IET Wireless Sensor Systems

Research Article

IET Journals
The Institution of Engineering and Technology

Simultaneous fast joint sparse recovery for WSN and IoT applications

ISSN 2043-6386
Received on 14th February 2019
Revised 21st October 2019
Accepted on 5th February 2020
E-First on 10th March 2020
doi: 10.1049/iet-wss.2019.0034
www.ietdl.org

Michael Melek[1] ✉, *Ahmed Khattab*[1], *Mohamed F. Abu-Elyazeed*[1]

[1]*Electronics and Electrical Communications Engineering Department, Cairo University, 12613 Giza, Egypt*

✉ *E-mail: michaelmelek@cu.edu.eg*

Abstract: Joint sparse recovery aims to recover a number of sparse signals having joint sparsity from multiple compressed measurements. Such a problem is finding increasing applications in wireless sensor networks (WSN) and Internet of Things (IoT) where multiple sensors collect measurements. However, existing algorithms lack both reconstruction accuracy and speed at the same time. In this study, the authors propose a joint sparse recovery algorithm, simultaneous fast matching pursuit (SFMP), which exploits some of the concepts developed for the fast matching pursuit (FMP) algorithm. SFMP achieves significant improvement in reconstruction time and speed compared to other related existing algorithms. In contrast to related algorithms, support selection is performed efficiently as the number of selected atoms is adapted from an iteration to another. Furthermore, signal estimation is performed avoiding large matrix inversion as in related algorithms. Moreover, simultaneously pruning the estimated signals, results in removing incorrectly selected ones. Due to the efficient selection strategy and the simultaneous pruning operation, the algorithm shows significant improvement in reconstruction accuracy from noisy measurements. SFMP achieves significant speed improvement over simultaneous orthogonal matching pursuit, and significant accuracy improvement over simultaneous compressive sampling matching pursuit, requiring a much smaller number of measurements.

1 Introduction

Compressed sensing (CS) is a recently developed sampling technique that is capable of acquiring a signal from samples collected at a much lower rate than the Nyquist rate [1, 2]. This technique is applicable to sparse and compressible signals. Signals can be reconstructed from their sub-Nyquist rate samples using ℓ_1 norm minimisation. However, since ℓ_1 norm minimisation is computationally expensive, various greedy recovery algorithms have been proposed for CS reconstruction, aiming to reduce reconstruction complexity without affecting reconstruction accuracy. Greedy algorithms include orthogonal matching pursuit (OMP) [3], compressive sampling matching pursuit (CoSaMP) [4], adaptive reduced-set matching pursuit (ARMP) [5] and fast matching pursuit (FMP) [6]. Such algorithms iteratively find the support of the sparse vector, and then estimate the vector based on this support.

Such traditional algorithms target the acquisition of a signal from compressed samples collected at a single node. Therefore, they are referred to as single measurement vector (SMV) setting. On the other hand, it has been shown that collecting compressed samples at multiple nodes (sensors) can significantly improve performance [7, 8]. Such a setting is referred to as multiple measurement vector (MMV) setting. Performance improvement can be noticed in an increase in the accuracy for the same number of measurements, or in a decrease in the number of required samples. In such a setting, each node obtains a compressed measurement from a sparse signal, where all sparse signals have joint sparsity. Such a framework has increasingly found applications in wireless sensor networks (WSN) and Internet of Things (IoT) [9, 10]. Some joint sparse recovery algorithms have been proposed to address such a setting. Such algorithms iteratively estimate the support of MMVs according to some sparsity model, and based on such support the sparse vectors are estimated. Such algorithms include simultaneous OMP (SOMP) [11, 12], simultaneous CoSaMP (SCoSaMP) and simultaneous hard thresholding pursuit (SHTP) [8].

However, existing joint recovery algorithms lack reconstruction accuracy and speed at the same time. This is attributed to two main factors. First, the selection strategy during support estimation is not efficient, since a fixed number of atoms (columns of the sensing matrix) is selected in each iteration. Secondly, signal estimation involves the calculation of the pseudo-inverse of the sensing matrix, which requires large matrix inversion. Accordingly, some algorithms may have high reconstruction accuracy, however, this comes at the expense of reconstruction speed, a penalty that may not be tolerable in many real-time applications. On the other hand, others are able to improve the reconstruction speed at the expense of accuracy.

In this paper, we propose a joint sparse recovery algorithm, called simultaneous FMP (SFMP), which exploits some of the concepts developed for the FMP algorithm [6]. The FMP algorithm is developed for the SMV setting and is not applicable to MMV settings since signal acquisition in SMV problems is based on measurements at a signal node. In contrast, in MMV problems multiple signals are simultaneously acquired. Our proposed algorithm is capable of joint sparse recovery of signals at significantly high accuracy and speed at the same time, compared to other related algorithms. SFMP is based on two main ideas. First, in contrast to fixed selection in other algorithms, the number of selected atoms in SFMP is variable from an iteration to the other, according to the distribution of the correlation values. This is done using a double thresholding strategy, where a reduced set of the correlation values is formed, and then values higher than a fraction of the maximum correlation value are selected. This results in improvements in both reconstruction accuracy and speed. Accuracy improvement is attributed to the optimum selection since fewer incorrect atoms are selected. Nonetheless, the support set is pruned in each iteration in order to exclude incorrectly selected atoms. Speed improvement is attributed to the significantly reduced number of iterations required.

Secondly, signal estimation in SFMP is performed in a more efficient way. While other algorithms directly obtain the pseudo-inverse of the sensing matrix, an operation involving the inversion of large matrices, SFMP obtains the pseudo-inverse in an iterative manner. In each iteration, such an inverse is obtained from data in the previous iteration. This results in a significant speed improvement.

The rest of the paper is organised as follows. Section 2 presents an overview of the related literature. Section 3 presents the system model and problem statement. We propose our SFMP algorithm in Section 4, and evaluate its performance in Section 5. We present our conclusions in Section 6.

2 Related work

The most basic joint sparse recovery algorithm is SOMP [11]. SOMP is an extension of the OMP algorithm [3]. In each iteration, the atom which contributes the most energy to as many of the input signals as possible is selected. This is done by maximising the sum of absolute correlations. However, this results in an excessively high reconstruction time, since the number of iterations required is equal to the sparsity (number of non-zero elements). Signal estimation is based on least square minimisation obtained using direct pseudo-inverse, which is computationally expensive. Furthermore, no pruning is performed for the estimated support set, so an incorrectly selected atom cannot be removed from the signal support, and will eventually degrade the performance of the algorithm.

Extensions to other CS recovery algorithms were proposed in the literature. For example, a simultaneous version of the CoSaMP algorithm, termed SCoSaMP is proposed in [8]. In each iteration, the $2k$ atoms of maximum contribution are selected, where k is the signal sparsity. Again, the selection here is fixed. Signal estimation is performed through the inefficient direct calculation of the pseudo-inverse of the sensing matrix. The estimated support set is pruned at the end of each iteration. However, the fixed selection strategy usually results in the selection of incorrect atoms, which may result in performance degradation if they are not pruned. Furthermore, the same work provides extensions to other greedy recovery algorithms: Iterative hard thresholding (IHT), normalised IHT, HTP, and normalised HTP (NHTP).

