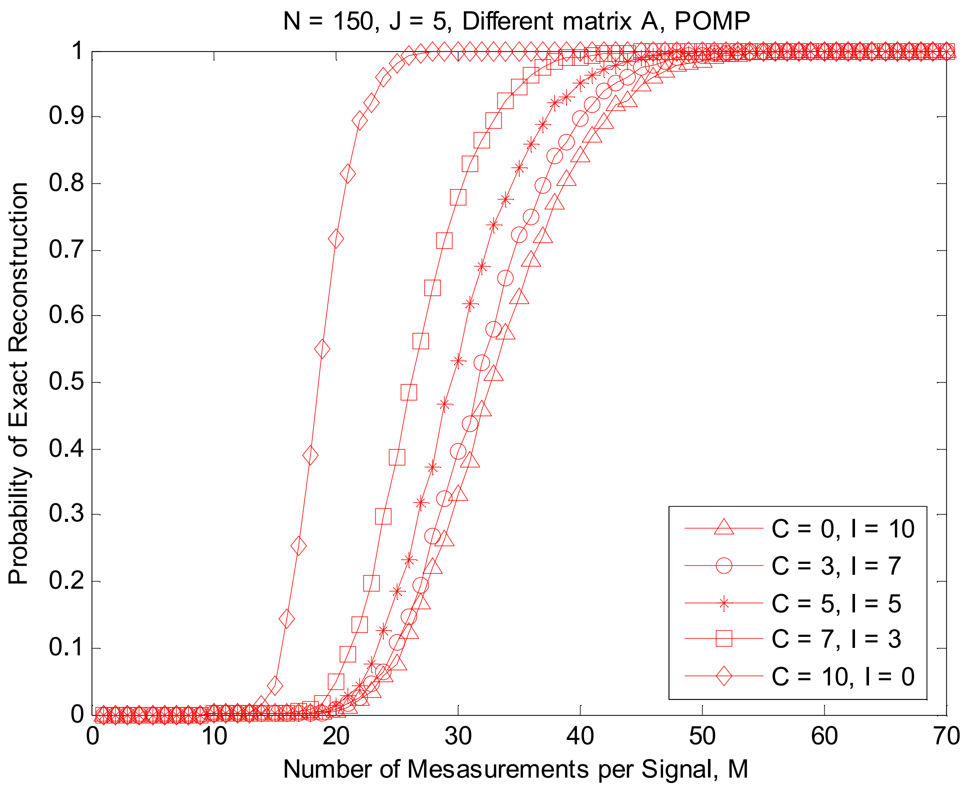


Figure 23. The stable performance of POMP algorithm

iii) The complexity of POMP

1) The complexity of Moore-Penrose pseudo inverse

We express how POMP works for MMV equation throughout pseudo-code. According to POMP algorithm, it requires pseudo-inverse calculation to get unique solution when we know the support set of original signal. In general, the calculation of matrix multiplication and inverse matrix require high complexity respectively $O(n^2)$ and $O(n^3)$ respectively. However, we already know that the number of sparsity k is very short in comparison with the length of signal n as $k << m < n$ so, the complexity of those calculations is simple as below **Table 7**.

The complexity of pseudo-inverse:

By using pseudo-inverse, we can get the nonzero values corresponding to support set. The pseudo inverse complexity is not high. From the relation $k << m < n$, we can get the complexity.

$$\left(\underbrace{A_S^T}_{k\times m}\underbrace{A_S}_{m\times k}\right)^{-1}\underbrace{A_S^T}_{k\times m}\underbrace{y}_{m\times 1} = x_S$$

Multiplication: $k \times m \times k + k \times k \times m + k \times m \times 1 \approx 2k^2m + km \approx O(mk^2)$

Inverse: $O(k^3)$. Therefore, total:$O(k^3) + O(mk^2) \approx O(mk^2) \quad \because k << m$

Table 7. Pseudo-inverse in MMV.

2) The complexity of POMP algorithm in terms of sparsity.

POMP algorithm has two stages for find support location which consists of common sparsity and innovation sparsity. In this section, we analyze the complexity of POMP algorithm related with sparsity and the number of sensors J. We assumed that the observed signal has both the C number of common sparstiy and the I number of innovation sparsity. In **Phase 1**, it will find the C number of sparsity and it requires such as calculations, for examples $\lambda_{j,t} = \arg\max_{i=1,\ldots,n} \sum_{j=1}^{J} \left| \langle \mathbf{r}_{j,t-1}, \mathbf{a}_{j,i} \rangle \right|$ and

$\mathbf{x}_{j,t}\left(\Lambda_{j,t}\right)=\mathbf{A}^{\dagger}_{j,\Lambda_t}\mathbf{y}_j$. After finishing the calculations of **Phase 1**, it starts the calculation of **Phase 2** for finding innovation sparsity. We consider $\lambda_{j,t}=\arg\max_{i=1,\dots,n}\left|\left\langle \mathbf{r}_{j,t-1},\mathbf{a}_{j,i}\right\rangle\right|$ and $\mathbf{x}_{j,t}\left(\Lambda_{j,t}\right)=\mathbf{A}^{\dagger}_{j,\Lambda_t}\mathbf{y}_j$ in **Phase 2**. The complexity of POMP algorithm is below.

The complexity of POMP

Phase 1

Considered parameter: signal length n, measurement m, sparsity k, the number of sensors J, common sparstiy C, innovation sparsity I. We already know the relationship $(C \approx I) < k << m < n$

1) $\lambda_{j,t}=\arg\max_{i=1,\dots,n}\underbrace{\sum_{j=1}^{J}\left|\left\langle \underbrace{\mathbf{r}_{j,t-1}}_{1\times m},\underbrace{\mathbf{a}_{j,i}}_{m\times 1}\right\rangle\right|}_{J\text{ times summation}}$

Inner product and sigma summation: $O(m)+O(J)\approx O(m)$ $\because$ In general $J<m$

By $J\times n$ iteration: $JnO(m)=O(Jnm)$

2) $\mathbf{x}_{j,t}\left(\Lambda_{j,t}\right)=\mathbf{A}^{\dagger}_{j,\Lambda_t}\mathbf{y}_j$

Pseudo-inverse: $O\left(mk^2\right)$

3) The number of iteration: C

Therefore, $C\left(O(Jmn)+O\left(mk^2\right)\right)\approx O(CJmn)$ $\because$ In general $k^2<n$

In conclusion, the complexity of **Phase 1** is $O(CJmn)$

Phase 2

1) $\lambda_{j,t}=\arg\max_{i=1,\dots,n}\left|\left\langle \mathbf{r}_{j,t-1},\mathbf{a}_{j,i}\right\rangle\right|$

Inner product : $O(m)$

By $J\times n$ iteration: $JnO(m)=O(Jnm)$

2) $\mathbf{x}_{j,t}\left(\Lambda_{j,t}\right)=\mathbf{A}^{\dagger}_{j,\Lambda_t}\mathbf{y}_j$

