Greedy Algorithms CoSaMP and OMP: The Benefits of Sparse in Levels Vectors

May 9th, 2021

Cléa Aumont
MATH 494: Compressed Sensing
Concordia University
40041392

Abstract

Sparse in levels vectors exhibit a local sparsity structure defined by a set of levels - the *sparsity in levels* model has shown important improvements in the greedy signal recovery algorithm Compressive Sampling Matching Pursuit and Orthogonal Matching Pursuit. This will be shown by conducting a series of numerical experiments with CoSaMP and OMP using a standard sparsity model and by comparing their performance when using a structured in levels sparsity model. CoSaMP's Recovery Guarantees for stability and robustness will be mentioned, as well as their extension to the in levels case.

1 Introduction

In the setting of *compressed sensing*, one attempts to recover a signal or vector x from a minimal number of linear measurements y. This implies finding the solution to an underdetermined linear system using various reconstruction algorithms such as optimization methods, greedy methods, and thresholding-based methods. One of the conditions under which signal recovery is possible is that x must have a sparse representation in some domain such as a Fourier or wavelet domain. In other words, we are trying to recover x from $Ax + e = y$ where $A \in \mathbb{C}^{m \times N}$, $m < N$, $x \in \mathbb{C}^N$ and the noise level $e \in \mathbb{C}^m$. Exploiting the sparsity of x becomes crucial in finding a unique solution to this system.

This paper discusses two greedy algorithms for signal recovery, namely Compressive Sampling Matching Pursuit (CoSaMP) and Orthogonal Matching Pursuit (OMP), and their overall efficiency when using a *structured sparsity* as opposed to a global sparsity. Although there exists a number of types of structured sparsity models such as block sparsity or tree sparsity, the focus of this paper will be the in levels sparsity model. This type of structure has shown important computational improvements in signal recovery over the standard sparsity model. This report is based on two main academic papers, specifically [1] and [2].

1.1 Outline

This paper will first introduce the algorithms CoSaMP and OMP in the sparse case only. In the third section, the in levels model will be presented, along with the CoSaMPL and OMPL algorithms using the in levels structured sparsity. The fourth section will contain numerical results that highlight the difference between the two models used in each algorithm. The last section will mention some recovery guarantees for CoSaMP only and how they translate to the in levels case with CoSaMPL.

2 Greedy Methods

In this section, the greedy algorithms OMP and CoSaMP are presented (their extensions to the levels case will be described in the next section). Orthogonal matching pursuit was first introduced in [3], as a modification of Matching Pursuit (MP). It improved convergence and computational efficiency using an additional orthogonalization step. The main improvement over MP is that OMP terminates in a finite number of steps, generally the sparsity level s of the vector x, so fewer iterations are needed, and the algorithm is guaranteed to converge [3]. OMP is defined by

Function: $x^\sharp = \mathrm{OMP}(A, y, s)$

Inputs: $A \in \mathbb{C}^{m\times N}$, $y \in \mathbb{C}^m$, $x^{(0)} = 0$, s global sparsity

Iterate: from $k = 1 : s$

$$j_k \in \underset{j=1\ldots N}{\operatorname{argmax}} |(A^*(y - Ax^{(k-1)}))_j|$$
$$S^{(k)} = S^{(k-1)} \cup \{j_k\}$$
$$x^{(k)} \in \operatorname{argmin}\{\|y - Az\|_2 \ : \quad z \in \mathbb{C}^N, \operatorname{supp}(z) \subseteq S^{(k)}\}$$

Output: $x^\sharp = x^{(s)}$

The nature of greedy algorithms is to make local optimal choices at every iteration, without necessarily providing a global optimal solution, but a reasonable approximation of one. In the case of OMP, the support of the sparse vector $x^{(n)}$ is approximated by adding one index to its running support at each iteration by making a greedy index selection. Because the linear measurements y are made up of a combination of s columns of the sensing matrix A, the algorithm chooses the column a_j that is the most correlated with the residual $(y - Ax)$ [4], adding the index j to the running support of $x^{(n)}$. This algorithm terminates after s iterations, and is computationally efficient for a small target sparsity, as the least square step will never exceed the dimensions $m \times s$. In fact this algorithm runs in $\mathcal{O}(smN)$ time [2].

