ORIGINAL ARTICLE

RMP: Reduced-set matching pursuit approach for efficient compressed sensing signal reconstruction

Michael M. Abdel-Sayed, Ahmed Khattab *, Mohamed F. Abu-Elyazeed

Electronics and Communications Engineering Department, Faculty of Engineering, Cairo University, Giza 12613, Egypt

GRAPHICAL ABSTRACT

Compressed sensing enables the acquisition of sparse signals at a rate that is much lower than the Nyquist rate. Compressed sensing initially adopted $\ell_1$ minimization for signal reconstruction which is computationally expensive. Several greedy recovery algorithms have been recently proposed for signal reconstruction at a lower computational complexity compared to the optimal $\ell_1$ minimization, while maintaining a good reconstruction accuracy. In this paper, the Reduced-set Matching Pursuit (RMP) greedy recovery algorithm is proposed for compressed sensing. Unlike existing approaches which either select too many or too few values per iteration, RMP aims at selecting the most sufficient number of correlation values per iteration, which improves both the reconstruction time and error. Furthermore, RMP prunes

A preliminary basic version of the RMP is accepted for presentation in IEEE International Conference on Image Processing (ICIP) 2016.

* Corresponding author. Fax: +202 3572 3486.

E-mail address: akhattab@ieee.org (A. Khattab).

Peer review under responsibility of Cairo University.

http://dx.doi.org/10.1016/j.jare.2016.08.005

2016 Production and hosting by Elsevier B.V. on behalf of Cairo University.

This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
Introduction

In order to perfectly reconstruct a signal from its samples, the signal is to be sampled at least at the Nyquist rate, which is double the signal’s highest frequency component. However, the Nyquist rate has two shortcomings. First, the Nyquist rate of many contemporary applications is so high that it is too expensive or even impossible to implement [1]. Second, the large number of acquired samples are not fully used in the reconstruction process or partially sacrificed. Recall that many applications have to further compress the sampled signal for efficient storage purposes or for transmission over a much limited bandwidth. For example, a typical digital camera has millions of imaging sensors, whereas the acquired image is usually compressed into a few hundred kilobytes. Thus, a significant amount of the acquired data—more specifically, the significant correlation content—is sacrificed [2].

Recently, compressed sensing has presented itself as an efficient sampling technique that samples the signals at a much lower rate compared to the Nyquist rate. Compressed sensing simultaneously performs sensing and compression; thus, the signal is sensed in a compressed form [1–7]. This results in a considerable reduction in the number of measurements that need to be stored and/or processed. Compressed sensing is applicable to either sparse or compressible signals which typically have few significant coefficients in a suitable basis or domain (e.g., Fourier and Wavelets). This includes a large variety of signals such as natural images, videos, MRI, and radar signals [8]. The original signal can be recovered by convex optimization or greedy recovery algorithms.

Several greedy recovery algorithms have been recently developed for sparse signal reconstruction [9–13]. These algorithms aim to reduce the computational complexity of the optimum \( \ell_1 \) minimization, while maintaining a high reconstruction accuracy. Such algorithms iteratively identify the signal support (its nonzero indices) by correlating the measured signal with the sensing matrix columns. A number of correlation values are selected in each iteration, and their indices are added to a set of identified supports. Existing algorithms perform selection from the whole correlation vector, which increases the reconstruction time. Furthermore, the majority of the existing algorithms perform non-tunable selection, which results in selecting either too few or too many elements, causing larger reconstruction time and error.

In this paper, the Reduced-set Matching Pursuit (RMP), a new thresholding-based greedy signal reconstruction algorithm for compressed sensing is introduced by extending the algorithm in Abdel-Sayed et al. [14]. As a greedy recovery algorithm, RMP forms an estimate of the support of the sparse signal in each iteration. Unlike the related algorithms, RMP efficiently estimates the signal support by selecting values from a reduced set of the correlation vector. Furthermore, the selection is performed in a signal-aware manner. That is, the number of selected elements per iteration changes based on the distribution of the correlation values. Therefore, RMP targets the selection of a sufficient number of elements per iteration. The signal is then estimated using least square minimization with nonzeros at indices from the identified support set. The signal is then pruned to exclude the incorrectly selected elements. The residual is calculated from the pruned signal, and the previous steps are repeated until a stopping condition is met. Simulation results show that RMP has a high reconstruction accuracy at a significantly low computational complexity compared to existing greedy recovery algorithms. Moreover, RMP is capable of sparse signal reconstruction from noiseless samples as well as from samples contaminated with additive noise. More specifically, the normalized time-error product of RMP is 87% to 95% less than that of \( \ell_1 \) minimization at high sparsity levels in the absence of noise. In the noisy samples case, the RMP normalized time-error product is 57% to 98% less than that of \( \ell_1 \) minimization depending on the signal to noise ratio (SNR).

Compressed sensing fundamentals

Consider a sparse signal \( x \in \mathbb{R}^n \) of sparsity level \( k \). A measurement system that samples this signal to acquire \( m \) linear measurements is typically modeled as:

\[
y = \Phi x, \tag{1}
\]

where \( \Phi \in \mathbb{R}^{m \times n} \) is the sensing or measurement matrix, and \( y \in \mathbb{R}^m \) is the measured vector or the samples.