Generalised subspace pursuit (GSP) [13] is an extension for the subspace pursuit (SP) algorithm, proposed for joint sparse recovery. Instead of selecting a number of atoms that is equal to k, as in the original SP algorithm, GSP selects a number that is a function of k and of the spark of the sensing matrix which is the minimum number of linearly dependent columns. However, such selection is still fixed and suffers from the same drawback as in the previously mentioned algorithms. Furthermore, signal estimation is performed through the inefficient direct calculation of the pseudo-inverse of the sensing matrix.

A new joint sparsity model is presented in [14] and is named mixed support-set model. Such a model is a generalisation over previously existing models. In this model, the support consists of two parts. One is common for all vectors, while the other is private for each individual vector. Based on the proposed model, two algorithms are proposed, joint OMP and joint SP. However, both algorithms suffer from the same drawbacks of fixed selection and inefficient signal estimation.

Other algorithms are based on extensions of the MUltiple SIgnal Classification (MUSIC) algorithm [15]. Such algorithms interpret the joint sparse recovery problem as a binary classification problem with respect to atoms. Among such algorithms is subspace-augmented MUSIC, which improves on MUSIC so that the support is reliably recovered under unfavourable conditions such as rank-defect and ill-conditioning [16]. Another extension to the MUSIC algorithm is semi-supervised MUSIC [17], in which labelled MMVs and some reliable unlabelled atoms are iteratively exploited for classifier construction. Other algorithms are based on 2D signal CS recovery [18, 19].

The aforementioned algorithms are based on a centralised setting, in which multiple nodes acquire different measurements and forward them to a fusion centre (FC), which in turn, performs joint recovery. Other algorithms are based on decentralised setting, in which no FC is required. In such a setting, a node exchanges information with its neighbours and arrives at a local estimation of the sparse signal. However, since a single node is limited in memory and processing capability, neighbouring nodes cooperate with each other to compensate for such limitations and achieve satisfactory performance [20]. Nonetheless, cooperation results in increased communication among nodes, which can be expensive. A distributed SP algorithm as a decentralised version of the JSP algorithm is proposed in [21]. Distributed iterative thresholding algorithms addressing the issues of the low complexity and memory limitations of nodes are proposed in [22]. A distributed parallel pursuit algorithm is presented in [23], which is based on the exchange of information about estimated support sets between nodes. Distributed compressive sensing methods that exploit time correlation of measurements as well as space correlation in IoT applications are proposed in [10].

Various works have proposed the acquisition of compressed data in real-world WSN and IoT applications. For example, in [24] audio sensors (microphones) collect compressed samples in order to reduce the transmitted data over the wireless channel to the FC. In [25, 26], compressed samples of temperature, humidity, light and wind speed sensors are collected and transmitted in order to save energy and prolong the network lifetime. In [27], electrocardiogram and electroencephalogram signals are acquired in a wireless body area network in order to reduce the number of measurements and minimise the energy consumption of the sensor nodes. In [28, 29], greenhouse monitoring systems based on compressive measurements of temperature and humidity are proposed. Furthermore, such data can have a sparse representation in a certain basis. For example, audio signals are sparse in DCT and DWT domains [30].

3 System model and problem statement

While our algorithm is applicable to any sparsity model, we follow the common support joint sparse model JSM-2 presented in [31], which is widely considered for sensor networks [32] and the references therein. Such model can represent practical scenarios in which multiple sensors acquire the same signal but with different phase shifts and attenuations caused by signal propagation. Applications of JSM-2 include IoT, WSN [25], MIMO radar [33], source localisation, neuromagnetic imaging, and equalisation of sparse communication channels [34]. In this model, all sparse signals acquired at different nodes share a common support, i.e. same indices for non-zero elements. However, the corresponding values can be different in general. Therefore, the sparse vectors of length n are expressed as

$$x_j = \mathbf{\Psi}\theta_j, \quad j \in \{1, 2, \ldots, J\} \tag{1}$$

where all θ_j have common support $\Omega \subset \{1, 2, \ldots, n\}$, $|\Omega| = k$, and $\mathbf{\Psi}$ is an $n \times n$ dictionary matrix. Hence, all signals are k-sparse and are constructed from the same k elements of the dictionary $\mathbf{\Psi}$ but with different coefficients.

The set consists of a number of L nodes or sensors, each acquiring a compressed measurement of the corresponding sparse vector given by

$$y_l = \mathbf{\Phi}x_l, \quad l \in \{1, 2, \ldots, L\} \tag{2}$$

where y_l is the compressed measurement at the node l and $\mathbf{\Phi}$ is the sensing matrix.

WSN nodes typically have limited computational power that is not suitable for the relatively high computational complexity of CS recovery algorithms. Therefore, we consider a centralised setting in which measurements are forwarded by the nodes to a FC, which performs the joint recovery, i.e. estimate the sparse signals $\{x_l\}_{l=1}^{L}$ given the compressed measurements $\{y_l\}_{l=1}^{L}$. This has the benefit of the reduction in the required complexity of the nodes. Moreover, there is no need for communication among nodes [20].

3.1 Application in real-life WSN and IoT systems

Even though CS might seem computationally complex for application in the resource-limited WSN and IoT systems, CS is counter-intuitively suitable for such networks [35]. This is attributed to two reasons. First, CS sampling captures only few

measurements of the signal that are linearly independent. These few samples are enough to reconstruct the signal as long as it is sparse in a certain domain. This makes CS compression agnostic of the other properties of the signal, and hence, applicable to various applications as long as the measured attribute is sparse in one or more domains. Secondly, CS reduces the energy spent in communicating the measurements to where the signal is accurately reconstructed as only a small number of measurements need to be transmitted. The signal reconstruction is implemented on a gateway or a FC which typically has high computational power and does not have the same energy limitations of the sensor nodes.

A detailed analysis of the potential of applying CS on the resource-limited WSN and IoT platforms for low-complexity and energy-efficient data acquisition is presented in [25]. We assume the system hardware model used in [25] since both our greedy reconstruction algorithm and the greedy reconstruction algorithm presented in [25] consider similar problem formulations that are based on the common support joint sparse model JSM-2. More specifically, data from seven different weather stations deployed near Monterey, CA were collected in [25]. The seven wireless sensor nodes are deployed in an IEEE 802.15.4 star network topology in which the sensor nodes send their measurements to a FC that reconstructs the signal using the few collected samples. Each weather station is equipped with three different kinds of sensors: temperature, relative humidity (highly correlated signals) and wind speed (less correlated signal). The wireless sensor node hardware and the FC are built using the ST microelectronics (STM32W108) fully-integrated system-on-chip solution. The STM32W108 platform has a 32-bit 24 MHz ARM Cortex-M3 microprocessor with a built-in 2.4 GHz IEEE 802.15.4-compliant transceiver, 128 KB Flash memory and 8 KB of RAM memory. The Sensirion SHT21 temperature and relative humidity sensor and the wind sensor are interfaced to the STM32W108 board to build a sensor node. This hardware setup has been used to evaluate several greedy algorithms in [25] in real-life and can be used to evaluate our algorithm. However, we defer the hardware implementation and real-life evaluation to a sequel paper to keep the focus on the proposed reconstruction approach.