Compressive Sampling Matching Pursuit was first proposed in 2008 in [2]. This algorithm addresses some of the weaknesses of OMP by accelerating the algorithm and providing strong recovery guarantees [2]. Before introducing the algorithm, the hard thresholding operator $H_s(x)$ must first be defined. Let the operator $L_s(x)$ denote the s largest absolute entries of the vector x. Now, $H_s(x)$ is defined as

$$H_s(x) = (H_s(x)_i)_{i=1}^N \quad H_s(x)_i = \begin{cases} x_i & i \in L_s(x) \\ 0 & \text{otherwise} \end{cases}$$

The thresholding operator $H_s(x)$ is used to enforce the sparsity s in the algorithm, and $L_s(x)$ is simply the support of $H_s(x)$.

The algorithm for CoSaMP is

Function: $x^\sharp = \text{CoSaMP}(A, y, s)$

Inputs: $A \in \mathbb{C}^{m\times N}$, $y \in \mathbb{C}^m$, $x^{(0)} = 0$, s global sparsity

Iterate: until a given error, or max number of iterations reached

$$U^{(n+1)} = \text{supp}(x^{(n)}) \cup L_{2s}(A^*(y - Ax^{(n)}))$$
$$u^{(n+1)} = \underset{z\in\mathbb{C}^N}{\text{argmin}}\{\|y - Az\|_2 : \text{supp}(z) \subseteq U^{(n+1)}\}$$
$$x^{(n+1)} = H_s(u^{(n+1)})$$

Output: $x^\sharp = x^{(n)}$

In OMP, only one index is added to the running support of $x^{(n)}$ at every iteration, whereas in CoSaMP, a full estimation of the support of $x^{(n)}$ is provided after one iteration, and it is modified and improved in subsequent iterations. In OMP, if an incorrect index is selected, it will remain in all iterations, as only s iterations are performed. CoSaMP addresses this issue by choosing more than one possible index at each iteration. In fact, it selects the $2s$ largest entries between A and the residual, then performs a least square minimization, and uses a thresholding operator to enforce this sparsity s. This makes CoSaMP much faster than OMP: it runs in $\mathcal{O}(mN)$ time instead of $\mathcal{O}(smN)$ [2].

3 Structured Sparsity

Before moving to the generalization of these algorithms, the sparse in levels setting must first be introduced. Instead of looking at a vector with a single sparsity s and arbitrary support, we consider a levels based structure for this sparsity. This sparse vector is said to exhibit a local sparsity pattern specified by the vector $\mathbf{s} = (s_1, s_2...s_r)$, with r levels.

Definition 3.1 *[1, Definition 3.1] Let $r \geq 1$, $M = (M_1, M_2...M_r)$, where $1 \leq M_1 < M_2 < ... < M_r = N$ and $s = (s_1, s_2...s_r)$, where $s_k \leq M_k - M_{k-1}$ for $k = 1, ...r$ with $M_0 = 0$. A vector $x = (x_i)_{i=1}^N \in \mathbb{C}^N$ is (s, M)-sparse if*

$$|supp(x) \cap \{M_{k-1} + 1, ...M_k\}| \leq s_k, \quad k = 1, ...r$$

The set of all (s, M)-sparse vectors is denoted as $\Sigma_{s,M} \subseteq \mathbb{C}^N$.

This definition means that we have a set of r levels; the partitioning of each level is stored in M, where the last partition must be equal to N; the local sparsities of each level is listed in $\mathbf{s}$; and the sum of $\mathbf{s}$ is the global sparsity. This generalizes the sparsity of a vector to an in levels model, where we can have the specific case of one level, where $M = (N)$ and $\mathbf{s} = (s_1)$, which is simply the single sparsity s of a vector. This generalization to an in levels sparsity model has shown important benefits in signal recovery in terms of speed and probability of convergence.

3.1 Motivations

This idea of sparse in level vectors was inspired by wavelets, specifically wavelet scales [5, Chapter 11]. When, for example, a Discrete Wavelet transform is applied to signal x using Haar wavelets, the Haar wavelet coefficients of x will exhibit a local sparsity pattern. The levels are determined by the wavelet scales, where each level increases exponentially in size by a factor 2^n. The local sparsities, s_k, are the Haar coefficients, and are the most concentrated in the first level, also the smallest level. As the levels get increasingly large they also get increasingly sparse - this is referred to as asymptotic sparsity, where asymptotic decay is observed in each level [5]. If the goal is to recover a sparse vector x from a number of linear measurements, it becomes simpler to infer where the sparse coefficients must be now that we have an idea of the underlying structure of the vector we are trying to recover.

