Sparse Signal Recovery via ECME Thresholding Pursuits

Heping Song¹ and Guoli Wang*¹

¹School of Information Science and Technology, Sun Yat-Sen University, Guangzhou 510006, P. R. China

Email: Heping Song - hepingsong@gmail.com; Guoli Wang* - isswl@mail.sysu.edu.cn;

*Corresponding author

Abstract

Recent years have witnessed increasing interest in sparse signal processing. The emerging theory of compressive sensing provides a new sparse signal processing paradigm for reconstructing sparse signals from the undersampled linear measurements. In this paper, we developed two approaches for finding sparse solutions to underdetermined inverse problem in compressed sensing. The described algorithms, named ECME thresholding pursuits (EMTP), introduced two greedy strategies that each iteration detects a support set \( I \) by thresholding the result of the ECME iteration and estimates the reconstructed signal by solving a truncated least-squares problem on the support set \( I \). Two effective support detection strategies (hard thresholding and dynamic thresholding) are devised for the sparse signals with components having a fast decaying distribution of nonzero components. The hard thresholding iteratively maintains \( k \) largest nonzeros in magnitude of the ECME iteration. The dynamic thresholding iteratively selects nonzero elements greater than an exponentially decreasing threshold. The experimental studies are presented to demonstrate that EMTP offers an attractive alternative to state-of-the-art algorithms for sparse signal recovery.

1 Introduction

Sparsity-exploiting has recently received a great amount of attention in the applied mathematics and signal processing community. Sparse signal processing algorithms have been developed for various applications. A recent Proceedings of the IEEE special issue on applications of sparse representation & compressive sensing devoted to this topic. Some of the exciting developments addressed in [1–7]. Compressed sensing, also
known as compressive sensing, compressive sampling (CS) [8,9] has been the subject of significant activity in sparse signal processing where one seeks to recover efficiently a sparse unknown signal vector of dimension $n$ via a much smaller number ($m$) of undersampled linear measurements. For $k$-sparse unknown signal $x_0 \in \mathbb{R}^n$, the sparse signal recovery is intimately related to solving a underdetermined system of linear equations $y = Ax$ with sparseness constraint

$$\left( P_{\ell_0} \right) : \min_{x_0} \|x_0\|_{\ell_0} \quad s.t. \quad Ax = y, \quad (1)$$

where $x_0$ is true signal to be recovered, $\|x_0\|_{\ell_0}$ is used to denote the number of nonzero components of $x_0$, $A \in \mathbb{R}^{m \times n}$ is the measurement matrix and $y \in \mathbb{R}^m$ is the measurement vector. The key insight in the pioneering work on CS [8,9] is to replace $\ell_0$ by $\ell_1$ for the $(P_{\ell_0})$ problem due to non-convexity and combinatorial effect. In [10], it is the basis pursuit problem

$$\left( \text{BP} \right) : \min_{x_0} \|x_0\|_{\ell_1} \quad s.t. \quad Ax = y. \quad (2)$$

Hereafter, numerous researchers developed various computational algorithms for sparse signal recovery [11,12]. Generally, there are three major classes of computational algorithms: convex relaxation, bayesian inference and greedy pursuit. Convex relaxation replaces the combinatorial problem $(P_{\ell_0})$ with a convex optimization problem (BP), such as basis pursuit [10], basis pursuit denoising [13], the least absolute shrinkage and selection operator (LASSO) [14] and least angle regression (LARS) [15]. Bayesian inference derives the maximum a posteriori estimator from a sparse prior model, such as sparse bayesian learning (SBL) [16,17] and bayesian compressive sensing (BCS) [18,19]. Another popular approach is to use greedy algorithms which iteratively refine a sparse solution by successively selecting one or more elements. These algorithms include matching pursuit (MP) [20], orthogonal matching pursuit (OMP) [21–23], subspace pursuit [24], compressive sampling matching pursuit (CoSaMP) [25] and iterative thresholding algorithms [26,27].