Pseudo-inverse: $O\left(mk^2\right)$

3) The number of iteration: I

Therefore, $I\left(O(Jmn)+O\left(mk^2\right)\right) \approx O(IJmn)$ $\because$ In general $k^2 < n$

In conclusion, the complexity of **Phase 2** is $O(IJmn)$

By **Phase 1** + **Phase 2**,

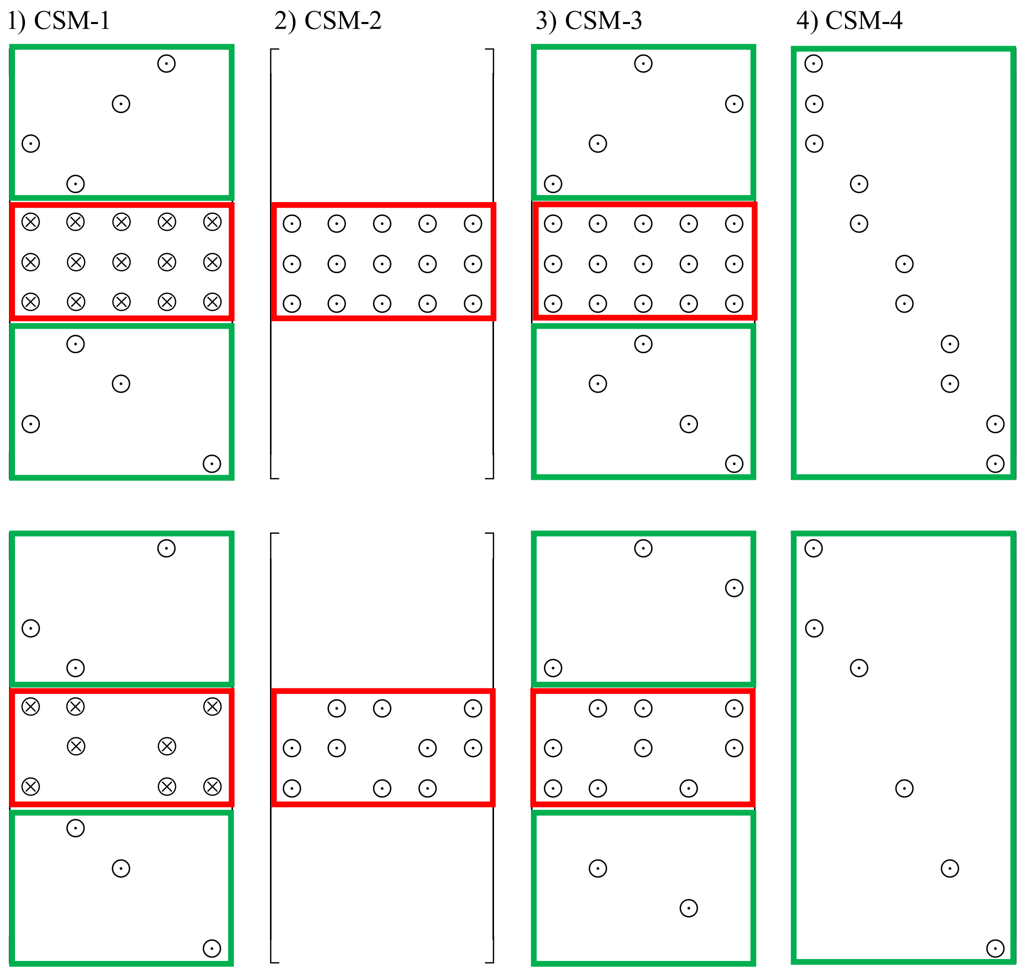
In conclusion, the complexity of POMP is $O(CJmn)+O(IJmn)=O\left(Jmn(C+I)\right) \approx O(Jmnk)$.

The complexity of POMP algorithm is affected by the parameters J, m, n, k .

Table 8. The complexity of POMP algorithm

6. Performance evaluation

In previous section, we already discuss the various correlated signals which are handled with references. For distinguishing those correlated signals, we named it correlated signal model (CSM) as following **Figure 23** and then we solved MMV equation (8) and (9) by using algorithms like modified equation method (MEM), SOMP, ReMBo, POMP. The movement of all the algorithms is handled in previous section.



⊗ : Same nonzero value

⊙ : Unknown nonzero value

Figure 24. Correlated signal model (CSM)

In **Figure 23**, we define correlated signal model which has various kinds of pattern. Most of the signal pattern can be defined by our CSM. CSM-1 and CSM-2 is similar with JSM-1 and JSM-2 which are handled in [6],[15] but we consider the case that has some vacancies in common part and innovation part and we changed the definition of innovation sparsity. In our paper, innovation sparsity has only one nonzero value in same row in different with [6],[15]. CSM-1 and CSM-3 have common part and innovation part together but CSM-1 has same value for common part. CSM-2 has only common part and CSM-4 has only innovation part. Other case which does not exist in **Figure 23** will not be considered in this paper. Now, we simulate the performance POMP algorithm compared with other methods as ReMBo, SOMP, MEM. Because some algorithms do not take advantages of joint decoding with specific signal model and sensing matrix, we summarized it as **Table 9**.

Signal model / Algorithm		Signal Type			
		CSM-1	CSM-2	CSM-3	CSM-4
MEM	Different A	O	X	X	X
	Same A	X	X	X	X
SOMP	Different A	△	O	△	X
	Same A	△	O	△	X
ReMBo	Different A	X	X	X	X
	Same A	△	O	△	X
POMP	Different A	O	O	O	X
	Same A	O	O	O	X

(O: It can get joint decoding advantages, △: It can get joint decoding advantages conditionally, X: It is impossible to get joint decoding advantages)

Table 9. The performance of various algorithms with related to correlated signals.

1) Correlated signal model -1

We generated the CSM-1 and different matrix **A** at each sensor and then we simulated the reconstruction performance for MEM, SOMP, and POMP algorithm. When the correlated signal has common part which has same value, the idea of MEM can take much advantages of joint decoding as **Figure 24**. Although the performance of POMP is not better than that of MEM, POMP can get the joint decoding advantages also. When the number of sensors increases in the case of POMP and MEM, the needed number of measurement for perfect reconstruction decreases. It means that those two methods can take advantages of joint decoding. However, the case of SOMP is not. When the number of sensors increases, the performance of SOMP becomes worst. The reason is that SOMP cannot find the innovation sparsity perfectly.