3.2 Extending OMP & CoSaMP to the levels case

Extending CoSaMP to the in levels case only changes the algorithm in the thresholding operators $H_s(x)$ and $L_s(x)$. For a set of levels $M = (M_1, ...M_r)$ and local sparsities $\mathbf{s} = (s_1...s_r)$, any vector $x \in \mathbb{C}^N$ can be uniquely expressed as $x = \sum_{k=1}^r x_k$, where $x_k \in \mathbb{C}^N$, with $\text{supp}(x_k) \subseteq \{M_{k-1}+1, ..., M_k\}$

[1]. Now, $L_{s,M}(x)$ can be defined as

$$L_{s,M}(x) = \bigcup_{k=1}^{r} L_{s_k}(x_k)$$

This is the union of the s_k largest absolute entries of each level $\{M_{k-1}+1, ..., M_k\}$. The hard thresholding operator can now be defined as

$$H_{s,M}(x) = (H_{s,M}(x)_i)_{i=1}^{N} \quad H_{s,M}(x)_i = \begin{cases} x_i & i \in L_{s,M}(x) \\ 0 & \text{otherwise} \end{cases}$$

This operator retains the largest s_k entries of each level, and sets everything else in that level to 0. The algorithm CoSaMPL is defined as follows

Function: $x^\sharp = \text{CoSaMPL}(A, y, s, M)$

Inputs: $A \in \mathbb{C}^{m\times N}$, $y \in \mathbb{C}^m$, $x^{(0)} = 0$, s set of local sparsities, M set of levels

Iterate: until a given error, or max number of iterations reached

$$\begin{aligned} U^{(n+1)} &= \text{supp}(x^{(n)}) \cup L_{2s,M}(A^*(y - Ax^{(n)})) \\ u^{(n+1)} &= \underset{z\in\mathbb{C}^N}{\text{argmin}}\{\|y - Az\|_2 : \text{supp}(z) \subseteq U^{(n+1)}\} \\ x^{(n+1)} &= H_{s,M}(u^{(n+1)}) \end{aligned}$$

Output: $x^\sharp = x^{(n)}$

A generalization of OMP to the in levels setting is given by

Function: $x^\sharp = \text{OMPL}(A, y, s, M)$

Inputs: $A \in \mathbb{C}^{m\times N}$, $y \in \mathbb{C}^m$, $x^{(0)} = 0$, s global sparsity

Iterate: from $k = 1 : s$

$$\begin{aligned} j_k &\in \underset{\substack{j=1\dots N \\ j\notin M_i, \forall s_i = 0}}{\text{argmax}} |(A^*(y - Ax^{(k-1)}))_j| \\ S^{(k)} &= S^{(k-1)} \cup \{j_k\} \\ x^{(k)} &\in \text{argmin}\{\|y - Az\|_2 \ : \quad z \in \mathbb{C}^N, \text{supp}(z) \subseteq S^{(k)}\} \end{aligned}$$

Update $\mathbf{s}$, decrement the corresponding level in which j was found

Output: $x^\sharp = x^{(s)}$

To generalize OMP to the in levels setting, the index j_k to be chosen has an additional constraint: it is only allowed to be selected if its current level $\{M_{k-1}+1, ..., M_k\}$ does not have a corresponding sparsity $s_k = 0$. If it does, then no index can be selected from that particular level, and the next largest index in absolute value is selected. After one iteration of the algorithm, the vector $\mathbf{s}$ of local sparsities is updated, with a decrement in the s_k of the kth level. This algorithm performs s iterations or terminates when the vector $\mathbf{s}$ contains only zeros.

4 Numerical Experiment

To illustrate the performance of each algorithm, two experiments are performed using the same settings. The sensing matrix A is a *Gaussian random matrix* with normalized columns and independent and identically distributed entries with mean 0 and variance $1/\sqrt{m}$. A is an $m \times N$ matrix where $N = 256$, and the linear measurements m vary, and the sparsity s of the vector x varies as well. For the in levels model, two cases are considered. The first case with 2 levels with $M = [N/2, N]$ exhibiting the sparsity pattern $\mathbf{s} = [3s/4, s/4]$. The second case has 4 levels with $M = [N/4, N/2, 3N/4, N]$ exhibiting the sparsity pattern $\mathbf{s} = [s/2, 0, s/2, 0]$.

4.1 Experiment One: 3 fixed sparsities and varying linear measurements

This first experiment allows m to vary from $m = 1$ to 256, and has three fixed sparsities: $s = 16$, $s = 32$, and $s = 60$. For each sparsity, two trials are performed. The first trial uses standard OMP, and OMPL with 2 and 4 levels. The second trial uses standard CoSaMP, and CoSaMPL with 2 and 4 levels. The relative error $\|x - x^\sharp\|_2/\|x\|_2$ is computed for each trial. If the relative error is less than 10^{-4}, the recovery is deemed successful, or the algorithm stops computing after 120 iterations. The success probability is defined by the average number of successful recoveries over 50 runs, and is plotted in the horizontal phase transition diagrams below.