Alternatively, the signal \( x \) may not be itself sparse, but it may be sparse in a certain basis \( \Psi \), i.e., \( x = \Psi s \), where \( s \) is a sparse vector. In this case, (1) is rewritten as:

\[
y = \Phi \Psi s = As, \tag{2}
\]

where \( \Psi \) is an \( n \times n \) matrix which columns form a basis in which \( x \) is sparse, and \( A = \Phi \Psi \) is an \( m \times n \) matrix.

Unlike legacy measurement systems, \( m \) is much less than \( n \) in compressed sensing as the dimension of the measured vector \( y \) is much lower than the dimension of the original signal \( x \). Yet, it was shown that the sparse (or compressible) signal \( x \) can be recovered using the few measurements captured by \( y \) provided that the sensing matrix satisfies the Restricted Isometry Property (RIP) [1,3].

A matrix \( A \) satisfies the restricted isometry property of order \( k \) if there exists a \( \delta_k \in (0, 1) \) such that:

\[
(1 - \delta_k) \|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta_k) \|x\|_2^2 \tag{3}
\]
holds for all \( k \)-sparse signals \( x \), where \( \| x \|_2 \) is the \( \ell_2 \) norm of the signal \( x \).

Random matrices of certain distributions satisfy the RIP with high probability [15]. More specifically, if the entries of a matrix are independent and identically distributed (i.i.d.) and follow a Gaussian, Bernoulli or sub-Gaussian distribution, the probability that the matrix does not satisfy the RIP is exponentially small.

The natural, and the most straightforward, approach to recover a sparse signal from a few set of measurements is by solving an \( \ell_0 \) norm optimization problem. However, the objective function of the \( \ell_0 \) optimization problem is nonconvex, and hence, finding the solution that approximates the true minimum is NP-hard [4]. One way to transform this NP-hard problem into something more tractable is to replace the \( \ell_0 \) norm with its convex approximation \( \ell_1 \) norm. In this case, the transformed problem can be solved as a linear program.

Donoho [4] suggested minimizing the \( \ell_1 \) norm \( \| \cdot \|_1 \) to reconstruct the sparse signal as follows:

\[
\hat{x} = \arg \min_z \| z \|_1 \text{ subject to } y = \Phi z.
\]  

In practice, the measured samples are typically contaminated with additive noise. In this case, the measured vector is given by

\[
y = \Phi x + e,
\]

where \( e \) is the sample noise and \( \| e \|_2 < \epsilon \). \( \ell_1 \) minimization can still be used to reconstruct the original sparse signal \( x \) with an error that cannot exceed the noise level \( \epsilon \) as follows [16]:

\[
\hat{x} = \arg \min_z \| z \|_1 \text{ subject to } \| y - \Phi z \|_2 \leq \epsilon.
\]

In both the noiseless sample and noisy sample cases, \( \ell_1 \) minimization is a powerful solution for the sparse problem. However, this solution is computationally expensive [1].

**Greedy recovery algorithms**

Motivated by the need to develop computationally inexpensive solutions, various greedy algorithms have been proposed in the literature for signal recovery. Greedy recovery algorithms iteratively attempt to find the signal support. In each iteration, the sparse signal is estimated based on the identified support set through least square minimization. Fig. 1 shows a generic block diagram of the main steps for such greedy algorithms. The function of each block is briefly described as follows:

1. **Correlation:** The residual \( r \) is correlated with the columns of the sensing matrix \( \Phi \) to form a proxy signal \( g \).

\[ y \rightarrow r \rightarrow \Phi \rightarrow g \]

**Fig. 1** General block diagram of recovery algorithms.

2. **Selection and support merging:** One or more of the elements of \( g \) with the largest absolute values are selected in each iteration. The indices of the selected elements are merged into the identified support set which is used to approximate the signal.

3. **Signal estimation:** The sparse signal is estimated based on the identified support using least square minimization. Some algorithms (thresholding-based algorithms) perform a pruning step to the estimated signal, keeping only the \( k \) largest absolute values of the signal, and setting the rest to zeros.

4. **Residual calculation:** The residual is calculated based on the estimated signal.

Greedy recovery algorithms can be classified into threshold-less algorithms and thresholding-based algorithms depending on whether or not they prune the estimated signal by applying a hard thresholding operator. In what follows, the main existing algorithms in each category are discussed and summarized in Fig. 2.

**Threshold-less greedy recovery algorithms**

The first greedy recovery algorithm is the Basic Matching Pursuit (BMP) [1,17]. BMP selects only one element from the correlation vector per iteration, and adds its index to the identified support set. However, the residual is calculated without performing least square minimization, which results in higher reconstruction error. Another simple greedy recovery algorithm is the Orthogonal Matching Pursuit (OMP) [9,18]. OMP performs least square minimization to estimate the signal, which results in improvement over BMP. However, OMP selects only one element from the correlation vector per iteration as in BMP. For a \( k \)-sparse signal, OMP needs \( k \) iterations in order to reconstruct the signal.