Iterative hard thresholding (IHT) [27] is a simple and powerful approach for sparse recovery. Recently, an alternative algorithm has been present to alleviate the convergence speed issue of IHT in [28,29]. Qiu and Dogandzic [28,29] derived an expectation-conditional maximization either (ECME) iteration from a probabilistic framework based on the signals distributed Gaussian with unknown variance, and proposed an acceleration method, termed double overrelaxation (DOR) thresholding scheme, to improve the convergence speed of the ECME algorithm. In addition, Qiu and Dogandzic [28,29] further present an automatic double overrelaxation (ADORE) thresholding method for conditions that the underlying sparsity
level is not available. For another, Wang and Yin [30] presented an iterative support detection (ISD) algorithm to refine the failed reconstructions by thresholding the solution of a truncated (BP) problem. Inspired by the theoretical and empirical evidence of good recovery performance of ECME [29] and ISD [30], we combine ECME [29] and ISD [30] to devise novel sparse signal recovery methods, dubbed as ECME thresholding pursuits (EMTP). EMTP detects a support set $I$ using a ECME iteration and estimates the reconstructed signal by solving a truncated least-squares problem on the support set $I$, and it iterates these two steps for a small number of times. We present two effective support detection strategies (hard thresholding and dynamic thresholding) for the sparse signals with components having a fast decaying distribution of nonzeros (called fast decaying signals in [30]), which include sparse Gaussian signals, sparse Laplacian signals and certain power-law decaying signals.

This paper considers the iterative greedy algorithms, and abstracts them into two types [31], One Stage Thresholding (OST) and Two Stage Thresholding (TST), as discussed in section 2 and 3. In section 4 we describe the proposed approaches. After that, section 5 details the experimental results. Finally, we conclude this paper in section 6.

**Notation**

We introduce the notation used in this paper:

- $x^{(t)}$: the algorithms described in this paper are iterative and the reconstructed signal $x$ in current iteration $t$ is denoted as $x^{(t)}$. The same convention is used for other vectors and matrices.

- $I, A_I$: index set $I$, the matrix $A_I$ denotes the submatrix of $A$ containing only those columns of $A$ with indexes in $I$. The same convention is used for vectors.

- $[1, n] \setminus I$: the complement of set $I$ in set $\{1, 2, \ldots, n\}$.

- supp($x$): the support set of a vector $x$, i.e. the index set corresponding to the nonzeros of $x$,
  \[ \text{supp}(x) = \{i : x_i \neq 0\} \]

- $H_k(x)$: the hard thresholding that sets all but the largest in magnitude $k$ elements of a vector $x$ to zero.

- $|x|, \|x\|_{\ell_p}, x^T$: the absolute value, $\ell_p$ norm and transpose of a vector $x$, respectively.

- $A^+$: the Moore-Penrose pseudoinverse of matrix $A \in \mathbb{R}^{m \times n}$. $A^+ = A^T (AA^T)^{-1}$ for $m \leq n$; $A^+ = (A^T A)^{-1} A^T$ for $m \geq n$. 


2 One Stage Thresholding

Qiu and Dogandzic derived an expectation-conditional maximization either (ECME) iteration based on a probabilistic framework [29],

\[
x^{(t)} = H_k(x^{(t-1)} + A^T(AA^T)^{-1}(y - Ax^{(t-1)})),
\]

(3)

Note that ECME iteration reduces to IHT step when the measurement matrix \(A\) has orthonormal rows (that is, \((AA^T)\) is the identity matrix). These one stage thresholding algorithms (e.g. IHT [27] and ECME [29]) are guaranteed to recover sparse signals and converge with limited iterations. However, OST is not the empirical choice for practical applications due to slow convergence. To this end, Qiu and Dogandzic proposed an acceleration method, termed double overrelaxation (DORE) thresholding scheme [28, 29], to improve the convergence speed of the ECME algorithm. DORE utilizes two overrelaxation steps

\[
x_1^{(t)} = x^{(t)} + \alpha_1(x^{(t)} - x^{(t-1)}),
\]

(4)

and

\[
x_2^{(t)} = x_1^{(t)} + \alpha_2(x_1^{(t)} - x^{(t-2)}),
\]

(5)

where \(\alpha_1, \alpha_2\) are the line search parameters. Finally, an additional hard thresholding step

\[
x^{(t)} = H_k(x_2^{(t)}),
\]

(6)

ensure that the resulting signal is guaranteed to be \(k\)-sparse. In addition, Qiu and Dogandzic further present an automatic double overrelaxation (ADORE) thresholding method for conditions that the underlying sparsity level is not available.