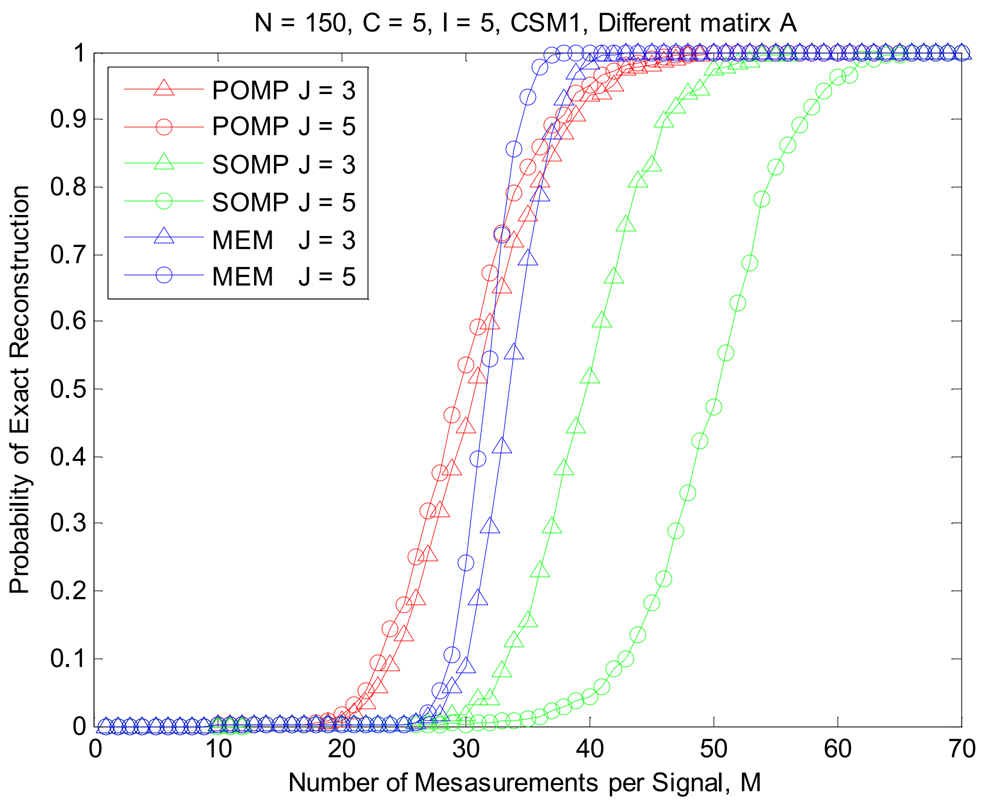


Figure 25. CSM-1 and Different matrix **A**

When same sensing matrix **A** is used at each sensor, then MEM methods cannot be used. In addition, the number of sensors increase, MEM is not a proper method for signal reconstruction. Because MEM method makes large equation which has much number of variables, the speed of algorithm is very slow. Therefore, we simulated only three algorithms, POMP, SOMP, ReMBo. The result is as following **Figure 25.** SOMP and ReMBo algorithm does not show stable performance in terms of increasing sensors. In contrast, the proposed algorithm can recover the CSM-1 perfectly when the number of measurement is enough. Moreover, it can get the advantages of joint decoding method.

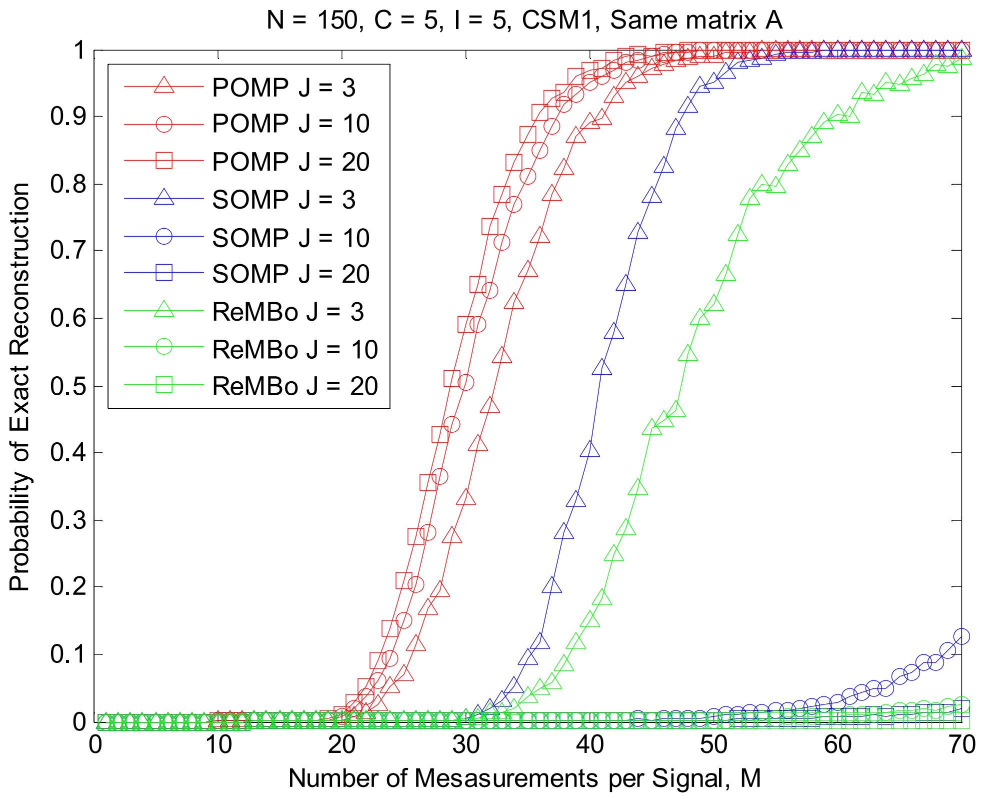


Figure 26. CSM-1 and Same matrix **A**

2) Correlated signal model -2

CSM-2 has only common sparsity which has different value. We used SOMP, POMP algorithm for CSM-2 signal recovery. MEM algorithm does not take any advantages of joint decoding with the signal pattern of CSM-2 and ReMBo algorithm cannot apply for different sensing matrix **A** at each sensor. Therefore, we did not consider those two methods for this simulation. In this simulation, SOMP can work best when the signal pattern has CSM-2. However, the performance of POMP algorithm is better than that of SOMP because the **Phase 2** of POMP can find the remaining support set which is not exploited in the stage of **Phase 1**. If we increase the number of sensor to infinity, the minimum number of measurement for perfect reconstruction converges to the number of sparsity. In this simulation, the two algorithms converge to 10.