Alternatively, other algorithms add more than one index per iteration, resulting in a faster convergence time. For instance, the Generalized Orthogonal Matching Pursuit (GOMP) selects a fixed number of elements per iteration [10]. Meanwhile, the Regularized Orthogonal Matching Pursuit (ROMP) chooses a set of \( k \) largest nonzero elements, then divides them into groups of comparable magnitudes and selects the group of maximum energy [19,20]. The Stagewise Weak Orthogonal Matching Pursuit (SWOMP) selects the elements with absolute values larger than or equal to \( \max(|g|) \), where \( 0 < z < 1 \) and \( \max(|g|) \) is the largest magnitude element.
in the correlation vector [21]. The Stagewise Orthogonal Matching Pursuit (StOMP) [22] selects the elements larger than a certain configurable value determined by the constant false alarm rate (CFAR) strategy originally developed for radar systems [11].

Other algorithms exploit the structure of the signal sparsity such as the Tree-based Orthogonal Matching Pursuit (TOMP) [23–25]. On the other hand, the Multipath Matching Pursuit models the problem of finding the candidate support of the signal as a tree search problem [26]. Finally, it is worth mentioning that some algorithms that fall under this category speed up the minimization step using iterative matrix inversion techniques [27].

**Drawbacks of threshold-less greedy algorithms**

Since BMP and OMP add only one index per iteration, they require a larger number of iterations than the rest of the algorithms. While ROMP improves the speed of OMP by selecting multiple elements per iteration, its reconstruction error is larger, especially for higher sparsity levels. The algorithm often results in adding a larger number of indices per iteration than is necessary, which usually includes ones not belonging to the support of the original signal. SWOMP and StOMP attempt to improve the selection stage by using different selection strategies. However, SWOMP still suffers from the same drawback of ROMP. Meanwhile, StOMP gives closer error performance to OMP, while requiring less execution time for higher sparsity levels. It is worth noting that none of the aforementioned algorithms contain a pruning step. Thus, incorrectly selected indices will appear in the signal estimate, which degrades the performance reflected by a deterioration in the reconstruction accuracy.

**Thresholding-based greedy recovery algorithms**

A common drawback in all the aforementioned greedy algorithms is that if an incorrect index is added to the support set in a certain iteration, it remains in all subsequent iterations, possibly degrading the performance. Thresholding-based algorithms handle this problem by applying a hard thresholding operator which removes one or more of the indices having the least energy from the identified support set. An example is the Compressive Sampling Matching Pursuit (CoSaMP) [12], which selects 2k elements per iteration and performs pruning after signal estimation. The Subspace Pursuit (SP) is another thresholding-based algorithm which selects k elements per iteration [13]. Pruning is then performed, followed by an extra least square minimization step. Iterative Hard Thresholding (IHT) is another thresholding-based recovery algorithm which recursively solves the sparse problem while applying the hard thresholding operator [28,29].

**Drawbacks of thresholding-based greedy algorithms**

Thresholding-based algorithms such as CoSaMP and SP add a pruning step at the end of each iteration. However, such algorithms select a fixed number of elements per iteration (e.g. 2k in CoSaMP and k in SP). Such a selection is constant for all iterations and does not adapt to the distribution of the values of correlation. Furthermore, it usually results in selecting too many elements causing a larger reconstruction time, since more than necessary components are sorted in each iteration. A large and fixed selection further increases the iteration time as more than necessary nonzero values have to be estimated by least square minimization. Selecting too many elements also reduces the accuracy of the signal estimate, especially for larger sparsity and when working on a noisy measurement, when incorrect indices are selected and kept through the subsequent pruning steps. Finally, the iterative nature combined with sacrificing the least square minimization step in the IHT algorithm results in an increased reconstruction time and error.

The rest of this paper is organized as follows. The RMP algorithm is proposed in the “Reduced-set Matching Pursuit” Section, and thoroughly evaluates its different performance aspects in the “Performance Evaluation and Discussions” Section. Section “Conclusions” concludes the paper.

**Reduced-set matching pursuit**

In this section, the Reduced-set Matching Pursuit (RMP), a thresholding-based greedy recovery algorithm is presented. RMP main goal is to reconstruct a sparse signal x from measurements given by (1) or (2) as accurately and efficiently as possible. In order to achieve these goals RMP performs 4 main steps. First, RMP iteratively identifies the support of the sparse signal by appropriately selecting elements from a significantly reduced set of the correlation values. This contrasts with existing algorithms in which the selection is performed from the whole correlation vector and is performed in a signal-agnostic manner in the majority of existing algorithms. Second, RMP estimates the sparse signal based on the identified support set. Even though RMP uses least square minimization to estimate the signal, its convergence time is much less than existing techniques since RPM least square minimization targets a significantly reduced set of indices. Third, RMP uses pruning to exclude the incorrectly selected elements, and hence, prevent such erroneous selections from degrading the performance. Fourth, a residual is then calculated to remove the estimated part from the measurement vector. These steps are repeated until a stopping criterion is met.

**RMP components**

In what follows, the four main components of the RMP algorithm are explained in detail.

**Support identification**

In order to reconstruct the sparse signal, its support (nonzero indices) needs to be identified. This is done iteratively, where in each iteration the identified support set is updated. First, the measured vector y is correlated with the columns of the sensing matrix Φ to obtain a correlation vector g. The non-zero indices of the sparse signal are expected to have relatively large magnitudes of correlation. Thus, some of the highest magnitude elements of the correlation vector are selected according to a specific “selection strategy”. The indices of the selected elements are merged with the identified support set.