4 Simultaneous fast matching pursuit

In this section, we present our proposed algorithm, SFMP. SFMP exploits some of the concepts developed for FMP [6]. The FMP algorithm is developed for the SMV setting, and hence, is not applicable to MMV problems, since signal acquisition in SMV problems is based on measurements at a signal node. The main goal of SFMP is to perform joint sparse recovery (simultaneous CS reconstruction) as accurately and efficiently as possible. In other words, SFMP addresses the MMV problem, in which MMVs are available through CS at multiple nodes. Each measurement vector corresponds to a different unknown sparse vector, where all sparse vectors share the same support.

SFMP is a greedy iterative algorithm which consists of the following components: (1) simultaneous selection; (2) signal estimation; (3) simultaneous pruning; and (4) total residual calculation. First, we give a brief description of its components.

1. *Simultaneous selection:* In contrast to related algorithms, SFMP targets the selection of an optimum number of atoms per iteration. The atoms with the highest correlation to all measurement vectors are selected. We adopt a selection strategy similar to the one developed in [5, 36] to adaptively select elements from a reduced set of the correlation values, adding them to the identified support set.
2. *Signal estimation:* Based on the identified support set, least-square minimisation is performed to estimate the sparse signal corresponding to each measurement vector. However, in contrast to related algorithms, this step is done iteratively avoiding large matrix inversion, which significantly improves the speed of the algorithm.
3. *Simultaneous pruning:* The identified support set is simultaneously pruned, removing the elements with the least contribution to all measurement vectors. This results in a cleaner support set since incorrectly selected elements are removed in each iteration.
4. *Total residual calculation:* The residual is calculated and summed over all reconstructed signals.

4.1 SFMP algorithm

In this section, we explain in detail the components of the SFMP algorithm. The main idea of a greedy recovery algorithm is to identify the support of the sparse signal(s), and then estimate the signal(s) with non-zero elements based on such support, while minimising the reconstruction error in the least square sense. Such steps are performed iteratively.

Given a measurement matrix $\boldsymbol{Y} = [y_1, y_2, \ldots, y_L]$, it is required to estimate the sparse vectors' matrix $\boldsymbol{X} = [x_1, x_2, \ldots, x_L]$. Initially, an empty identified support set is created. This set will be updated in each iteration with the estimated common support of the sparse signals. The residual matrix $\boldsymbol{R}$ is initially set to the measurement matrix $\boldsymbol{Y}$. The following operations are performed in each iteration.

4.1.1 Simultaneous support identification: In SMV problems, in which signals are to be estimated from a single measurement vector, the measurement vector is correlated with all columns of the sensing matrix. Indices of columns corresponding to the top correlation magnitudes are expected to belong to the support of the original sparse signal. According to some selection strategy, a number of the top magnitude correlation values are selected, and their indices are added to the identified support set. The first source of improvement in recovery algorithms targeting MMV problems, compared to those targeting SMV problems, is performing correlation over all measurement vectors, finding the columns that are most correlated to all measurement vectors. This results in finding the columns with the most contribution to all vectors. This is done by multiplication of the conjugate transpose of the sensing matrix by the residual matrix, which is initially set to the measurement matrix

$$\boldsymbol{G} = \boldsymbol{\Phi}^* \boldsymbol{R} \tag{3}$$

The norm of each row of $\boldsymbol{G}$ is a measure of the correlation of the corresponding column of $\boldsymbol{\Phi}$ to all measurement vectors, and accordingly, a measure of its contribution. Consequently, a number of indices corresponding to the top magnitude norms is selected and added to the identified common support set. Our selection strategy is based on the one previously develop in [5, 36]. Selection is performed in two steps. First, the elements of the correlation norm vector are reduced to a set containing the βk top magnitude elements, where β is a parameter and k is the common sparsity of the signals. Then, we select from the reduced set elements of magnitude larger than a fraction α of the maximum value of the norm vector, where α is another parameter.

The proper selection of α and β leads to the selection of an optimum number of elements per iteration. It was shown in [36] that for such a selection strategy, moderate values of both $\alpha \in [0.5, 0.7]$ and $\beta \in [0.15, 0.75]$ result in the best performance. Furthermore, the performance of such a selection approach is not sensitive to the specific values of α and β, as long as they are selected in the aforementioned moderate range.

In this way, we avoid common selection problems in related algorithms. Namely, we avoid the selection of too few elements of the norm vector, which leads to an excessively large number of iterations, which in turn significantly increases the reconstruction time. We also avoid the selection of too many elements, which results in selecting more incorrect elements not belonging to the support set, which in turn reduces the reconstruction accuracy.

4.1.2 Signal estimation: Based on the identified common support set, sparse signals are estimated for each measurement vector separately. This is done by minimising the least square error between the estimated sparse signal $\hat{x}_l$ and the corresponding measurement vector $\boldsymbol{y}_l$, given by $\| \boldsymbol{y}_l - \boldsymbol{\Phi}\hat{x}_l \|_2^2$.

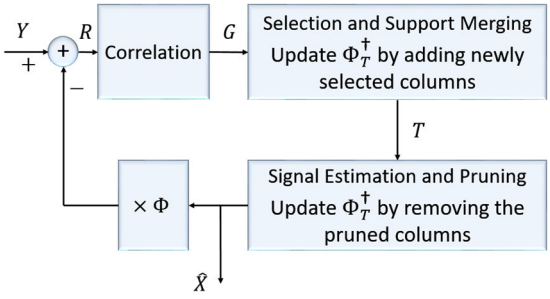

Fig. 1 *Block diagram of the algorithm*

Input: Sensing matrix Φ, measurement matrix Y, sparsity k, number of nodes L, parameters α and β.
Initialize: $\hat{X}^{[0]} = 0$, $R^{[0]} = Y$, $T^{[0]} = \emptyset$.
for $i = 1; i := i + 1$ until the stopping criterion is met **do**
  $G^{[i]} \leftarrow \Phi^* R^{[i-1]}$ {Form correlation matrix}
  $W \leftarrow \texttt{SelectSupport}(G^{[i]}, \alpha, \beta)$ {simultaneous support selection}
  $U \leftarrow W \setminus T$ {Indices of newly added atoms not previously identified}
  $T \leftarrow \begin{pmatrix} T & U \end{pmatrix}$ {Support merging}
  for $l = 1; \quad l \leq L; \quad l = l + 1$ **do** {loop all users estimating signals based on support T}
    if i==1 **then**
      $F \leftarrow (\Phi_U^T \Phi_U)^{-1}$ {Gram matrix used for pseudo-inverse}
      $\Phi^\dagger \leftarrow F\Phi_U^T$
    else {After first iteration, Gram matrix is obtained iteratively }
      $F \leftarrow \texttt{GramInverse}(\Phi, U) \quad \{F = (\Phi_T^T \Phi_T)^{-1}, \quad$ update inverse after adding U to $T\}$
      $\Phi_T^\dagger \leftarrow F\Phi_T^T$
    end if
    $b|_T \leftarrow \Phi_T^\dagger y, b|_{T^c} \leftarrow 0$ {Signal estimation}
    $\hat{X}_l = \hat{x}^{[i]}\{l^{th}$ column of the estimated signal matrix$\}$
  end for
    $T_{pruned} \leftarrow \texttt{RowwiseNorm}(\hat{X})$ {find indices with least contribution}
    $V \leftarrow T \setminus L_k(T_{pruned})$ {Pruned indices}
    $\hat{X}^{[i]} \leftarrow H_k(\hat{X}^{[i]})$ {Simultaneous pruning}
  if $|V| > 0$ **then**
    $T \leftarrow T \setminus V$ {Update support set by removing pruned indices}
    $F \leftarrow \texttt{PruneGramInverse}(F, V)$ {Update $F = (\Phi_T^T \Phi_T)^{-1}$ removing the pruned columns of Φ at indices V}
  end if
    $R \leftarrow Y - \Phi\hat{X}^{[i]}$ {Update residuals}
end for
Output: Sparse matrix $\hat{X}$