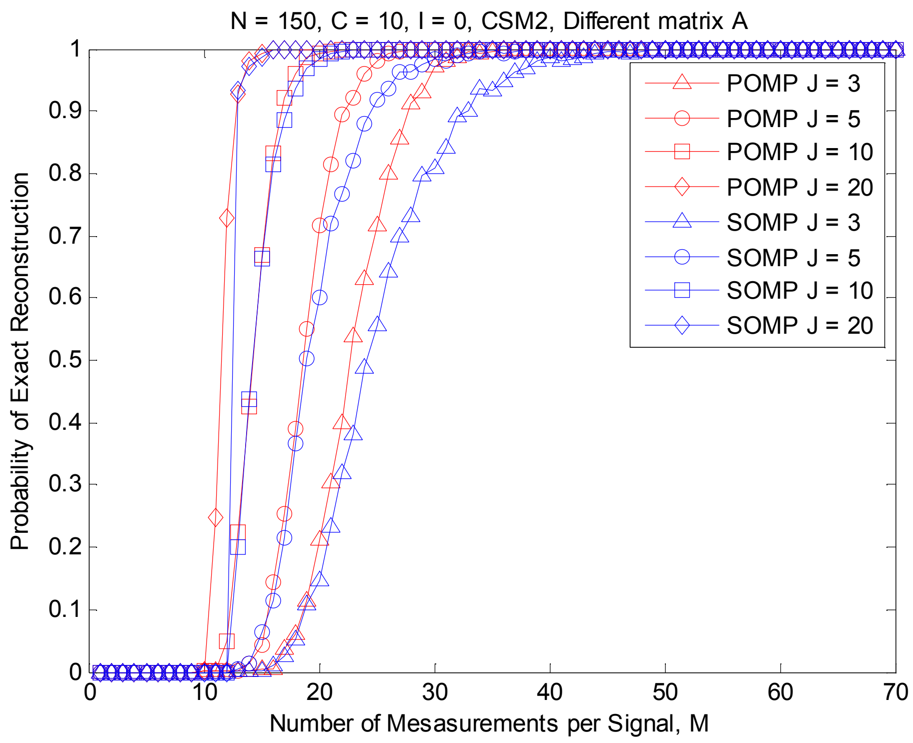


Figure 27. CSM-2 and Different matrix **A**

In this case, we solved MMV problem which has same matrix A at each sensor. It means that the number of equation is same regardless of the number of sensors. In contrast, if the number of sensors increases, the number of unknowns also increases. It makes the given equation to be not solved easily.

Since CSM-2 model has only common sparsity, ReMBo algorithm shows stable performance regardless of the number of sensors. If the information of total sparsity can be preserved by using ReMBo, its performance is guaranteed. In the case of POMP algorithm, it also shows stable reconstruction performance regardless the number of sensors. Since the stage of **Phase 2** can find the missed support set at each sensor, it does not show performance deterioration as the result of SOMP in **Figure 27**.

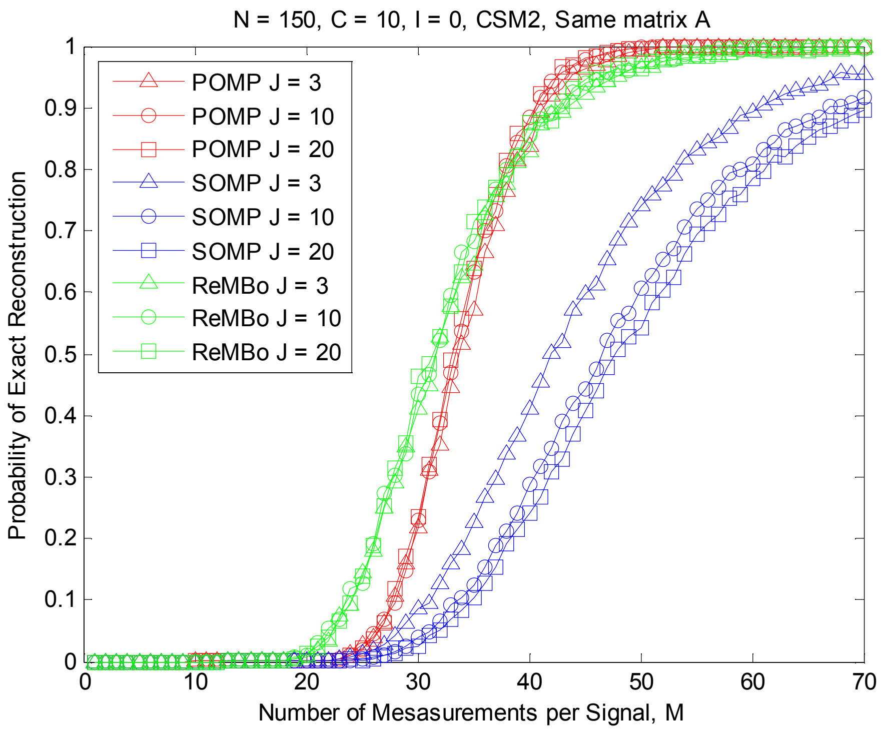


Figure 28. CSM-2 and Same matrix **A**

3) Correlated signal model -3

In this case, as the number of sensor J increase, the innovation sparsity also increases considerably. When the number of J is 20, then the total sparsity is 105. Since SOMP algorithm can find common support set which each sensor has commonly, the much number of total sparsity caused from innovation part makes error for finding exact solution. However, POMP algorithm finds the common sparsity at first and then uses separate decoding for all the sensors. As results, its performance is stable as **Figure 28**. Even if original signal has innovation sparsity for each sensor, POMP algorithm can get the advantages of joint decoding methods. As the number of sensors increases, the needed number of measurements for perfect reconstruction decreases.

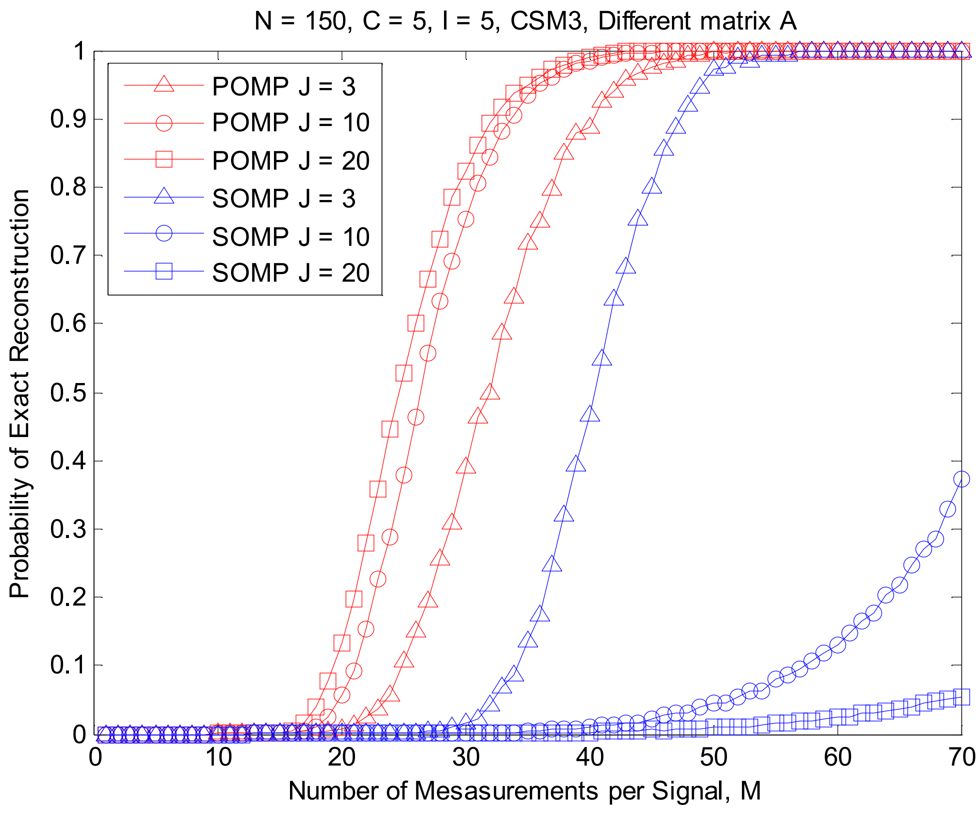


Figure 29. CSM-3 and Different matrix **A**

7. Conclusion

In this paper, we discussed the application of compressive sensing (CS) for wireless sensor networks (WSNs). We assumed a WSN consisting of spatially distributed sensors and one fusion center (FC). The sensor nodes take signal samples and pass their acquired signal samples to the FC. When the FC receives the transmitted data from the sensor nodes, it aims to recover the original signal waveforms, for later identification of the events possibly occurring in the sensed region. (Section 2.1)