3 Two Stage Thresholding

The algorithms described in this paper fall into the category of a general family of iterative greedy pursuit algorithms. Following [31], we adopt the name “Two Stage Thresholding” (TST). Consider a CS recovery problem, the sparse recovery algorithms aim to detect the support set \(I\) and approximate \(y\) using

\[
y = A_I x_I.
\]

(7)

Starting with initial solution \(x = 0\), TST iterates between 2 main steps:
Step 1: Support detection: detect the support set $I$ of the signal $x$, i.e. select atoms of measurement matrix $A$ which have been used to generate $y$, in other words, determine active atoms in sparse representation of a signal $x$. In some literatures, this step also is called basis selection or atom selection.

Step 2: Signal estimation: update the signal $x$ using the least-squares solution on the detected support set $I$. $x_I = \arg \min_z \{ \| y - A_I z \|_2^2, \supp(z) \subseteq I \}$,

Many algorithms (e.g. Orthogonal Matching Pursuit (OMP) [23], Subapace Pursuit [24], Compressed Sensing Matching Pursuit (CoSaMP) [25] and Gradient Pursuits (GP) [32]) developed various approaches for Step 1 or Step 2.

4 ECME Thresholding Pursuits

In this section, we describe our proposed approaches, named ECME Thresholding Pursuits (EMTP), that combine OST and TST. The detailed description of the proposed algorithms are presented as follows

Step 1: Initialization:
- Initialize the reconstruction signal $x^{(0)} = 0$,
- initialize residual signal $r^{(0)} = y$,
- initialize support set $I^{(0)} = \emptyset$,
- and set the iteration counter $t = 1$.

Step 2: Support detection:
- Update signal approximation: $x^{(t)} = x^{(t-1)} + A^\dagger r^{(t-1)}$.
- Detect the support set $I$:
  - strategy 1: hard thresholding $I^{(t)} = \supp(H_k(x^{(t)}))$;
  - strategy 2: dynamic thresholding $I^{(t)} = \{ i : |x_i^{(t)}| > \beta \max_i |x_i^{(t)}| \}$.

Step 3: Signal estimation:
- Estimate the signal: $x_I^{(t)} = A_i^{\dagger} y$,
- $x_{[1,n] \setminus I^{(t)}} = 0$.
- Update the residual:
\[ r^{(t)} = y - Ax^{(t)}. \]

**Step 4: Halting:**

Check the stopping condition and, if it is not yet time to stop, increment \( t = t + 1 \) and return to step 2. If it is time to stop, the recovered signal \( x \) has nonzero entries in support set \( I^{(t)} \) and corresponding support vector lies in \( x_{I^{(t)}}^{(t)} \).

We present two thresholding proposals, and corresponding algorithm \( k \)-EMTP (shown as Algorithm 1) and \( \beta \)-EMTP (shown as Algorithm 2), where \( \beta \in (0, 1) \) is the thresholding parameter.

**Remarks:**

1. EMTP updates the leading elements of \( x^{(t)} \) using a least-squares solution on the support set \( I \). However, DORE updates all the elements of \( x^{(t)} \) using double overrelaxation steps. Finally, DORE needs a hard thresholding to ensure the new approximation is \( k \)-sparse. \( \beta \)-EMTP demands no prior knowledge about the underlying sparsity level \( k \).

2. EMTP is a special case of TST and utilizes OST in the support detection stage. It is different from OMP [23], Subapace Pursuit [24] and CoSaMP [25]. OMP, Subapace Pursuit and CoSaMP use the correlation between the residual signal and the atoms of the measurement matrix \( A \) to detect support set.

3. Like other greedy pursuits such as Subapace Pursuit [24] and CoSaMP [25], \( k \)-EMTP fixes the cardinality of support set \( I \) and removes previous false detections. However, \( \beta \)-EMTP refines the support set \( I \) which is increasing and nested over the iterations.

4. The dynamic thresholding strategy used in \( \beta \)-EMTP is inspired by ISD [30]. It finds significant nonzeros by comparing a threshold rather than maintaining fixed number \((k)\) of items. It is appealing for the conditions that the underlying sparsity level \( k \) is not available.