Fig. 2 *Algorithm 1 Simultaneous fast matching pursuit*

Most related algorithms perform such minimisation using multiplication with the pseudo-inverse of the sensing matrix, given by $\boldsymbol{\Phi}_T^\dagger = (\boldsymbol{\Phi}_T^{\mathrm{T}}\boldsymbol{\Phi}_T)^{-1}\boldsymbol{\Phi}_T^{\mathrm{T}}$, where Φ_T is a matrix containing columns of the sensing matrix $\boldsymbol{\Phi}$ at indices from the support set T, which is updated in each iteration. This requires inversion of large matrices, which increases the complexity of the algorithm. In contrast, SFMP obtains the pseudo-inverse iteratively, where it is calculated in each iteration using data from the previous iteration. This is done by updating the inverse of the Gram matrix $(\boldsymbol{\Phi}_T^{\mathrm{T}}\boldsymbol{\Phi}_T)^{-1}$ in each iteration by adding to $\boldsymbol{\Phi}_T$ the newly selected columns in that iteration.

In order to perform such update without recalculating the inverse in each iteration, we make use of the Schur–Banachiewicz inverse formula [37]. Let $\boldsymbol{Q}$ be a matrix containing the newly added columns corresponding to the newly selected indices. And let $\boldsymbol{\Phi}_i$ denote the matrix containing columns of the sensing matrix at indices T in the ith iteration, where we drop the subscript T for notation clarity. Thus, the updated sensing matrix in iteration i is given by $\boldsymbol{\Phi}_i = \begin{pmatrix} \boldsymbol{\Phi}_{i-1} & \boldsymbol{Q} \end{pmatrix}$. Then, we get the inverse $(\boldsymbol{\Phi}_i^{\mathrm{T}}\boldsymbol{\Phi}_i)^{-1}$ using data from the previous iteration applying the Schur–Banachiewicz inverse formula as follows:

$$(\boldsymbol{\Phi}_i^{\mathrm{T}}\boldsymbol{\Phi}_i)^{-1} = \begin{pmatrix} B^{-1} + B^{-1}CS_c^{-1}DB^{-1} & -B^{-1}CS_c^{-1} \\ -S_c^{-1}DB^{-1} & S_c^{-1} \end{pmatrix} \tag{4}$$

where $B = \boldsymbol{\Phi}_{i-1}^{\mathrm{T}}\boldsymbol{\Phi}_{i-1}$, $C = \boldsymbol{\Phi}_{i-1}^{\mathrm{T}}\boldsymbol{Q}$, $D = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{\Phi}_{i-1}$, $E = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}$, and $S_c = E - DB^{-1}C$. Here $B^{-1} = (\boldsymbol{\Phi}_{i-1}^{\mathrm{T}}\boldsymbol{\Phi}_{i-1})^{-1}$ is available from the last iteration and is only calculated directly in the first iteration. The complete details are presented in [6].

4.1.3 Simultaneous pruning: Through successive iterations, some erroneous indices (not belonging to the support of the sparse signals) are inevitably selected. The ability of SFMP to identify and exclude such elements contributes to its high reconstruction accuracy, as well as to its noise robustness. In an SMV problem, the idea of pruning is to exclude the indices of the atoms with the least contribution to the estimated signal. However, in an MMV problem, in order to make use of the multiple measurements that we collect, we exclude atoms with the least contribution to all the estimated signals. The ℓ_1 norms of the rows of the matrix $\hat{\boldsymbol{X}}$ are calculated. Indices corresponding to the top magnitude k rows are retained, while the rest are removed from the signals support, setting their contribution to zero.

Since the matrix $(\boldsymbol{\Phi}_T^{\mathrm{T}}\boldsymbol{\Phi}_T)^{-1}$ is required for the following iterations, it is updated as well, but this time removing pruned columns from $\boldsymbol{\Phi}_T$. Again, this is done iteratively without recalculating the new inverse. We apply the Schur–Banachiewicz formula but in a reverse manner compared to the signal estimation stage. Given the inverse $(\boldsymbol{\Phi}_T^{\mathrm{T}}\boldsymbol{\Phi}_T)^{-1}$, we seek to update it by removing from $\boldsymbol{\Phi}_T$ the pruned columns.

4.1.4 Total residual calculation: In an SMV problem, the residual is calculated by subtracting the contribution of the pruned estimated signal from the measurement vector. Here, in an MMV problem, the contribution of all pruned estimated vectors (columns of $\hat{\boldsymbol{X}}$) is subtracted from the corresponding measurement vectors (matrix $\boldsymbol{Y}$) obtaining the residual matrix $\boldsymbol{R}$.

The residual matrix is obtained as follows:

$$\boldsymbol{R} = \boldsymbol{Y} - \boldsymbol{\Phi}\hat{\boldsymbol{X}} \tag{5}$$

Then the total residual norm is obtained by summing the norms of the residuals for all the vectors. The ℓ_2 norm of each column of $\boldsymbol{R}$ is obtained and then they are summed.

In successive iterations, the residual matrix is used in the simultaneous support identification stage as previously explained. The algorithm is repeated until any of the following conditions is satisfied:

(i) the norm of the total residual is less than ϵ_1;
(ii) the difference between the total residuals in two successive iterations is less than ϵ_2; or
(iii) the total residual in an iteration is larger than that in the previous iteration.

We summarise the proposed algorithm in Algorithm 1. The operator $L_k(\cdot)$ returns the index set of the k largest absolute values of the elements of its argument vector. The hard thresholding operator $H_k(\cdot)$ retains only the k elements with the largest absolute values and sets the rest to zero. Algorithms 2–4 list functions used in SFMP for simultaneous support selection, Gram matrix inversion, and simultaneous pruning, respectively. Fig. 1 illustrates a block diagram of the algorithm (See Figs. 2–5).