The selection strategy is one of the main factors on which the performance of the recovery algorithm depends. The selection stage should be able to select elements corresponding to
nonzero indices of the original sparse signal. It should not select too few elements, which leads to an excessively large number of iterations, which in turn causes a larger reconstruction time. Nor should it select too many elements, which leads to performing calculations on a much larger amount of data (which includes sorting, matrix inversion, and least square minimization). Not only does this increase the reconstruction time, but it also causes the selection of elements which indices do not belong to the support of the original signal, which leads to an increase in the reconstruction error. Therefore, it is necessary for the algorithm to achieve a compromise in the number of selected elements per iteration. Existing techniques either select too few elements [9,10,18] or too many elements [12,13,19,20,22], which increases their reconstruction time or reduces their reconstruction accuracy respectively.

In contrast, RMP targets the selection of a sufficient enough number of elements using a double thresholding technique. RMP selects the indices which most likely belong to the support of the original signal, without taking too few or too many indices per iteration. Based on the distribution of the absolute values of \( g \), the number of selected elements is not constant for all iterations (even though \( x \) and \( \beta \) are constants). For steeper distributions of the absolute values of \( g \), fewer elements are selected. For flatter distributions, more elements are selected.

RMP achieves this goal in two steps. First, the elements from which selection is performed are reduced to a set containing the \( \beta k \) top magnitude elements. Then, elements whose magnitudes are larger than a fixed fraction \( 0 < \alpha < 1 \) of the maximum element are selected from the reduced set, and their indices are added to the support set. The proper selection of the constant values of the \( x \) and \( \beta \) parameters leads to the selection of an optimum number of elements per iteration, which in turn contributes to a high reconstruction accuracy and a low reconstruction complexity.

**Signal estimation**

After the selection and support merging stage, a new signal estimate \( \hat{x} \) is formed based on the merged support set. This is performed using least square minimization. That is, the algorithm finds the signal \( \hat{x} \) which minimizes \( ||y - \Phi \hat{x}||_2 \), while having non-zeros at the indices obtained from the identified support set. Such minimization is done via the multiplication of the pseudo-inverse given by

\[
\Phi^*_T = (\Phi_T^T\Phi_T)^{-1}\Phi_T^T,
\]

where \( \Phi_T \) is a matrix that contains the columns of \( \Phi \) with indices in the identified support set \( T \). It should be noted here that the calculation of the pseudo-inverse requires the inversion of a matrix whose size is dependent on the number of indices in the identified support set. Since RMP selects an optimum number of elements per iteration, which is much smaller than that selected by other existing algorithms, the size of the matrix is smaller, and the reconstruction is faster.

**Pruning**

Next, the estimated signal is pruned. Pruning is a technique that is used to enhance the performance of recovery algorithms [12]. Recovery algorithms inevitably select one or more elements whose indices do not belong to the support set of the original signal during the reconstruction process. Without pruning, such elements remain in the signal estimate during the consecutive iterations, which reduces the reconstruction accuracy. Hence, convergence is slower and the reconstruction time is generally affected.

In RMP, the estimated signal is pruned by removing the elements which have the least contribution to the estimated signal from the identified support set. RMP only keeps those corresponding to the \( k \) largest magnitude components of the estimated signal. The benefit of the pruning step is even more evident in the reconstruction of signals from samples contaminated with noise.

**Residual calculation**

A residual is then calculated by subtracting the contribution of the estimated signal from the measured vector. The residual is given by

\[
r = y - \Phi \hat{x}.
\]

This residual is then correlated with the columns of the sensing matrix for the successive iterations. The previous steps are repeated until a stopping criterion is met. RMP terminates if the norm of the residual is less than \( \epsilon_1 \) or if the difference between the norms of the residuals in two successive iterations is less than \( \epsilon_2 \), whichever occurs first. Otherwise, a maximum of \( k \) iterations are performed.

**RMP algorithm**

Initially, the signal estimate is set to a zero vector and the residual to the measured vector \( y \). In each iteration, the following steps are performed:

1. **Signal proxy formation:** A signal proxy, \( g \), is formed by correlating the residual with the sensing matrix columns.
2. **Selection and support merging:** The vector \( g \) is sorted in a descending order of absolute values. The elements whose absolute values are larger than or equal to \( \alpha \max_j ||g_j|| \), where \( 0 < \alpha < 1 \), are selected from a reduced set containing the \( \beta k \) largest magnitude elements. The indices of the selected elements are united with the already identified support set.
3. **Signal estimation:** An estimate of the signal is formed by least square minimization. This is done via multiplication by the pseudo-inverse of the sensing matrix.
4. **Pruning:** The \( k \) largest magnitude components in the signal estimate are retained. The rest are set to zero.
5. **Residual calculation:** The new residual is calculated from the pruned signal.

At the end of each iteration, the RMP algorithm checks whether the norm of the residual is less than \( \epsilon_1 \) or whether the difference between the norms of the residuals in two successive iterations is less than \( \epsilon_2 \). If either condition is met, the RMP algorithm terminates. Otherwise, RMP terminates after a maximum of \( k \) iterations.

Algorithm 1 summarizes the RMP algorithm. 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.