5. EMTP resembles ISD [30]. EMTP and ISD have the same idea in support detection step with iterative behavior, however, the support detection step bases on different sparse recovery methods, ECME and BP respectively. EMTP updates the reconstruction signal using a least-squares solution on the detected support set \( I \). However, ISD iteratively refines the BP solution on the complement of the detected support set \( I \).
6. β-EMTP with large β obtains high quality reconstruction from a small number of measurements. However, because support set \( I \) grows slowly, β-EMTP takes a larger number of iterations (to be discussed in Section 5). \( k \)-EMTP wrongly detected elements can often be pruned out in later iterations.

7. Like ISD (as discussed in [30]), β-EMTP only performs well for fast decaying signals. It does not work on sparse signals that decay slowly or have no decay at all (e.g. trinary and binary sparse signals). \( k \)-EMTP performs worse for the sparse signals that nonzero elements have similar magnitude [31, 33]. However, we can apply EMTP to non fast decaying signals via linear or non-linear pre-mapping [34, 35].

**Algorithm 1 k-EMTP Algorithm**

**Input:** Measurement matrix \( A \), measurements \( y \), sparsity level \( k \)

**Output:** The reconstructed signal \( x \)

1. **Initialization:**
2. \( t = 1 \)  //iteration number
3. \( x^{(0)} = 0 \)  //initial signal
4. \( r^{(0)} = y \)  //initial residual
5. \( I^{(0)} = \emptyset \)  //initial support set
6. **while** halting criterion **false** **do**
7. \( x^{(t)} = H_k(x^{(t-1)} + A^\dagger r^{(t-1)}) \)
8. \( I^{(t)} = \{i : x^{(t)}_i \neq 0\} \)
9. \( x^{(t)}_{I^{(t)}} = A^\dagger y_{I^{(t)}} \)
10. \( x^{(t)}_{[1:n] \setminus I^{(t)}} = 0 \)
11. \( r^{(t)} = y - Ax^{(t)} \)
12. \( t = t + 1 \)
13. **end while**
14. **return** \( x \)
Algorithm 2 β-EMTP Algorithm

Input: Measurement matrix $A$, measurements $y$, thresholding parameter $\beta$
Output: The reconstructed signal $x$

1: Initialization:
2: $t = 1$ //iteration number
3: $x(0) = 0$ //initial signal
4: $r(0) = y$ //initial residual
5: $I(0) = \emptyset$ //initial support set
6: while halting criterion false do
7: $x(t) = x(t-1) + A^\dagger r(t-1)$
8: $I(t) = \{ i : |x_i(t)| > \beta^t \max_i |x_i(t)| \}$
9: $x_{I(t)} = A^\dagger y$
10: $x_{[1:n]\setminus I(t)} = 0$
11: $r(t) = y - Ax(t)$
12: $t = t + 1$
13: end while
14: return $x$

5 Experimental Results

To assess the performance of the proposed approaches, we conduct numerical experiments on computer simulations. All algorithms were implemented and tested in MATLAB v7.6 running on Windows XP with 2.53GHz Intel Celeron CPU and 2GB of memory. We compared the proposed approaches with the accelerated ADORE/DORE approaches. The code of ADORE/DORE is available on the authors homepage. We initialized $x$ with zero sparse signal. The search length of ADORE was set to 1. All results were averaged over 100 times Monte Carlo problem instances. We used our unoptimized code. The least-squares solution $x = \arg\min \|y - Ax\|^2_2$ was implemented using MATLAB pseudoinverse function by $x = \text{pinv}(A)\times y$. The MATLAB code was partially adapted from [30,33,36]. In all the experiments, the measurement matrix $A$ generated by uniform spherical ensemble (USE), i.e. we generate the measurement matrix by sampling each entry independently form the standard norm distribution and then normalize each column to have unit norm. The MATLAB code is given by

```matlab
A = randn(m, n); A = A ./ repmat(sqrt(sum(A.^2)), [m 1]);
```

The underlying $k$-sparse vectors were generated by randomly selecting a support set of size $k$ and each entry in the support set is sampled uniformly from a specific distribution. The sparse signals were generated in MATLAB by

```matlab
x = zeros(n,1); support = randperm(n); x(support(1:k)) = v;
```
and the nonzeros $v$ generated by following code.