4.2 Complexity analysis

We perform the complexity analysis for the SFMP algorithm. First, we analyse the complexity of each stage of the algorithm, then we find the overall complexity.

4.2.1 Simultaneous support identification: The complexity of this stage is dominated by the matrix–matrix multiplication $\mathbf{\Phi}^{*}\boldsymbol{R}^{[i-1]}$, which is $\mathcal{O}(Lmn)$. Sorting is performed using quicksort or mergesort in $\mathcal{O}(n\log_2 n)$. Support merging can be performed in $\mathcal{O}(k)$. The overall complexity of this step is $\mathcal{O}(Lmn)$.

4.2.2 Signal estimation and pruning: SFMP avoids large matrix inversion which is used in related algorithms. We denote the average number of elements selected per iteration by p. This value is much smaller than the sparsity k of the signal. The complexity of this step is dominated by the iterative calculation of the inverse of B. Since B is symmetric, B^{-1} is symmetric as well. We also have $D = C^{\mathrm{T}}$. Thus, we first compute DB^{-1} and transpose it to obtain $B^{-1}C$. The rest of matrices products used for signal estimation and pruning are easily calculated using the aforementioned matrices in $\mathcal{O}(kmp)$. Inverting small matrices of size p is $\mathcal{O}(p^3)$. This is a very small cost paid for avoiding the inversion of much larger matrices. The pruning step keeps the top magnitude k elements, which is done by sorting the vector in $\mathcal{O}(k\log_2 k)$. Therefore, the overall complexity of this stage is given by $\mathcal{O}(kmp + p^3)$. Empirically, we take $p = \sqrt{k}$. Thus, the complexity of this stage is $\mathcal{O}(k^{1.5}m)$ per node. The total complexity is $\mathcal{O}(Lk^{1.5}m)$.

4.2.3 Residual calculation: The complexity of this step is dominated by the matrix–matrix multiplication given by $\mathcal{O}(Lmk)$.

4.2.4 Overall complexity: The overall complexity per iteration is $\mathcal{O}(Lmn + Lk^{1.5}m)$. It should be noted here that k is much smaller than n. Furthermore, SFMP requires a much smaller number of iterations compared to SOMP as illustrated by simulation results.

5 Simulation results

5.1 Simulation setup

In this section, we compare the performance of our proposed algorithm, SFMP, against other main greedy matching pursuit algorithms, SOMP [7, 11] and SCoSaMP [8]. Such algorithms are the counterparts of two of the most well-known and cited SMV algorithms [3, 4].

For each algorithm, the reported results are the average of the metrics evaluated for ten independent trials. In each trial, we generate a random sparse signal of length $n = 1000$. For SFMP, we take $\alpha = 0.7$ and $\beta = 0.25$ [36].

5.2 Performance metrics

The performance metrics that we use to compare our proposed SFMP algorithm against other related algorithms are as follows:

- The reconstruction time t in seconds, which is the average time required to reconstruct the sparse vectors from the measurement vectors.
- The total reconstruction error which is the average reconstruction error relative to the ℓ_2 norm of the vector defined as $\sum_{l=1}^{L} \| x_l - \hat{x}_l \|_2 / \| x_l \|_2$.

5.3 Simulation results

We perform three sets of experiments to evaluate the performance of the algorithms. In the first two experiments, we use noiseless measurements. First, we vary the number of measurements at each sensor, fixing the number of sensors used. This illustrates the reconstruction capability of the algorithm, since a good recovery algorithm is capable of achieving high performance using a smaller number of measurements. Then, we vary the number of sensors fixing the number of measurements. This shows the improvement achieved by using multiple measurements. Finally, in our third experiment, we examine the effect of noisy measurements, varying the signal-to-noise ratio (SNR), while fixing the number of measurements and sensors.

Input: Correlation matrix G, sparsity k, parameters α and β.
$g \leftarrow \texttt{RowwiseNorm}(G)$ {g is a vector of row-wise ℓ_2 norm of G}
$J \leftarrow L_{\beta k}(g)$ {Indices of βk largest magnitude elements in g}
$W \leftarrow \{j : g_j \geq \alpha \max_l |g_l|, j \in J\}$ {indices of elements in J larger than or equal to $\alpha \max_l |g_l|$}
Output: Selected support W

Fig. 3 *Algorithm 2 SelectSupport function*

Input: Sensing matrix Φ, newly added support U, merged support set T
$Q \leftarrow \Phi_U$
$B^{-1} \leftarrow F, C \leftarrow \Phi_T^T Q$, $D \leftarrow Q^T \Phi_T$, $E \leftarrow Q^T Q$
$S_c \leftarrow E - DB^{-1}C$
$F \leftarrow \begin{pmatrix} B^{-1} + B^{-1}CS_c^{-1}DB^{-1} & -B^{-1}CS_c^{-1} \\ -S_c^{-1}DB^{-1} & S_c^{-1} \end{pmatrix}$
$\{F = (\Phi_T^T \Phi_T)^{-1}\}$
Output: Gram matrix inverse F

Fig. 4 *Algorithm 3 GramInverse function*

Input: Gram matrix before pruning F, sensing matrix Φ, indices to be pruned$V$, pruned support set T
$t \leftarrow |T|, v \leftarrow |V|$ {Cardinality of sets T and V }
$\begin{pmatrix} B_{t-v\times t-v} & C_{t-v\times v} \\ D_{v\times t-v} & E_{v\times v} \end{pmatrix} \leftarrow F$ {Partition F into four matrices}
$S_c \leftarrow E - DB^{-1}C$
$N_{11} \leftarrow B^{-1} + B^{-1}CS_c^{-1}DB^{-1}$, $N_{12} \leftarrow -B^{-1}CS_c^{-1}$,
$N_{21} \leftarrow -S_c^{-1}DB^{-1}, N_{22} \leftarrow S_c^{-1}$
$F \leftarrow N_{11} - N_{12}N_{22}^{-1}N_{21}$ {Update $F = (\Phi_T^T \Phi_T)^{-1}$ removing the pruned columns}
Output: Inverse of Gram matrix after removing pruned columns F

Fig. 5 *Algorithm 4 PruneGramInverse function*

5.3.1 Impact of the number of measurements: In our first experiment, we examine the reconstruction capability of the simulation algorithms in terms of accuracy and speed. We compare the performance of the algorithms against the number of measurements taken at each sensor. Measurements are collected by $L = 20$ sensors, each taking compressed measurements of the corresponding sparse vector, and forwarding them to an FC. The sparse vectors have common support, according to the JSM-2 model, and their common sparsity is $s = 200$, i.e. each having 200 non-zero elements. The number of measurements m taken at each node is varied from 100 to 500 at steps of 50.

Figs. 6 and 7 show the reconstruction error and time, respectively, against the number of measurements for the simulated algorithms. Tables 1 and 2 list the corresponding values. SFMP achieves less reconstruction error compared to SOMP and SCoSaMP. At a number of measurements $m = 200$, both SFMP and SOMP achieve perfect reconstruction. However, our SFMP achieves remarkably less reconstruction error compared to SOMP when the measurements are <200. SCoSaMP, on the other hand, requires a much larger number of measurements $m = 450$ to achieve perfect reconstruction. Both optimum simultaneous selection and simultaneous pruning contribute to the superior accuracy of SFMP.