We discussed that CS is the possible solution which provides simpler signal acquisition and compression. CS is suitable for the wireless sensor networks since it allows removal of intermediate stages such as sampling the signal and gathering the sampled signals at one collaboration point which would usually be the case in a conventional compression scheme. Using CS, the amount of signal samples that need to be transferred to the FC from the sensors can be significantly reduced. This may lead to reduction of power consumption at the sensor nodes, which was discussed in Section 4.1. In summary, each sensor with CS can save power by not needing to run complex compression operations on board and by cutting down signal transmissions.

Distributed sensors usually observe a single globally occurring event and thus the observed signals are often correlated with each other. We considered two types of correlations: intra- and inter-sensor signal correlation in Section 4.2. We provided the sparse signal models which encompass both types of correlation and explained the components existing in correlated signal by using common sparsity, innovation sparsity, total sparsity. (Section 4.3)

The FC receives the compressed signals from the sensors. The FC then recovers the original signal waveforms from the compressed signals using a CS recovery algorithm. We considered various types of algorithms for joint decoding (OSGA, PDIP, SOMP, ReMBo). A CS recovery algorithm is referred to as the joint recovery scheme when it utilizes inter-sensor signal correlation as well. In contrast, when the inter-sensor signal correlation is not utilized, it is referred to as the separate recovery scheme. In the joint recovery scheme, correlation information is used for taking advantages of joint recovery as shown Eq. (22). In the separate recovery scheme, a sensor signal recovery is done individually and independently from the recovery of other sensor signals. We compared the results of the joint recovery with those of the separate recovery scheme. We have shown that correlation information can be exploited by using joint recovery and the number of measurements needed for exact reconstruction can be significantly reduced as shown in **Figure 18** and **Figure 19**. It means that the traffic volume transmitted from the sensors to the FC can decrease significantly without degrading the quality of the recovery performance. (Section 5)

Finally, we proposed POMP algorithm which find support set throughout 2 stages. It uses joint decoding method and separate decoding method properly. We showed how the proposed algorithm works and analyzed the complexity of POMP algorithm. Its performance outperform in compared with previous algorithms as MEM, ReMBo, SOMP. In addition, it shows stable performance regardless of the pattern of various correlated signals. (Section 6)

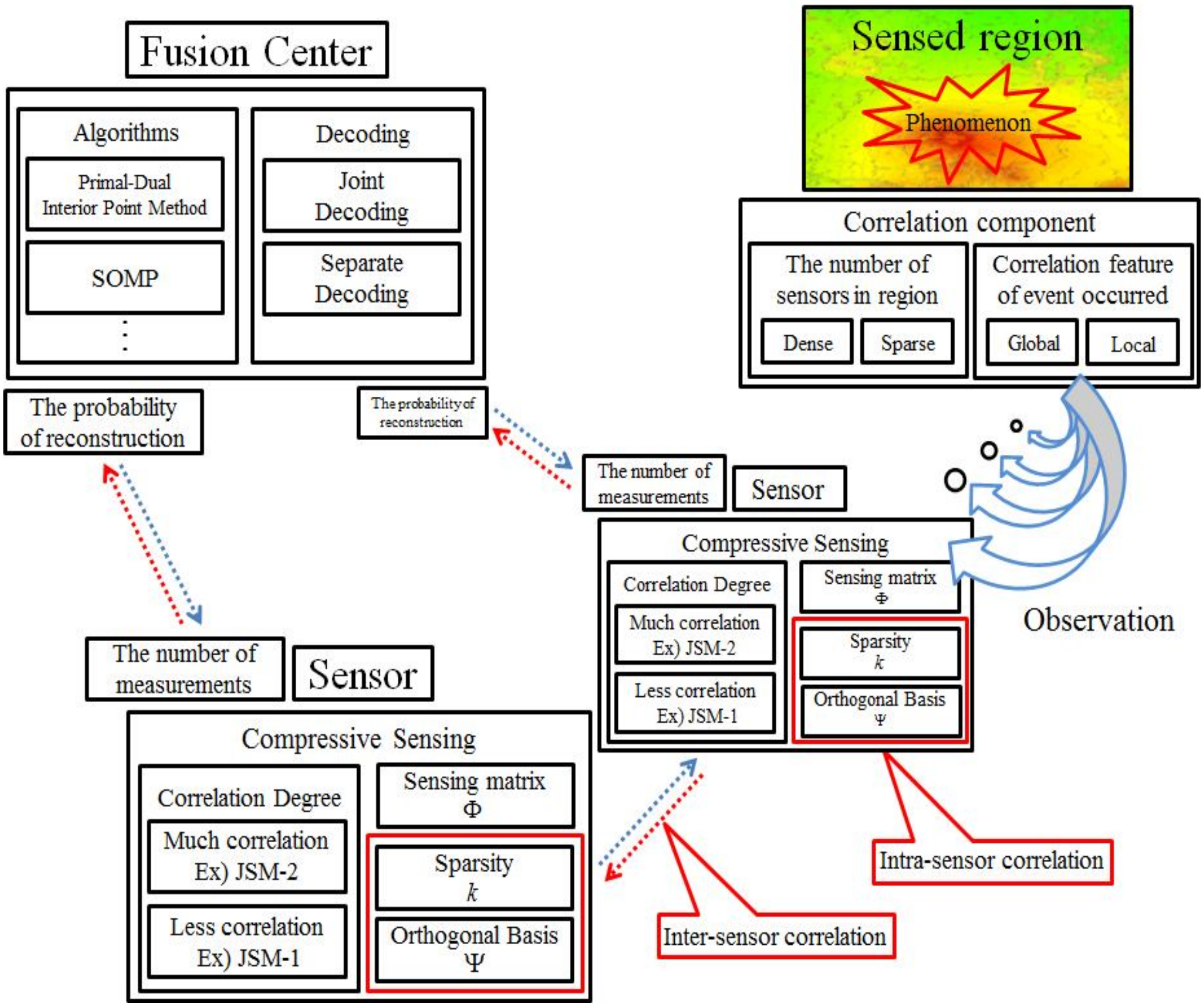


Figure 30. Summary of CS application in WSN

8. Appendix

8.1. Primal-dual interior point method (PDIP)

The L_1 minimization in Eq. (14) can be recast as linear programming. Here we examine this relationship. Clearly, the L_1 minimization problem in Eq. (14) is not linear programming because its cost function is not linear. However, by using a new variable, we can transform it to linear programming. Thus, the problem that we want to solve is

$$\begin{gathered}\min_{(x,u)} \sum_i u_i \\ \text{subject to} \\ \forall_i \left|x(i)\right| \le u_i \\ \mathbf{Ax} = \mathbf{b}\end{gathered} \tag{24}$$