**Algorithm 1.** Reduced-set Matching Pursuit.

| Input: Sensing matrix $\Phi$, measurement vector $y$, sparsity level $k$, parameters $x$ and $\beta$. |
|---|
| **Initialize:** $\hat{x}^{(0)} = 0$, $r^{(0)} = y$, $T^{(0)} = \emptyset$. |
| for $i = 1; i \leq i + 1$ until the stopping criterion is met do |
| $g^{(i)} = \Phi^T r^{(i)}$ \{Form signal proxy\} |
| $J = \{j : |g^{(i)}| \geq \beta \max_{j' \in J} |g^{(i)}| \}$ \{Indices of $\beta$ largest magnitude elements in $g$\} |
| $W = \{j : |g^{(i)}| \geq \beta \max_{j' \in J} |g^{(i)}|, j \in J \}$ \{Indices of elements in $J$ larger than or equal to $\beta \max |g^{(i)}|\} |
| $T = W \cup \supp(\hat{x}^{(i-1)})$ \{Support merging\} |
| $b_1 = \Phi \hat{x}^{(i)}$, $b_2 = 0$ \{Signal estimation\} |
| $\hat{x}^{(i)} = H_k(b)$ \{Prune approximation\} |
| $r^{(i)} = y - \Phi \hat{x}^{(i)}$ \{Update residual\} |
| end for |
| **Output:** Reconstructed signal $\hat{x}$ |

**The effect of $\alpha$ and $\beta$**

The performance of the RMP algorithm is governed by the proper selection of its $\alpha$ and $\beta$ parameters. Here, the effect of $\alpha$ and $\beta$ on the performance of RMP is discussed. In the Performance Evaluation Section, simulations are used to obtain their best value ranges and verify that the RMP algorithm performance is not sensitive to a particular choice in such a range. There are three different ranges for $\alpha$ for which the performance drastically changes.

First, when $\alpha$ is very small and close to zero, all the elements in the reduced set are selected. Having large values of $\beta$ in this case may improve the performance, but will cause a larger reconstruction time. This is due to the selection of a larger number of indices per iteration than what is necessary. For small $\alpha$ and small values of $\beta$, the reconstruction error is larger, since a very small number of indices are selected, which is not enough to select the correct support of the signal. Furthermore, a larger number of iterations are required, which in turn leads to a larger reconstruction time.

Second, for larger values of $\alpha$ close to 1, the number of selected indices per iteration is too small. Thus, a large number of iterations are required and the reconstruction time is larger regardless the value of $\beta$.

Third, when $\alpha$ is neither too close to 0 nor too close to 1, the best compromise is achieved. The number of selected elements per iteration are neither too large (as in the first case) nor too small (as in the second one). Such a moderate choice of $\alpha$ will also relax the requirements on $\beta$ which will also tend to be moderate as there will be no need to select a large number of indices. This leads to improvements in the reconstruction time and accuracy. Simulation results show that the exact choice of the $\alpha$ and $\beta$ values in this moderate range does not significantly affect the performance.

**Noise robustness**

In many signal reconstruction applications, the measured samples are contaminated with additive white noise. Therefore, it is necessary for the recovery algorithm to be able to reconstruct the sparse signal from noisy samples as accurately as possible. Next, the reconstruction capability of RMP when the measured samples are contaminated with additive white noise as given by (5) is discussed.

Since the measured signal $y$ is contaminated with noise, the correlation vector $g$ is noisy as well. This may result in the selection of incorrect elements from $g$ in some iterations, depending on the signal-to-noise ratio (SNR). The higher the SNR, the higher the probability of selecting incorrect elements, and vice versa. Consequently, a signal estimate is formed with some elements of the support set at incorrect indices. Now, if the recovery algorithm does not have a pruning step, there is no way to exclude such elements from the identified support set, and the performance of the algorithm will deteriorate. On the other hand, algorithms which have a pruning step, such as RMP, are capable of excluding incorrectly added elements in each iteration, and iterating until the correct ones are found. Thus a more accurate estimate of the support set is generated, and consequently a more accurate estimate of the signal is formed. Such incorrectly identified elements are pruned with high probability after the signal estimate is formed, since they have the least contribution to the original signal.

Furthermore, RMP selects a smaller number of elements per iteration, compared to other thresholding-based algorithms that perform pruning, causing its performance to be more robust in the presence of noise. This is because selecting a larger number of noisy elements than is necessary per iteration (as the case with other related algorithm) makes such algorithms more error-prone. Recall that the pruning step excludes the elements of the support set which have the least contribution to the estimated signal. When there are too many elements present in the noisy signal estimate, pruning may keep some of the incorrectly added ones due to noise. This results in a larger error for lower SNR levels for such algorithms. Therefore, RMP outperforms other thresholding-based algorithms in applications that suffer from noise.

**Performance metrics**

In the next section, the performance of RMP against existing related techniques as well as the original $\ell_1$ minimization is evaluated. The used performance metrics are as follows:

- The reconstruction time $t$ in seconds, which is the time required to reconstruct the sparse signal from the measurement signal.
- The reconstruction error $e$, which is the reconstruction error relative to the $\ell_2$ norm of the signal defined as $\|x - \hat{x}\|_2/\|x\|_2$. We introduce the normalized time-error product in which the product of the time and error of each algorithm is normalized over the largest product value of all algorithms, that is:

$$\text{Normalized time} - \text{error product} = \frac{t_i \cdot e_i}{\max_j \{t_j \cdot e_j\}}, \quad (9)$$

where $t_i$ and $e_i$ are the reconstruction time and reconstruction error of algorithm $i$ at sparsity level $j$, respectively. This metric accounts for the trade-off between time and error, since some algorithms give higher reconstruction accuracy at the expense of higher computational complexity.
Other metrics are also considered that help understand the differences in the dynamics of how each algorithm reconstructs the original signal such as:

- The number of iterations performed by the algorithm.
- The average number of selected elements per iteration.
- The average size of the merged support set. For thresholding-based algorithms, this is taken before pruning for the sake of fairness in comparison.