Examining the reconstruction time of the algorithms, we notice the significant speed improvement that SFMP achieves compared to SOMP. While both algorithms achieve perfect reconstruction at $m = 200$, SFMP takes only 0.03 s, while SOMP takes a much larger time of 1.1 s. This is due to the selection strategy of SFMP, which aims to select an optimum number of elements per iteration. On the other hand, SOMP selects only one element per iteration, which results in a much larger number of iterations. Furthermore, avoiding large matrix inversion contributes to the improved speed of SFMP. It is noted that SCoSaMP achieves close reconstruction time to SFMP, however, its reconstruction accuracy is poor, and it requires a much larger number of measurements to achieve the same accuracy as SFMP. This shows that the speed improvement of SFMP does not come at the expense of its accuracy.

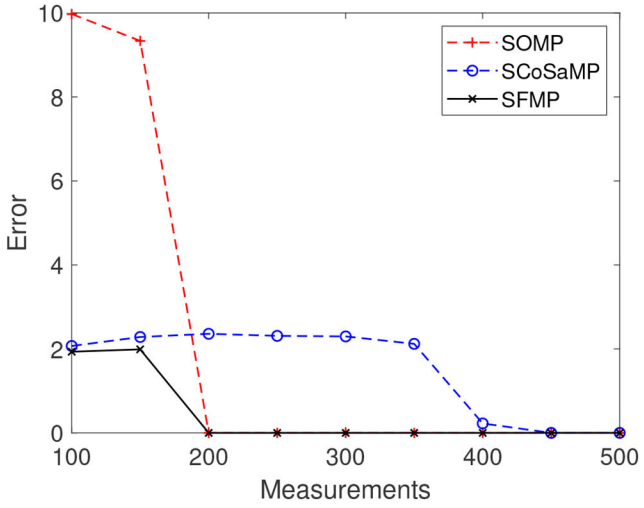

Fig. 6 *Reconstruction error versus number of measurements*

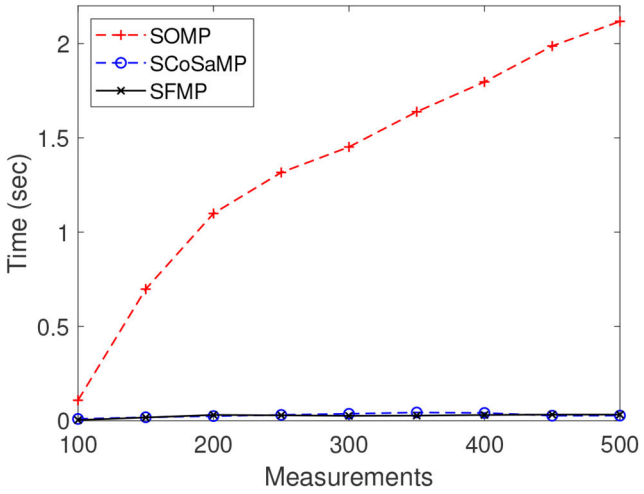

Fig. 7 *Reconstruction time versus number of measurements*

Table 1 Reconstruction error

Measurements	100	150	200	250	300	350	400	450	500
SOMP	9.98	9.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SCoSaMP	2.07	2.28	2.36	2.31	2.30	2.12	0.23	0.00	0.00
SFMP	1.93	1.99	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 2 Reconstruction time in seconds

Measurements	100	150	200	250	300	350	400	450	500
SOMP	0.11	0.70	1.10	1.32	1.45	1.64	1.80	1.99	2.12
SCoSaMP	0.01	0.02	0.02	0.03	0.04	0.04	0.04	0.03	0.03
SFMP	0.00	0.02	0.03	0.03	0.03	0.03	0.03	0.03	0.03

The number of iterations required by the algorithms is shown in Fig. 8. Since SOMP selects a single element per iteration, it requires a much larger number of iterations compared to SCoSaMP and SFMP. This accounts for the significant increase in the reconstruction time of SOMP. On average, the number of iterations required by SOMP is 168, compared to 2 for SCoSaMP and 6 for SFMP. SFMP performs slightly more iterations than SCoSaMP. However, this is due to the adaptive selection strategy of SFMP, which results in fewer erroneously selected indices and better reconstruction accuracy.

5.3.2 Impact of the number of sensors: In the next experiment, we illustrate the benefit from using multiple sensors to reconstruct the signals as opposed to using a single sensor. Figs. 9 and 10 illustrate the reconstruction time and error, respectively, against the number of sensors. The signals have a common sparsity $s = 200$ and the number of measurements $m = 200$. The number of sensors is varied from 1 to 20. Using a single sensor, none of the simulated algorithms are able to reconstruct the sparse signal using the number of measurements $m = 200$. However, as the number of sensors reaches 15, the signals are perfectly reconstructed by SFMP and SOMP with zero error. SFMP requires a significantly smaller reconstruction time of 0.03 s, while SOMP takes 1.12 s. On the other hand, SCoSaMP is not able to reconstruct the sparse signals using $m = 200$ measurements even as the number of sensors is increased beyond 20. This shows that the reconstruction capability of SFMP is superior to that of SCoSaMP. This is mainly attributed to the selection strategy of SFMP which selects an optimum number of elements per iteration. SCoSaMP, on the other hand, selects a fixed number of elements per iteration. Such fixed