The solution of the above equation is equal to the solution of the L_1 minimization problem [16]. Many approaches to solving Eq. (24) have been studied and developed. Here, we discuss the primal-dual interior point (PDIP) method, which is an example of gradient-type algorithms. First, we have the Lagrangian function of Eq. (24), as follows:

$$L(\mathbf{t}, \boldsymbol{\lambda}, \mathbf{v}) = \begin{bmatrix}\mathbf{0}_1^{\mathrm{T}} & \mathbf{1}^{\mathrm{T}}\end{bmatrix}\mathbf{t} + \mathbf{v}^{\mathrm{T}}\left(\begin{bmatrix}\mathbf{A} & \mathbf{0}_2\end{bmatrix}\mathbf{t} - \mathbf{b}\right) + \boldsymbol{\lambda}^{\mathrm{T}}\left(\begin{bmatrix}\mathbf{e} & -\mathbf{e} \\ -\mathbf{e} & -\mathbf{e}\end{bmatrix}\mathbf{t}\right) \tag{25}$$

where $\mathbf{e}$ is the $n \times n$ identity matrix, $\mathbf{0}_1$ is the zero vector, $\mathbf{0}_2$ is the $m \times n$ zero vector, and $\mathbf{1}$ is the $n \times 1$ vector whose elements are all one, $\mathbf{t} := \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \end{bmatrix} \in \mathbf{R}^{2n\times 1}$, $\mathbf{v} \in \mathbf{R}^{m\times 1}$, and $\boldsymbol{\lambda} \in \mathbf{R}^{2n\times 1} \geq 0$. From the Lagrangian function, we have several KKT conditions,

$$\begin{aligned} &\begin{bmatrix} \mathbf{0} \\ \mathbf{1} \end{bmatrix} + \begin{bmatrix} \mathbf{A}^{\mathrm{T}} \\ \mathbf{0}^{\mathrm{T}} \end{bmatrix} \mathbf{v}^* + \begin{bmatrix} \mathbf{e} & -\mathbf{e} \\ -\mathbf{e} & -\mathbf{e} \end{bmatrix} \boldsymbol{\lambda}^* = \mathbf{0}_3 \\ &\begin{bmatrix} \mathbf{A} & \mathbf{0}_2 \end{bmatrix} \mathbf{t}^* - \mathbf{b} = \mathbf{0}_4 \\ &\begin{bmatrix} \mathbf{e} & -\mathbf{e} \\ -\mathbf{e} & -\mathbf{e} \end{bmatrix} \mathbf{t}^* \leq \mathbf{0}_1 \\ &\left(\boldsymbol{\lambda}^*\right)^{\mathrm{T}} \begin{bmatrix} \mathbf{e} & -\mathbf{e} \\ -\mathbf{e} & -\mathbf{e} \end{bmatrix} \mathbf{t}^* = 0, \boldsymbol{\lambda}^* \geq \mathbf{0}_3 \end{aligned} \qquad (26)$$

where $\mathbf{0}_3$ is the $2n \times 1$ zero vector, and $\mathbf{0}_4$ is the $m \times 1$ zero vector. The main point of the PDIP is to seek the point $\left(\mathbf{t}^*, \boldsymbol{\lambda}^*, \mathbf{v}^*\right)$ that satisfies the above KKT conditions. This is achieved by defining a mapping function $\mathrm{F}\left(\mathbf{t}, \boldsymbol{\lambda}, \mathbf{v}\right) : \mathbf{R}^{(2n+m)\times 1} \to \mathbf{R}^{(2n+m)\times 1}$, which is

$$\mathrm{F}\left(\mathbf{t}, \boldsymbol{\lambda}, \mathbf{v}\right) = \begin{bmatrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \end{bmatrix} + \begin{bmatrix} \mathbf{A}^{\mathrm{T}} \\ \mathbf{0}^{\mathrm{T}} \end{bmatrix} \mathbf{v} + \begin{bmatrix} \mathbf{e} & -\mathbf{e} \\ -\mathbf{e} & -\mathbf{e} \end{bmatrix} \boldsymbol{\lambda} \\ \left(\boldsymbol{\lambda}^*\right)^{\mathrm{T}} \begin{bmatrix} \mathbf{e} & -\mathbf{e} \\ -\mathbf{e} & -\mathbf{e} \end{bmatrix} \mathbf{t}^* \\ \begin{bmatrix} \mathbf{A} & \mathbf{0}_2 \end{bmatrix} \mathbf{t} - \mathbf{b} \\ \end{bmatrix} = \mathbf{0}_4 \in \mathbf{R}^{(2n+m)\times 1}, \begin{bmatrix} \mathbf{e} & -\mathbf{e} \\ -\mathbf{e} & -\mathbf{e} \end{bmatrix} \mathbf{t}^* \leq \mathbf{0}_1, \boldsymbol{\lambda}^* \geq \mathbf{0}_3 \qquad (27)$$

where $\mathbf{0}_4$ is the $(2n+1)\times 1$ zero vector. Now, we would like to find the point $\left(\mathbf{t}^*, \boldsymbol{\lambda}^*, \mathbf{v}^*\right)$ satisfying $\mathrm{F}\left(\mathbf{t}^*, \boldsymbol{\lambda}^*, \mathbf{v}^*\right) = \mathbf{0}_4$. Here, we use a linear approximation method. From the Taylor expansions of the function $\mathrm{F}\left(\mathbf{t}, \boldsymbol{\lambda}, \mathbf{v}\right)$, we have

$$\mathrm{F}\left(\mathbf{t}+\Delta\mathbf{t},\boldsymbol{\lambda}+\Delta\boldsymbol{\lambda},\mathbf{v}+\Delta\mathbf{v}\right)\approx\mathrm{F}\left(\mathbf{t},\boldsymbol{\lambda},\mathbf{v}\right)+\nabla_{(\mathbf{t},\boldsymbol{\lambda},\mathbf{v})}\mathrm{F}\left(\mathbf{t},\boldsymbol{\lambda},\mathbf{v}\right)\begin{bmatrix}\Delta\mathbf{t}\\ \Delta\mathbf{v}\\ \Delta\boldsymbol{\lambda}\end{bmatrix} \tag{28}$$