The sparse Gaussian signals were generated in MATLAB by

$$v = \text{randn}(k, 1);$$

The sparse Laplacian signals were generated by

$$z = \text{rand}([k, 1]);$$
$$v = \text{zeros}([k, 1]);$$
$$\text{in} = z \leq 0.5;$$
$$\text{ip} = z > 0.5;$$
$$v(\text{in}) = \frac{1}{\lambda} \log(2 * z(\text{in}));$$
$$v(\text{ip}) = -\frac{1}{\lambda} \log(2 * (1-z(\text{ip})));$$

and the power-law decaying signals were generated by

$$v = \text{sign} (\text{rand}(k, 1)) .* ((1:k)^{-1/\lambda})';$$

The variable $\lambda$ controls the rate of decay. We set $\lambda=10$ and $\lambda=0.5$ for sparse Laplacian signals and power-law decaying signals, respectively.

For fair comparison, we stopped iterations once the relative error fell below a certain convergence tolerance or the number of iterations is greater than 100. The convergence tolerance is given by

$$\frac{\|y - r(t)\|_2}{\|y\|_2} \leq 10^{-6}. \quad (8)$$

We empirically evaluate reconstruction performance in terms of signal-to-noise ratio (SNR) and number of iterations. The SNR is defined as

$$\text{SNR (dB)} = -20 \log_{10} \frac{\|x_0 - x\|_2}{\|x_0\|_2}, \quad (9)$$

where $x_0$ is the true signal, $x$ is the recovered signal.

5.1 Influence Of Thresholding Parameter

We evaluated the influence of thresholding parameter $\beta$ by varying $\beta$ from 0.4 to 0.7. First, we fixed signal dimension $n = 400$, the underlying sparsity level $k = 20, 40, 80$ and varied the number of measurements.

The plots of influence of thresholding parameter $\beta$ for sparse Gaussian signals are presented in Figure 1, 2 and 3, respectively. It is clear that larger thresholding parameter achieves better recovery performance.
with fewer measurements. However, the cost is more iterations. Large $\beta$ is particularly competitive when the number of measurements is fairly small. It is worth noting that the number of iterations is smaller than the underlying sparsity level $k$ for exact recovery, especially for large $k$. As shown in Figure 1, 2 and 3, $\beta$-EMTP achieves the SNR (dB) around 300, that is, the relative error almost as low as the double precision. Second, we tested the comparisons by fixing the number of measurements $m = 80, 140, 200$, as shown in Figure 4, 5 and 6, respectively. Finally, we enlarged the dimension of signals ($n = 2000$), as shown in Figure 7 and 8. The latter tests come to similar conclusions as the first test. We used $\beta$-EMTP with $\beta = 0.5$ for later experiments.

5.2 Comparisons With The Accelerated OST
We present comparisons in terms of SNR (dB) and number of iterations. We omit the comparisons with $\ell_1$ minimization methods and some other OST/TST methods, as they have already been investigated in [31, 33]. First, we fixed signal dimension $n = 400$, the underlying sparsity level $k = 20, 40, 80$ and varied the number of measurements. The results were plotted in Figure 9, 10 and 11, respectively. Next, we tested the comparisons by fixing the number of measurements $m = 80, 140, 200$, as shown in Figure 12, 13 and 14, respectively. Finally, the test set used larger dimension signals ($n = 2000$). The corresponding results were depicted in Figure 15 and 16. Figure 9, 10 and 11 show the average SNR (dB) (top panel) and number of iterations (bottom panel) for each indeterminacy level $m/n$. EMTP achieves significantly larger SNR (dB) than ADORE/DORE. For exact recovery, EMTP obtains the relative error almost as low as the double precision. $\beta$-EMTP constantly needs small number of iterations and $k$-EMTP needs smallest number of iterations for exact recovery. However, ADORE needs hundreds of iterations. Figure 9 was zoomed in for better illustration. Figure 12, 13 and 14 show the average SNR (dB) and number of iterations for each sparsity level $k/m$. For larger dimension signals ($n = 2000$), the results depicted in Figure 15 and 16. The latter tests come to similar conclusions as the first test. These figures do not fully capture the performance of $\beta$-EMTP since we only set $\beta = 0.5$, however $\beta$-EMTP achieved almost superior performance than ADORE/DORE.