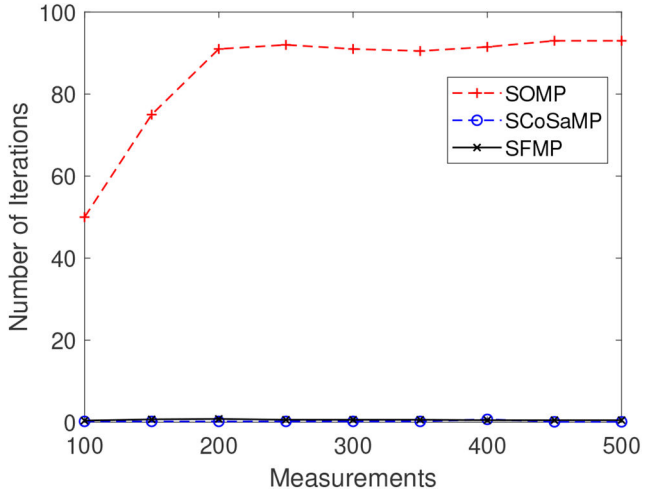

Fig. 8 *Iterations versus number of measurements*

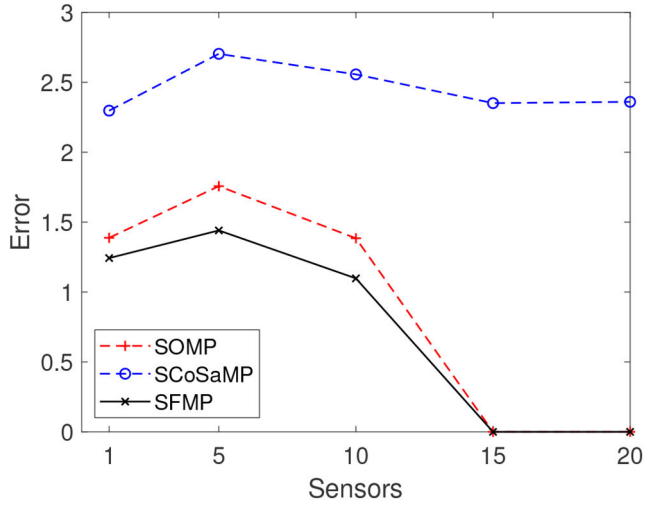

Fig. 9 *Reconstruction error versus number of sensors*

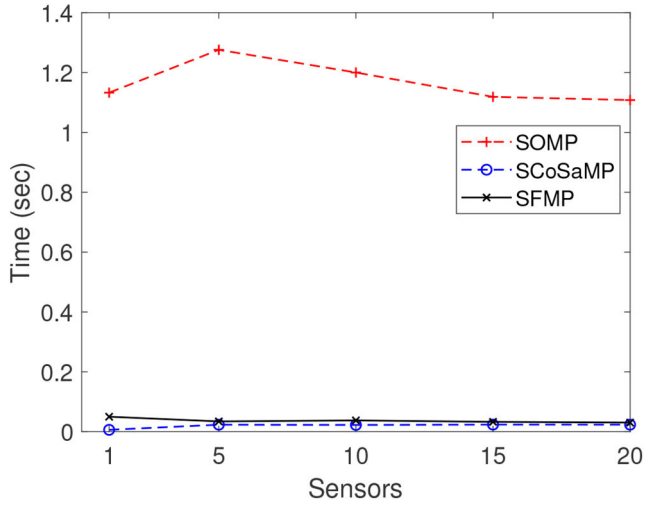

Fig. 10 *Reconstruction time versus number of sensors*

selection results in selection of too many elements per iteration, which results in selecting too many incorrect elements.

5.3.3 Impact of SNR: Next, we turn to the case in which the measurements are contaminated with additive white Gaussian noise. In this experiment, we plot the reconstruction error for the simulated algorithms against the SNR. We take a number of measurements $m = 200$ and a number of sensors $L = 20$. The sparsity of all signals $s = 200$.

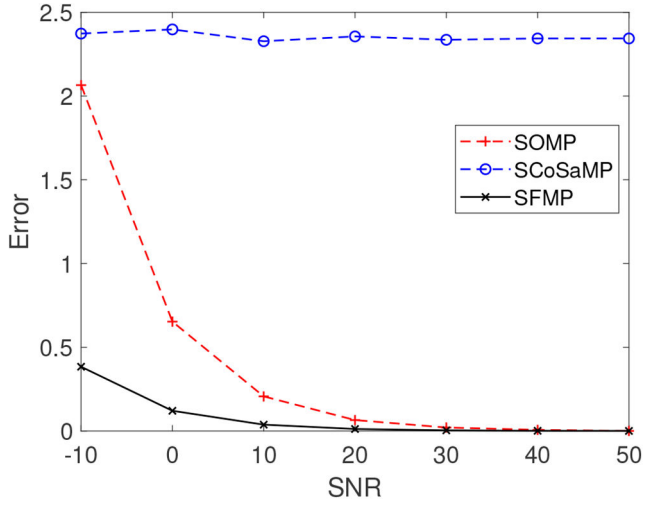

Fig. 11 *Reconstruction error versus SNR*

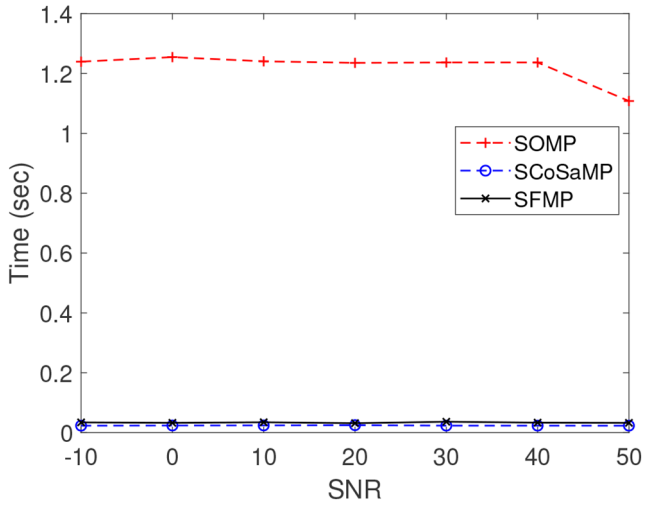

Fig. 12 *Reconstruction time versus SNR*

Figs. 11 and 12 depict the reconstruction error and time, respectively, against SNR. The proposed algorithm SFMP achieves the least reconstruction error among the simulated algorithms. At an SNR of 10 dB, SFMP gives an error of 0.038, while SOMP gives 0.2. Again for SCoSaMP, the number of measurements $m = 200$ is not enough to achieve accurate reconstruction and the error is 2.3. SFMP achieves the least reconstruction time of 0.03 s, which is significantly faster than SOMP which takes 1.2 s.

The superior noise robustness of SFMP is attributed to its optimum simultaneous selection (which prevents too many incorrect elements from being selected), and to simultaneous pruning (which excludes the few incorrectly selected elements).

6 Conclusion

In this paper, we have proposed SFMP, which is a fast and efficient joint sparse recovery algorithm. SFMP achieves significant improvement in reconstruction accuracy due to selecting an optimum number of elements per iteration, and due to the simultaneous pruning of the support set in each iteration. Furthermore, SFMP achieves significant speed improvement due to avoiding larger matrix inversion in the signal estimation, and due to its optimum selection strategy, which results in fewer iterations. Compared to related algorithms, SFMP achieves significant speed improvement over SOMP (0.03 as opposed to 1.1 s). SFMP also achieves significant accuracy improvement over SCoSaMP, requiring a much smaller number of measurements (200 as opposed to 450 measurements) to achieve the same reconstruction accuracy.