Thus, solving the above equations yields the direction $\left(\Delta\mathbf{t},\Delta\mathbf{v},\Delta\boldsymbol{\lambda}\right)$. Next, we seek the proper step length along the direction that does not violate $\begin{bmatrix}\mathbf{e} & -\mathbf{e}\\ -\mathbf{e} & -\mathbf{e}\end{bmatrix}\mathbf{t}^* \le \mathbf{0}_1$ and $\boldsymbol{\lambda}^* \ge \mathbf{0}_3$. The pseudo code for the PDIP algorithm is shown in **Table 10**.

```
The primal-dual interior point method algorithm:
1. Initialize:
Choose $\mathbf{v}^0 \in \mathbf{R}^{m\times 1}$, $\boldsymbol{\lambda}^0 \ge \mathbf{0}_3$, and $\mathbf{t}^0 = \left[\mathbf{x}^0 \quad \mathbf{u}^0\right]^{\mathrm{T}}$, where $\mathbf{x} = \mathbf{A}^{\dagger}\mathbf{b}$, and $\mathbf{u}^0 = \left|\mathbf{x}^0\right| + \alpha\left|\mathbf{x}^0\right|$ and iteration number $k = 1$. (The $\mathbf{A}^{\dagger} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T$ is the Moore-Penrose pseudo-inverse of $\mathbf{A}$ and $\mathbf{A}^T$ denotes the transpose of $\mathbf{A}$.)
2. Find the direction vectors $\left(\Delta\mathbf{t},\Delta\mathbf{v},\Delta\boldsymbol{\lambda}\right)$:
$\begin{bmatrix}\Delta\mathbf{t}\\ \Delta\mathbf{v}\\ \Delta\boldsymbol{\lambda}\end{bmatrix} = -\left[\nabla_{(\mathbf{t}^k,\boldsymbol{\lambda}^k,\mathbf{v}^k)}\mathrm{F}\left(\mathbf{t}^k,\boldsymbol{\lambda}^k,\mathbf{v}^k\right)\right]^{-1}\mathrm{F}\left(\mathbf{t}^k,\boldsymbol{\lambda}^k,\mathbf{v}^k\right)$.
3. Find the proper step length:
Choose the largest $\alpha$ satisfying $\left\|\mathrm{F}\left(\mathbf{t}^k+\alpha,\boldsymbol{\lambda}^k+\alpha,\mathbf{v}^k+\alpha\right)\right\|_2^2 \le \left\|\mathrm{F}\left(\mathbf{t}^k,\boldsymbol{\lambda}^k,\mathbf{v}^k\right)\right\|_2^2$.
4. Update parameters:
$\mathbf{t}^{k+1} = \mathbf{t}^k + \alpha\Delta\mathbf{t},\ \mathbf{v}^{k+1} = \mathbf{v}^k + \alpha\Delta\mathbf{v},\ \boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \alpha\Delta\boldsymbol{\lambda}$.
5. Update the signal:
$\mathbf{x}^{k+1} = \mathbf{x}^k + \mathbf{t}[1:n]$.
6. Increment the iteration number $k$:
Increase iteration number $k = k+1$, and return to Step 2 if $\left\|\mathbf{y}-\mathbf{A}\mathbf{x}^k\right\|_2^2 > eps$.
```

Table 10. Primal-dual interior point method algorithm.

8.2 Orthogonal matching pursuit (OMP)

The orthogonal matching pursuit (OMP) is a famous greedy-type algorithm [17]. OMP produces a solution within k steps because it adds one index to the sparse set Λ at each iteration. The strategy of OMP is outlined in **Tables 11** and **12**.

Input	Output
An $m \times n$ measurement matrix $\mathbf{A}$ An $m-$dimensional data vector $\mathbf{y}$ The sparsity level k of the ideal signal	An estimate $\hat{\mathbf{x}}$ in $\mathbf{R}^n$ for the ideal signal. A set Λ_k containing k elements from $\{1,...,n\}$ An $m-$dimensional approximation $\hat{\mathbf{y}}_k$ of the data $\mathbf{y}$ An $m-$dimensional residual $\mathbf{r}_k = \mathbf{y} - \hat{\mathbf{y}}_k$

Table 11. Inputs and outputs of OMP algorithm.

The OMP algorithm:
1. Initialize: Let the residual vector be $\mathbf{r}_0 = \mathbf{y}$, the sparse set $\Lambda_0 = \{\}$, and iteration number $t = 1$.
2. Find the index λ_t **:** $\lambda_t = \underset{i=1,...,n}{\arg\max} \left\| \langle \mathbf{r}_{t-1}, \mathbf{a}_i \rangle \right\|$. The $\mathbf{a}_i$ is the i th column vector of matrix $\mathbf{A}$.
3. Update set: $\Lambda_t = \Lambda_{t-1} \cup \{\lambda_t\}$.
4. Signal estimate: $\mathbf{x}_t(\Lambda_t) = \mathbf{A}^{\dagger}_{\Lambda_t}\mathbf{y}$ and $\mathbf{x}_t(\Lambda_t^C) = \mathbf{0}$, where $\mathbf{x}_t(\Lambda_t)$ is the set of elements whose indices are corresponding to the sparse set.
5. Get new residual: $\hat{\mathbf{y}}_t = \mathbf{A}_t \mathbf{x}_t$, $\mathbf{r}_t = \mathbf{y} - \hat{\mathbf{y}}_t$.
6. Increment t **:** Increase iteration number $t = t+1$, and return to Step 2 if $t < k$.

Table 12. OMP algorithm.

Let us examine the above OMP algorithm. In step 2, OMP selects one index that has a dominant impact on the residual vector $\mathbf{r}$. Then, in step 3, the selected index is added to the sparse set, and the sub matrix $\mathbf{A}_{\Lambda_t}$ is constructed by collecting the column vectors of $\mathbf{A}$ corresponding to the indices of the sparse set Λ_t. OMP estimates the signal components corresponding to the indices of the sparse set and updates the residual vector by removing the estimated signal components in steps 4 and 5, respectively. Finally, OMP finishes its procedures when the cardinality of the sparse set is k.

OMP is a greedy-type algorithm because it selects the one index regarded as the optimal decision at each iteration. Thus, its performance is dominated by its ability to find the sparse set exactly. If the sparse set is not correctly reconstructed, OMP's solution could be wrong. Because OMP is very easy to understand, a couple of modified algorithms based on OMP have been designed and developed. For further information on the OMP algorithm and its modifications, interested readers are referred to two papers [8],[18].

8.3 Simultaneous orthogonal matching pursuit (SOMP)

We introduce another greedy-type algorithm based on OMP as an example: simultaneous orthogonal matching pursuit (SOMP) in [8]. This greedy algorithm has been proposed for treating multiple measurement vectors for **JSM-2** when the sparse locations of all sensed signals are the same. Namely, SOMP algorithm handles multiple measurements $\mathbf{y}_j$ as an input, when j is the index of distributed sensors, $j \in \{1,2,...,J\}$. In a later section, we use this algorithm to recover **JSM-2**. The pseudo code for SOMP is shown in **Table 13** and **14**.