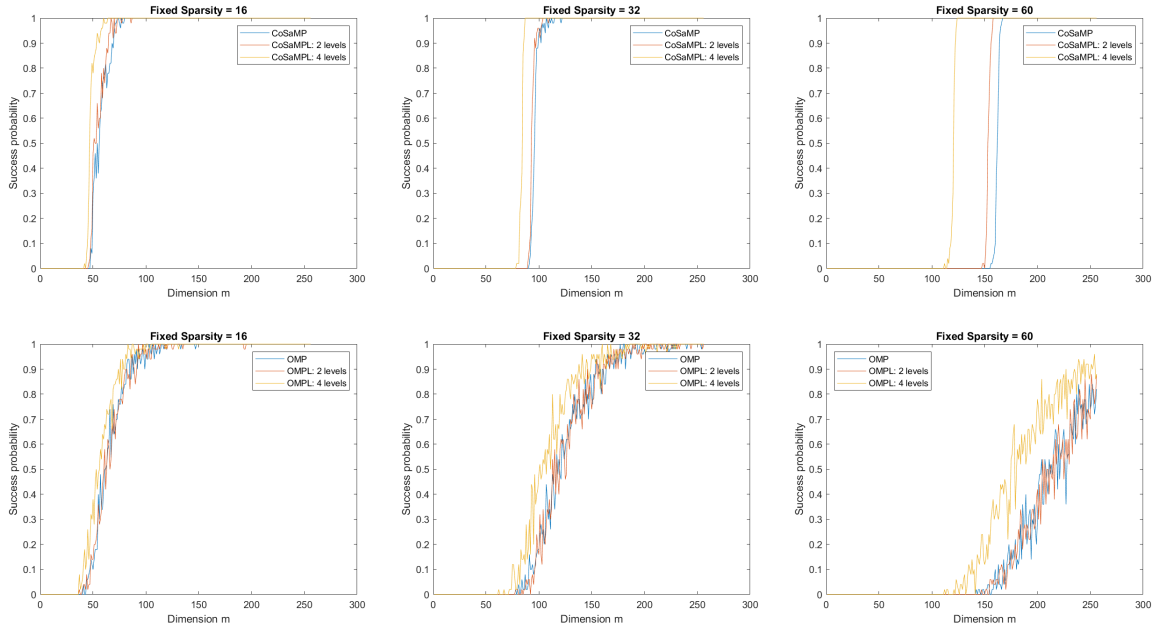

Figure 1: *Horizontal phase transition plots with success probability versus number of linear measurements m for OMP, OMPL, CoSaMP and CoSaMPL. The 3 fixed sparsities are $s = 16$, $s = 32$, and $s = 60$. The 2 level sparse vector has $M = [N/2, N]$ and $s = [3s/4, s/4]$. The 4 level sparse vector has $M = [N/4, N/2, 3N/4, N]$ and $s = [s/2, 0, s/2, 0]$.*

4.2 Experiment Two: varying sparsity and varying linear measurements

For the second experiment, m varies from $m = 20$ to 240 in increments of 10, and the sparsity varies from $s = 4$ to 60 in increments of 4. This experiment uses the same sparsity structure as Experiment One for the level-based algorithms. For each algorithm (OMP, OMPL with 2 and 4 levels, CoSaMP, and CoSaMPL with 2 and 4 levels) and each combination of m and s, the relative error $\|x - x^\sharp\|_2/\|x\|_2$ is computed. If the relative error is less than 10^{-4}, the recovery is deemed successful, or the algorithm stops computing after 120 iterations. The success probability is defined by the average number of successful recoveries over 50 runs, and plotted in the phase transition diagrams below.