**Performance evaluation and discussions**

**Simulation setup**

In this section, the performance of the proposed RMP algorithm against the performance of the following algorithms: $\ell_1$ minimization, OMP, ROMP, IHT, SWOMP, StOMP, SP, and CoSaMP is illustrated via MATLAB simulations. For each algorithm, the reported results are the average of the

---

**Fig. 3** Impact of $\alpha$ and $\beta$ on (a) reconstruction time, (b) reconstruction error, (c) number of iterations, (d) the average number of selected elements per iteration, and (e) Normalized time-error product at a sparsity level of 70.
metrics evaluated for 100 independent trials. In each trial, a random sparse signal of length \( n = 1000 \) of uniformly distributed integers from 0 to 100 is generated. This paper only presents the results of \( m = 250 \) measurements. The results of other values of \( m \) are omitted since similar observations were obtained. The only difference is that as \( m \) increases (or decreases), the errors occur at higher (or lower) sparsity levels.

The sensing matrix \( \Phi \) of dimensions \( m \times n \) is randomly generated from i.i.d. Gaussian distribution with columns having unit \( \ell_2 \) norm.

For SWOMP, \( \alpha = 0.7 \) is used, which is the same value used in [21]. For IHT, the step size, \( \mu \), is tuned by obtaining the metrics at a sparsity level of 70 using values of \( \mu \) ranging from 0.1 to 1 with 0.1 steps. It was found that \( \mu = 0.3 \) results in the least normalized time-error product; therefore, this value is used for IHT in the following simulations. For SoOMP, the implementation that is available as a part of the SparseLab Toolbox for Matlab is used.

For the noiseless case, the results of the different metrics for sparsity levels ranging from 10 to 150 are reported. For the noisy case, AWGN is added to the measured samples at different values of SNR. The results of the metrics against SNR from \(-10\) dB to \(50\) dB at a sparsity level of 70 are reported.

The effect of \( \alpha \) and \( \beta \)

Before comparing the performance of RMP against the other existing algorithms, the effect of its \( \alpha \) and \( \beta \) parameters is studied first to obtain their best values. In order to study the effect of the \( \alpha \) and \( \beta \) parameters, the value of \( \alpha \) is varied from 0.1 to 1 with 0.1 steps, and the value of \( \beta \) from 0.05 to 2 with 0.1 steps. The different performance aspects (namely, reconstruction time, error, the number of iterations, the number of selected elements per iteration, and the normalized time-error product) metrics are depicted for the different pair of \((\alpha, \beta)\) in Fig. 3(a) to (e), respectively. These results are averaged over 100 independent trials per \((\alpha, \beta)\) pair at different sparsity levels. Only the results at a sparsity level of 70 are reported here. However, similar results and conclusions were obtained at the other sparsity levels.

For smaller values of \( \alpha \) up to 0.5, values of \( \beta \) larger than 0.75 cause larger reconstruction time, as shown in Fig. 3(a). As explained in the previous section, a larger number of indices per iteration are selected as illustrated in Fig. 3(d). For very small values of \( \beta \) with small \( \alpha \) value, the reconstruction error is larger as depicted in Fig. 3(b). A very small number of indices are selected and a larger number of iterations are required, as shown in Fig. 3(c), which in turn leads to a larger reconstruction time. For such low values of \( \alpha \), values of \( \beta \) ranging from about 0.15 to 0.75 give the smallest normalized time-error product as depicted in Fig. 3(e).

In the other end of values of \( \alpha \) ranging from 0.8 to 1, the number of selected indices per iteration is too small. Thus, a large number of iterations are required, and hence, the reconstruction time is larger.

In contrast, values of \( \alpha \) ranging from 0.5 to 0.7 give the best performance compromise. The number of selected elements per iteration is neither too large, as in the first range, nor too small, as in the second one. For this range, \( \beta \) ranging from about 0.15 to 0.75 gives the smallest normalized time-error product.

It is noted that the performance of the algorithm is not very sensitive to the values of \( \alpha \) and \( \beta \) as long as they are in the aforementioned optimum range. It can be also noted that as the value of \( \alpha \) increases, the effect of \( \beta \) becomes less evident. This is due to the fact that the number of selected indices is mainly limited by \( \alpha \) in this case. Similar results are obtained for sparsity levels ranging from 50 to 100. The values \( \alpha = 0.7 \) and \( \beta = 0.5 \) are used for the other sparsity levels until 150.
Reduced-set matching pursuit signal reconstruction

Table 1 Normalized time-error product ×100 (noiseless case).