Input	Output
An $m \times n$ measurement matrix $\mathbf{A}_j$ An $m-$dimensional data vector $\mathbf{y}_j$ The sparsity level k of the ideal signal	An estimate $\hat{\mathbf{x}}_j$ in $\mathbf{R}^n$ for the ideal signal. A set Λ_k containing k elements from $\{1,...,n\}$ An $m-$dimensional approximation $\hat{\mathbf{y}}_{j,k}$ of the data $\mathbf{y}_j$ An $m-$dimensional residual $\mathbf{r}_{j,k} = \mathbf{y}_j - \hat{\mathbf{y}}_{j,k}$

Table 13. Inputs and outputs of SOMP algorithm.

```
The SOMP algorithm:
1. Initialize:
Let the residual matrix be $\mathbf{r}_{j,0} = \mathbf{y}_{j,0}$. The sparse set $\Lambda_0 = \{\}$, and iteration number $t = 1$.
2. Find the index $\lambda_t$: $\lambda_t = \arg\max_{i=1,...,n} \sum_{j=1}^{J} \left| \left\langle \mathbf{r}_{j,t-1}, \mathbf{a}_{j,i} \right\rangle \right|$.
 The $\mathbf{a}_{j,i}$ is the $i$ th column vector of matrix $\mathbf{A}_j$.
3. Update set: $\Lambda_t = \Lambda_{t-1} \cup \{\lambda_t\}$.
4. Signal estimate: $\mathbf{x}_{j,t}(\Lambda_t) = \mathbf{A}_{j,\Lambda_t}^{\dagger}\mathbf{y}_j$ and $\mathbf{x}_{j,t}(\Lambda_t^C) = \mathbf{0}$, where $\mathbf{x}_{j,t}(\Lambda_t)$ is the set of elements whose indices are corresponding to the sparse set.
5. Get new residual: $\hat{\mathbf{y}}_{j,t} = \mathbf{A}_{j,t}\mathbf{x}_{j,t}$, $\mathbf{r}_{j,t} = \mathbf{y}_j - \hat{\mathbf{y}}_{j,t}$.
6. Increment $t$: Increase iteration number $t = t + 1$, and return to Step 2 if $t < k$.
```

Table 14. SOMP algorithm.

8.4 Reduce and boost (ReMBo)

ReMBo algorithm is for recovering correlated signals. The authors in [9] insisted that the algorithm improves the recovery probability of any suboptimal methods for signal matrix $\mathbf{X}$. Its idea is simple and effective. They transformed the matrix $\mathbf{X}$ to a single vector $\mathbf{x}$ and do $\mathbf{Y}$ to a single measurement vector $\mathbf{y}$. After modifying MMV equation to SMV, they apply any algorithm for SMV. We attached ReMBo algorithm from [9].

Input	Output
An $m \times n$ measurement matrix $\mathbf{A}_j$	An estimate $\hat{\mathbf{x}}_j$ in $\mathbf{R}^n$ for the ideal signal.
An $m-$dimensional data vector $\mathbf{y}_j$	Support set $\hat{S}$
The sparsity level k of the ideal signal	flag

Table 15. Inputs and outputs of ReMBo algorithm.

```
The ReMBo algorithm:
Control parameters : $k$, $\varepsilon$, Maxiters
1. Initialize:
Set iter = 1, flag = false.
2. while (iter $\le$ Maxiter) and (flag is false) do
Draw a random vector $\mathbf{a}$ of length $j$ according to randomly generated distribution.
    $\mathbf{y} = \mathbf{Aa}$
    Solve $\mathbf{y} = \mathbf{Ax}$ using SMV algorithm and save the solution $\mathbf{x}$.
    $\hat{S} = I(\mathbf{x})$
    If $\left(|\hat{S}| \le K\right)$ and $\left(\|\mathbf{y} - \mathbf{Ax}\|_2 \le \varepsilon\right)$ then
        flag = true
    else
        flag = false
    end if
    Construct $\mathbf{X}$ using $\hat{S}$ and pseudo inverse
    iter = iter + 1
end while
return $\mathbf{X}$, $\hat{S}$, flag
```

Table 16. ReMBo algorithm.

9. Reference

[1] Heung-No Lee, “Introduction to compressed sensing with coding theoretic perspective”, Spring, 2011.

[2] Jae-Gun Choi, Sang-Jun Park, Heung-No Lee, “Chapter 15: Compressive sensing and its application in wireless sensor networks”, CRC Press, December 13, 2012

[3] D. L. Donoho, “Compressed sensing,” *IEEE Trans. Inform. Theory*, vol. 52, no. 4, pp. 1289-1306, Apr. 2006

[4] D. L. Donoho and J. Tanner, “Precise undersampling theorems,” *Proc. IEEE*, vol. 98. pp. 913-924, May 2010.

[5] R. G. Baraniuk, “Lecture notes: Compressed sensing,” *IEEE Signal Process. Mag.*, pp. 118-121, July 2007.

[6] M. F. Duarte, S. Sarvotham, D. Baron, M. B. Wakin, and R. G. Baraniuk, “Distributed compressed sensing of jointly sparse signals,” *Asilomar Conf. on Signals, Systems and Computers*, pp. 1537-1541, 2005.

[7] A. Y. Yang, M. Gastpar, R. Bajcsy, and S. S. Sastry, "Distributed sensor perception via sparse representation," to appear in *Proc. IEEE.*

[8] J. A. Tropp, A. C. Gilbert, and M. J. Strauss, “Simultaneous sparse approximation via greedy pursuit,” in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 725, pp. v/72I-v/724, 2005.

[9] M. Mishali and Y. C. Eldar, “Reduce and boost: Recovering arbitrary sets of jointly sparse vectors,” *IEEE Trans. Signal Process.*, vol. 56, no. 10, pp. 4692-4702, 2008.

[10] J. Solobera, “Detecting forest fires using wireless sensor networks with Waspmote.”

[11] A. Hac, “*Wireless Sensor Network Designs*,” John Wiley & Sons, Ltd., 2003.

[12] D. L. Donoho and M. Elad, “Optimally sparse representationin general dictionaries via ell-1 minimization,” *Proc, Nat, Aca, Sci.,* vol. 100, pp. 2197-2202, 2002

[13] D. L. Donoho and M. Elad, “Maximal sparsity representation via ℓ_1 minimization,” *Proc. Natl. Acad. Sci.*, vol. 100, pp. 2197-2202, March 4, 2003.

[14] Jie Chen and Xiaoming Huo, “Theoretical results on sparse representations of multiple measurement vectors”, *IEEE Signal Process.*, pp. 4634-4643, Dec. 2006.

[15] D. Baron, M. F. Duarte, S. Sarvotham, M. B. Wakin, and R. G. Baraniuk, “An information theoretic approach to distributed compressed sensing,” in *Proc. 43rd Allerton Conf. Comm., Control, Comput.*, Sept. 2005.

[16] E. Candes and J. Romberg, Caltech, L_1 -Magic: Recovery of sparse signals via convex programming, Oct 2005.

[17] J. A. Tropp and A. C. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit,” *IEEE Trans. Inform. Theor.*, vol. 53, no. 12, pp. 4655-4666, Dec. 2007.

[18] M. E. Davies and Y. C. Eldar, “Rank awareness in joint sparse recovery,” *Arxiv preprint arXiv:1004.4529*, 2010.