7 References

[1] Candès, E.J.: 'Compressive sampling'. Proc. of the Int. Congress of Mathematicians, Madrid, Spain, 2006, vol. 3, pp. 1433–1452

[2] Lustig, M., Donoho, D.L., Santos, J.M., *et al.*: 'Compressed sensing MRI', *IEEE Signal Process. Mag.*, 2008, **25**, (2), pp. 72–82

[3] Tropp, J., Gilbert, A.C.: 'Signal recovery from partial information via orthogonal matching pursuit', *IEEE Trans. Inf. Theory*, 2007, **53**, (12), pp. 4655–4666

[4] Needell, D., Tropp, J.A.: 'CoSaMP: iterative signal recovery from incomplete and inaccurate samples', *Commun. ACM*, 2010, **53**, (12), pp. 93–100

[5] Abdel-Sayed, M.M., Khattab, A., Abu-Elyazeed, M.F.: 'Adaptive reduced-set matching pursuit for compressed sensing recovery'. IEEE Int. Conf. on Image Processing (ICIP), Phoenix, Arizona, 2016

[6] Abdel-Sayed, M.M., Khattab, A., Abu-Elyazeed, M.F.: 'Fast matching pursuit for wideband spectrum sensing in cognitive radio networks', *Wirel. Netw.*, 2019, **25**, pp. 131–143

[7] Tropp, J.A., Gilbert, A.C., Strauss, M.J.: 'Algorithms for simultaneous sparse approximation. Part I: greedy pursuit', *Signal Process.*, 2006, **86**, (3), pp. 572–588

[8] Blanchard, J.D., Cermak, M., Hanle, D., *et al.*: 'Greedy algorithms for joint sparse recovery', *IEEE Trans. Signal Process.*, 2014, **62**, (7), pp. 1694–1704

[9] Gubbi, J., Buyya, R., Marusic, S., *et al.*: 'Internet of things (IoT): a vision, architectural elements, and future directions', *Future Gener. Comput. Syst.*, 2013, **29**, (7), pp. 1645–1660

[10] Xiao, S., Li, T., Yan, Y., *et al.*: 'Compressed sensing in wireless sensor networks under complex conditions of Internet of things', *Cluster Comput.*, 2019, **22**, pp. 14145–14155

[11] Tropp, J.A., Gilbert, A.C., Strauss, M.J.: 'Simultaneous sparse approximation via greedy pursuit'. Proc. (ICASSP'05). IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, 2005, Philadelphia, PA, 2005, vol. 5

[12] Determe, J.-F., Louveaux, J., Jacques, L., *et al.*: 'On the exact recovery condition of simultaneous orthogonal matching pursuit', *IEEE Signal Process. Lett.*, 2016, **23**, (1), pp. 164–168

[13] Feng, J.-M., Lee, C.-H.: 'Generalized subspace pursuit for signal recovery from multiple-measurement vectors'. 2013 IEEE Wireless Communications and Networking Conf. (WCNC), Shanghai, China, 2013

[14] Sundman, D., Chatterjee, S., Skoglund, M.: 'Greedy pursuits for compressed sensing of jointly sparse signals'. 19th European Signal Processing Conf., 2011, pp. 368–372

[15] Schmidt, R.: 'Multiple emitter location and signal parameter estimation', *IEEE Trans. Antennas Propag.*, 1986, **34**, (3), pp. 276–280

[16] Lee, K., Bresler, Y., Junge, M.: 'Subspace methods for joint sparse recovery', *IEEE Trans. Inf. Theory*, 2012, **58**, (6), pp. 3613–3641

[17] Wen, Z., Hou, B., Jiao, L.: 'Joint sparse recovery with semisupervised music', *IEEE Signal Process. Lett.*, 2017, **24**, (5), pp. 629–633

[18] Stanković, S., Orović, I.: 'An approach to 2D signals recovering in compressive sensing context', *Circuits Syst. Signal Process.*, 2017, **36**, (4), pp. 1700–1713

[19] Lekić, N., Žarić, M.L., Orović, I., *et al.*: 'Adaptive gradient-based analog hardware architecture for 2D under-sampled signals reconstruction', *Microprocess. Microsyst.*, 2018, **62**, pp. 72–78

[20] Coluccia, G., Ravazzi, C., Magli, E.: '*Compressed sensing for distributed systems*' (Springer, New York, USA, 2015)

[21] Sundman, D., Chatterjee, S., Skoglund, M.: 'Methods for distributed compressed sensing', *J. Sensor Actuator Netw.*, 2014, **3**, (1), pp. 1–25

[22] Ravazzi, C., Fosson, S.M., Magli, E.: 'Distributed iterative thresholding for ℓ_0/ℓ_1-regularized linear inverse problems', *IEEE Trans. Inf. Theory*, 2015, **61**, (4), pp. 2081–2100

[23] Sundman, D., Chatterjee, S., Skoglund, M.: 'Design and analysis of a greedy pursuit for distributed compressed sensing', *IEEE Trans. Signal Process.*, 2016, **64**, (11), pp. 2803–2818

[24] Cao, D.-Y., Yu, K., Zhuo, S.-G., *et al.*: 'On the implementation of compressive sensing on wireless sensor network'. 2016 IEEE First Int. Conf. on Internet-of-Things Design and Implementation (IoTDI), Berlin, Germany, 2016, pp. 229–234

[25] Brunelli, D., Caione, C.: 'Sparse recovery optimization in wireless sensor networks with a sub-Nyquist sampling rate', *Sensors*, 2015, **15**, (7), pp. 16654–16673

[26] Amarlingam, M., Mishra, P.K., Prasad, K.D., *et al.*: 'Compressed sensing for different sensors: a real scenario for WSN and IoT'. 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT), Reston, VA, 2016, pp. 289–294

[27] Ravelomanantsoa, A., Rabah, H., Rouane, A.: 'Simple and efficient compressed sensing encoder for wireless body area network', *IEEE Trans. Instrum. Meas.*, 2014, **63**, (12), pp. 2973–2982

[28] Chung, W.-Y., Villaverde, J.F.: 'Implementation of compressive sensing algorithm for wireless sensor network energy conservation'. Proc. of the 1st Int. Electronic Conf. Sensors Applications, 2014

[29] Tafa, Z., Ramadani, F., Cakolli, B.: 'The design of a zigbee-based greenhouse monitoring system'. 2018 7th Mediterranean Conf. on Embedded Computing (MECO), Budva. Montenegro, 2018

[30] Griffin, A., Tsakalides, P.: 'Compressed sensing of audio signals using multiple sensors'. Proc. 16th European Signal Processing Conf., 2008, pp. 1–5

[31] Sarvotham, S., Baron, D., Wakin, M., *et al.*: 'Distributed compressed sensing of jointly sparse signals'. Asilomar Conf. on Signals, Systems and Computers, Pacific Grove, CA, 2005

[32] Wimalajeewa, T., Varshney, P.K.: 'Application of compressive sensing techniques in distributed sensor networks: a survey', arXiv e-prints, (Sep 2017), arXiv:1709.10401

[33] Park, S., Yu, N.Y., Lee, H.-N.: 'An information theoretic study for noisy compressed sensing with joint sparsity model-2', arXiv preprint arXiv:1604.00742, 2016

[34] Van Den Berg, E., Friedlander, M.P.: 'Theoretical and empirical results for recovery from multiple measurements', *IEEE Trans. Inf. Theory*, 2010, **56**, (5), pp. 2516–2527

[35] Caione, C., Brunelli, D., Benini, L.: 'Distributed compressive sampling for lifetime optimization in dense wireless sensor networks', *IEEE Trans. Ind. Inf.*, 2012, **8**, pp. 30–40

[36] Abdel-Sayed, M.M., Khattab, A., Abu-Elyazeed, M.F.: 'RMP: reduced-set matching pursuit approach for efficient compressed sensing signal reconstruction', *J. Adv. Res.*, 2016, **7**, (6), pp. 851–861

[37] Björck, A.: '*Numerical methods for least squares problems*' (Siam, Philadelphia, USA, 1996)