| Sparsity | 60 | 70 | 80 | 90 | 100 | 110 | 120 | 130 | 140 | 150 |
|----------|----|----|----|----|-----|-----|-----|-----|-----|-----|
| L1 Norm  | 0.00| 0.00|0.00|0.10|1.09 |2.24 |3.25 |4.02 |4.32 |4.95 |
| OMP      | 0.01| 0.05|0.20|0.48|0.90 |1.31 |1.62 |1.92 |2.37 |2.86 |
| ROMP     | 0.05| 0.20|0.33|0.25|0.27 |0.31 |0.27 |0.27 |0.25 |0.27 |
| IHT      | 0.25| 0.55|0.89|1.16|1.43 |1.80 |2.04 |2.39 |2.77 |3.17 |
| SWOMP    | 0.00| 0.01|0.13|0.26|0.35 |0.39 |0.37 |0.40 |0.43 |0.45 |
| StOMP    | 0.00| 0.02|0.11|0.21|0.26 |0.30 |0.31 |0.31 |0.29 |0.30 |
| SP       | 0.00| 0.00|0.04|0.20|0.64 |2.15 |8.09 |12.75|16.53|21.80|
| CoSaMP   | 0.00| 0.01|1.62|100 |21.96|23.28|27.23|29.93|34.27|39.53|
| RMP      | 0.00| 0.00|0.03|0.09|0.14 |0.18 |0.21 |0.22 |0.24 |0.27 |

The highlighted cells represent the least normalized time-error product.

and $\beta = 0.25$ are selected to be used in the rest of the simulations.

Performance comparison

In what follows, the simulations results that demonstrate the performance advantages of RMP compared to other existing algorithms are presented. While the presented plots only show the results of the most relevant algorithms, the results of all the algorithms are also tabulated for interested readers.

Noiseless case

First, the case in which the signal is not contaminated with noise is considered. Fig. 4(a) depicts the reconstruction time versus the signal sparsity level. $\ell_1$ minimization is omitted since it takes considerably longer time. The proposed RMP has the least reconstruction times. This is due to the selection of a just sufficient number of elements per iteration. SWOMP and ROMP achieve slightly higher reconstruction times. It should be noted that both SWOMP and ROMP are not thresholding-based (i.e., they do not perform pruning) which causes larger reconstruction error. The reconstruction time of other thresholding-based algorithms increases rapidly at sparsity levels of 70 for CoSaMP and 100 for SP. This is due to the selection of a larger number of elements.

Fig. 4(b) shows the reconstruction error as a function of the sparsity level. For low sparsity levels, most of the algorithms produce very low errors, giving accurate signal estimates. However, as the sparsity of the signal increases, the differences between the reconstruction capability of the algorithms start to become significant. The optimal $\ell_1$ minimization has the least error – despite its extremely long reconstruction time. The proposed algorithm, RMP, has the lowest error compared to all other greedy algorithms for most of the sparsity levels. However, beyond a sparsity level of about 100, the error for all algorithms is too large to be used in practical applications.

The proposed normalized time-error product metric captures both performance aspects. Fig. 4(c) shows the normalized time-error product as a function of sparsity. RMP has the smallest product for most sparsity levels except for sparsity levels around 80 where $\ell_1$ minimization is slightly smaller. This means that RMP achieves a high reconstruction accuracy at low complexity compared to other algorithms including $\ell_1$ minimization (which achieves slightly higher accuracy but at the expense of significantly longer time). Table 1 lists the normalized time-error product of all the simulated algorithms for noiseless samples.

Noisy case

Next, the case in which the signal is contaminated with additive noise is considered. Fig. 5(a) depicts the reconstruction time versus the SNR for the noisy case. RMP has the least reconstruction time for all values of SNR values. Again the graph for $\ell_1$ minimization is omitted since it is considerably higher than the rest of the algorithms.

Fig. 5(b) illustrates the error for the noisy case. $\ell_1$ minimization has the lowest error for higher values of SNR, followed by RMP. For lower SNR, RMP and SP give the least error. It can be seen that SWOMP, StOMP, and ROMP have high reconstruction error, especially at lower values of SNR. This is due to the fact that they do not perform pruning. While CoSaMP performs pruning, the large number of selected elements per iteration makes it more error-prone.

Fig. 5(c) shows the normalized time-error product for the noisy case. As with the noiseless case, RMP has the smallest product for all SNR levels in the noisy case. This implies that RMP is more robust against noise compared to the rest of the algorithms as it has a high reconstruction accuracy at a low complexity – even under low SNR levels. Table 2 lists the full normalized time-error product of all the simulated algorithms for noisy samples.

Dynamics of different algorithms

Finally, the dynamics of the different algorithms are discussed in order to better explain how RMP achieves its outstanding performance. More specifically, the number of iterations taken by each algorithm for the noiseless case, the average number of
selected elements per iteration, and the average size of the merged support set before pruning are investigated.

OMP selects one element per iteration and performs a number of iterations equal to the sparsity level, thus taking a relatively large reconstruction time. Meanwhile, ROMP and SWOMP select a larger number of elements without pruning, thus performing a much smaller number of iterations and requiring much lower reconstruction time. By design, StOMP performs a maximum of a fixed number of iterations, which is set to 10. This leads to a lower reconstruction time than OMP. However, the fact that none of the aforementioned threshold-less algorithms perform pruning leads to a larger error.