5.3 Performance For Other Fast-Decaying Signals
As discussed in [30, 33], thresholding-based sparse recovery methods achieve superior performance only if the nonzero elements of signals have a fast-decaying distribution. We present empirical studies on other fast-decaying signals. Sparse Laplacian signals containing $k = 40$ nonzero elements for dimension $n = 400$
were tested and corresponding result was plotted in Figure 17. For larger dimension signals \((n = 2000)\), the result depicted in Figure 18. Pow-law decaying signals containing \(k = 40\) nonzero elements for dimension \(n = 400\) were tested and corresponding result was plotted in Figure 19. For larger dimension signals \((n = 2000)\), the result depicted in Figure 20. ADORE/DORE were derived from a probabilistic framework based on Gaussian distribution. Surprisingly, they can also work for fast-decaying signals, as depicted in Figure 17, 18, 19 and 20. The conclusions for sparse Gaussian signals also can be generalized for sparse Laplacian signals and pow-law decaying signals. Pow-law decaying signals for low indeterminacy level \(m/n\) make an exception that \(\beta\)-EMTP achieves inferior performance.

5.4 Summary
To sum up, we have compared EMTP with ADORE/DORE. EMTP has significant advantages over the accelerated OST in terms of SNR and number of iterations. EMTP can significantly reduce the number of iterations required and achieves significantly higher SNR. For low indeterminacy level \(m/n\), EMTP requires fewer measurements. \(\beta\)-EMTP can work with no prior knowledge of the underlying sparsity level \(k\) yet achieves recovery performance better than ADORE. Among all methods, \(\beta\)-EMTP appeared to be the best.

6 Conclusions And Future Work
In this paper, we propose ECME thresholding pursuits (EMTP) for sparse signal recovery. EMTP detects a support set \(I\) using a ECME iteration and estimates the reconstructed signal by solving a truncated least-squares problem on the support set \(I\), and it iterates these two steps for a small number of times. We present two effective support detection strategies (hard thresholding and dynamic thresholding) for the sparse signals with components having a fast decaying distribution of nonzero components. The experimental studies are presented to demonstrate that EMTP offers an attractive alternative to state-of-the-art algorithms for sparse signal recovery. EMTP can significantly reduce the number of iterations required. We then turn to the problem of reducing the computational cost of each iteration, especially for large scale applications. The future research includes three directions:

1. EMTP requires the pre-computation and storage of the pseudoinverse matrix \(A^\dagger\). So we replace ECME iteration by IHT step in the support detection stage will be a advisable choice. IHT do not require the computation and storage of an pseudoinverse matrix \(A^\dagger\), which leads to significant advantage for large scale applications.
2. In signal estimation stage, i.e. update reconstructed signal by solving least-squares problem, EMTP uses the orthogonal projection, that is, by calculating $x_I = A_I^y$ where $A_I$ is the pseudoinverse of $A_I$.

We will approximate the orthogonal projection efficiently by gradient pursuits [32].

3. This paper devotes efforts to devise computational algorithms which are experimentally reproducible and provides empirical studies as useful guidelines for practical applications, further, the theoretical analysis of EMTP will be developed for future investigations.

Acknowledgements

This work was supported by the National Science Foundation of China under grant No. 61074167.
References


Figures

Figure 1 - Influence of thresholding parameter $\beta$ for sparse Gaussian signals with $n = 400$, $k = 20$: comparisons in terms of SNR (dB) and number of iterations.

Figure 2 - Influence of thresholding parameter $\beta$ for sparse Gaussian signals with $n = 400$, $k = 40$: comparisons in terms of SNR (dB) and number of iterations.

Figure 3 - Influence of thresholding parameter $\beta$ for sparse Gaussian signals with $n = 400$, $k = 80$: comparisons in terms of SNR (dB) and number of iterations.

Figure 4 - Influence of thresholding parameter $\beta$ for sparse Gaussian signals with $n = 400$, $m = 80$: comparisons in terms of SNR (dB) and number of iterations.

Figure 5 - Influence of thresholding parameter $\beta$ for sparse Gaussian signals with $n = 400$, $m = 140$: comparisons in terms of SNR (dB) and number of iterations.

Figure 6 - Influence of thresholding parameter $\beta$ for sparse Gaussian signals with $n = 400$, $m = 200$: comparisons in terms of SNR (dB) and number of iterations.

Figure 7 - Influence of thresholding parameter $\beta$ for sparse Gaussian signals with $n = 2000$, $k = 100$: comparisons in terms of SNR (dB) and number of iterations.

Figure 8 - Influence of thresholding parameter $\beta$ for sparse Gaussian signals with $n = 2000$, $m = 400$: comparisons in terms of SNR (dB) and number of iterations.

Figure 9 - Sparse Gaussian signals with $n = 400$, $k = 20$: comparisons in terms of SNR (dB) and number of iterations.

Figure 10 - Sparse Gaussian signals with $n = 400$, $k = 40$: comparisons in terms of SNR (dB) and number of iterations.

Figure 11 - Sparse Gaussian signals with $n = 400$, $k = 80$: comparisons in terms of SNR (dB) and number of iterations.

Figure 12 - Sparse Gaussian signals with $n = 400$, $m = 80$: comparisons in terms of SNR (dB) and number of iterations.

Figure 13 - Sparse Gaussian signals with $n = 400$, $m = 140$: comparisons in terms of SNR (dB) and number of iterations.
**Figure 14** - Sparse Gaussian signals with $n = 400$, $m = 200$: comparisons in terms of SNR (dB) and number of iterations.

**Figure 15** - Sparse Gaussian signals with $n = 2000$, $k = 100$: comparisons in terms of SNR (dB) and number of iterations.

**Figure 16** - Sparse Gaussian signals with $n = 2000$, $m = 400$: comparisons in terms of SNR (dB) and number of iterations.

**Figure 17** - Sparse Laplacian signals with $n = 400$, $k = 40$: comparisons in terms of SNR (dB) and number of iterations.

**Figure 18** - Sparse Laplacian signals with $n = 2000$, $k = 100$: comparisons in terms of SNR (dB) and number of iterations.

**Figure 19** - Pow-law decaying signals with $n = 400$, $k = 40$: comparisons in terms of SNR (dB) and number of iterations.

**Figure 20** - Pow-law decaying signals with $n = 2000$, $k = 100$: comparisons in terms of SNR (dB) and number of iterations.
$n=400, k=20, m=40:10:200$

**SNR (dB):**

- EMTP-$\beta=0.4$
- EMTP-$\beta=0.5$
- EMTP-$\beta=0.6$
- EMTP-$\beta=0.7$

**Number of Iterations:**

- EMTP-$\beta=0.4$
- EMTP-$\beta=0.5$
- EMTP-$\beta=0.6$
- EMTP-$\beta=0.7$
n=400, k=40, m=80:10:240

![Graph showing SNR (dB) vs. m/n for different values of EMTP-β.](image)

- EMTP-β = 0.4
- EMTP-β = 0.5
- EMTP-β = 0.6
- EMTP-β = 0.7

![Graph showing Number of Iterations vs. m/n for different values of EMTP-β.](image)

- EMTP-β = 0.4
- EMTP-β = 0.5
- EMTP-β = 0.6
- EMTP-β = 0.7
n=400, m=80, k=4:2:40

SNR (dB)

EMTP − β=0.4
EMTP − β=0.5
EMTP − β=0.6
EMTP − β=0.7

Number of Iterations

EMTP − β=0.4
EMTP − β=0.5
EMTP − β=0.6
EMTP − β=0.7
n=400, m=140, k=4:4:72

EMTP $\beta=0.4$
EMTP $\beta=0.5$
EMTP $\beta=0.6$
EMTP $\beta=0.7$

Number of Iterations

EMTP $\beta=0.4$
EMTP $\beta=0.5$
EMTP $\beta=0.6$
EMTP $\beta=0.7$
n=400, m=200, k=4:4:100

- EMTP−β=0.4
- EMTP−β=0.5
- EMTP−β=0.6
- EMTP−β=0.7
n=400, k=20, m=40:10:200

SNR (dB)

Number of Iterations

ADORE
DORE
EMTP $- \beta = 0.5$
EMTP $- k$
n=400, m=80, k=4:2:40

SNR (dB)

Number of Iterations

ADORE
DORE
EMTP−β=0.5
EMTP−k
n=400, m=140, k=4:4:72

SNR (dB)

Number of Iterations

- ADORE
- DORE
- EMTP\(−\beta=0.5\)
- EMTP\(−k\)
n=400, k=40, m=80:10:240

SNR (dB)

- ADORE
- DORE
- EMTP
- EMTP-β=0.5
- EMTP-k

Number of Iterations

- ADORE
- DORE
- EMTP
- EMTP-β=0.5
- EMTP-k