Next, the SP, CoSaMP, and RMP thresholding-based algorithms are studied. CoSaMP has the largest merged support set size, followed by SP. This not only causes a larger reconstruction time, but also causes a larger reconstruction error, especially for higher sparsity levels. On the other hand, the selection strategy of RMP results in adding a much smaller number of indices per iteration. This keeps the support set size significantly smaller in successive iterations, giving a relatively lower time and error. While RMP requires a larger number of iterations up to about a sparsity level of 70, the operations are performed on a much smaller amount of data. The overall result is a high reconstruction accuracy at a lower complexity.

Conclusions

This paper has introduced RMP: a new thresholding-based greedy algorithm for signal recovery for compressed sensing applications. RMP targets the selection of just a sufficient number of elements per iteration. This is performed by appropriately selecting elements from a reduced set of correlation values. Pruning is then performed to exclude incorrectly selected elements. Simulation results for both the noiseless and noisy cases have shown that the proposed RMP algorithm is superior to the main existing greedy recovery algorithms both in terms of reconstruction time and accuracy. Furthermore, RMP is even superior to $\ell_1$ minimization in terms of normalized time-error product, a measure which accounts for the trade-off between the reconstruction time and error.

Conflict of interest

The authors have declared no conflict of interest.

Compliance with ethics requirements

This article does not contain any studies with human or animal subjects.

References

[1] Eldar YC, Kutyniok G. Compressed sensing: theory and applications. Cambridge University Press; 2012.
[2] Candès EJ. Compressive sampling. Proceedings of the international congress of mathematicians, vol. 3. p. 1433–52.
[3] Candès EJ, Wakin MB. An introduction to compressive sampling. IEEE Signal Proc Mag 2008;25(2):21–30.
[4] Donoho DL. Compressed sensing. IEEE Trans Inf Theory 2006;52(4):1289–306.
[5] Boche H, Calderbank R, Kutyniok G, Vybíral J. Compressed sensing and its applications. Springer; 2015.
[6] Foucart S, Rauhut H. A mathematical introduction to compressive sensing, vol. 1. Springer; 2013.
[7] Baraniuk RG. Compressive sensing. IEEE Signal Proc Mag 2007;24(4).
[8] Li Z, Xu W, Wang Y, Lin J. A tree-based regularized orthogonal matching pursuit algorithm. In: 22nd International conference on telecommunications (ICT). p. 343–7.
[9] Tropp J, Gilbert AC. Signal recovery from partial information via orthogonal matching pursuit; 2005.
[10] Wang J, Kwon S, Shim B. Generalized orthogonal matching pursuit. IEEE Trans Signal Process 2012;60(12):6202–16.
[11] Richards MA. Fundamentals of radar signal processing. Tata McGraw-Hill Education; 2005.
[12] Needell D, Tropp JA. Cosamp: iterative signal recovery from incomplete and inaccurate samples. Commun ACM 2010;53(12):93–100.
[13] Dai W, Milenkovic O. Subspace pursuit for compressive sensing signal reconstruction. IEEE Trans Inf Theory 2009;55(5):2230–49.
[14] Abdel-Sayed MM, Khattab A, Abu-Elyazed MF. Adaptive reduced-set matching pursuit for compressed sensing recovery. In: IEEE international conference on image processing.
[15] Baraniuk R, Davenport M, DeVore R, Wakin M. A simple proof of the restricted isometry property for random matrices. Construct Approx 2008;28(3):253–63.
[16] Candès EJ, Romberg JK, Tao T. Stable signal recovery from incomplete and inaccurate measurements. Commun Pure Appl Math 2006;59(8):1207–23.
[17] Mallat SG, Zhang Z. Matching pursuits with time-frequency dictionaries. IEEE Trans Signal Process 1993;41(12):3397–415.
[18] Tropp JA, Gilbert AC. Signal recovery from random measurements via orthogonal matching pursuit. IEEE Trans Inf Theory 2007;53(12):4655–66.
[19] Needell D, Vershynin R. Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit. Found Comut Math 2009;9(3):317–34.
[20] Baraniuk RG, Cevher V, Duarte MF, Hegde C. Model-based compressive sensing. IEEE Trans Inf Theory 2010;56(4):1982–2001.
[21] Blumensath T, Davies ME. Stagewise weak gradient pursuits. IEEE Trans Signal Process 2009;57(11):4333–46.
[22] Donoho DL, Tsaig Y, Drori I, Starck J-L. Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit. IEEE Trans Inf Theory 2012;58(2):1094–121.
[23] La C, Do MN. Tree-based orthogonal matching pursuit algorithm for signal reconstruction. In: IEEE international conference on image processing. p. 1277–80.
[24] Baraniuk RG, Cevher V, Duarte MF, Hegde C. Model-based compressive sensing. IEEE Trans Inf Theory 2010;56(4):1982–2001.
[25] Bui H, La C, Do M. A fast tree-based algorithm for compressed sensing with sparse-tree prior. Elsevier Signal Process 2015;108:628–41.
[26] Kwon S, Wang J, Shim B. Multipath matching pursuit. IEEE Trans Inf Theory 2014;60(5):2986–3001.
[27] Huang G, Wang L. High-speed signal reconstruction for compressive sensing applications. J Signal Process Syst 2014:1–12.
[28] Blumensath T, Davies ME. Iterative thresholding for sparse approximations. J Fourier Anal Appl 2008;14(5–6):629–54.
[29] Blumensath T, Davies ME. Iterative hard thresholding for compressed sensing. Appl Comput Harmon Anal 2009;27(3):265–74.