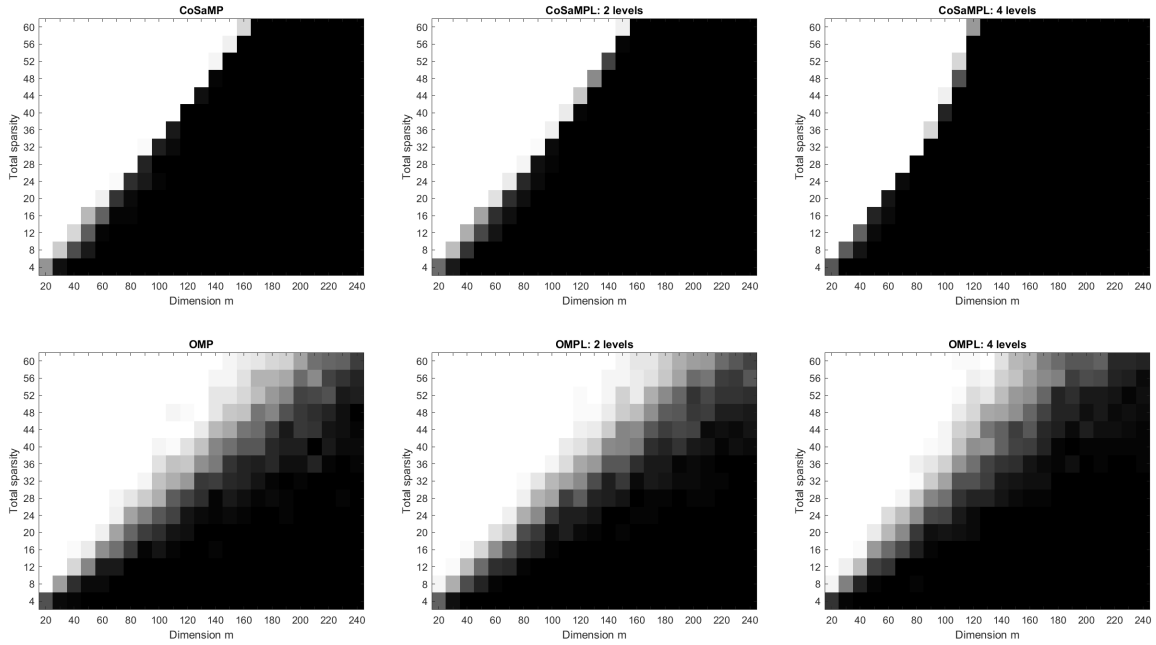

Figure 2: *Phase transition plots displaying the probability of success with total sparsity versus number of linear measurements m. Testing standard CoSaMP and OMP, as well as CoSaMPL and OMPL in 2 and 4 levels with* $M = [N/2, N]$, $s = [3s/4, s/4]$ *and* $M = [N/4, N/2, 3N/4, N]$, $s = [s/2, 0, s/2, 0]$ *respectively.*

We say that an algorithm is sharp when a small variation in m or s causes either no variation in the success probability or a drastic one. The various shades of grey are an indication that OMP and OMPL are less sharp than CoSaMP and CoSaMPL. In an attempt to improve the sharpness of OMP and OMPL, these algorithms are run with $1.5s$ iterations instead of s iterations. With an increase in iterations, if an incorrect index was previously selected, another one can now be selected in the subsequent $s/2$ iterations. In some cases the output vector $x^\sharp$ can be k-sparse, where $k \geq s$; however, the entries that exceed the sparsity level s are negligible (smaller than 10^{-10}). Below are the phase transition plots for OMP and OMPL using the same setting with 2 and 4 levels.

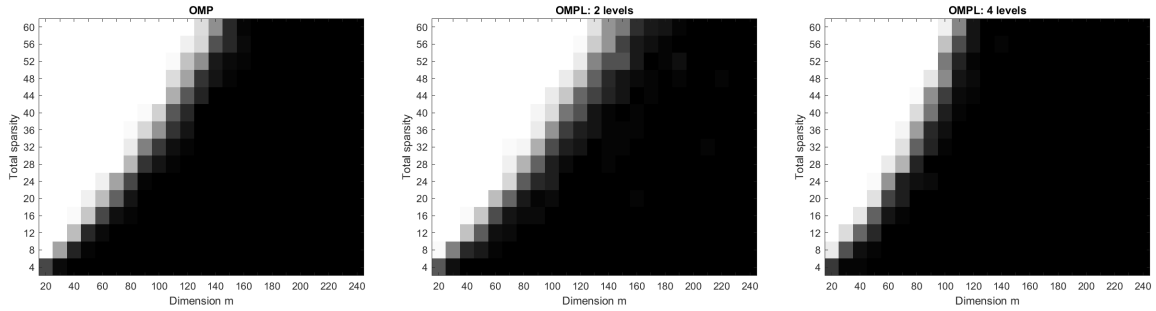

Figure 3: *Phase transition plots displaying the probability of success with total sparsity versus number of linear measurements m. Testing OMP, as well as OMPL in 2 and 4 levels with* $M = [N/2, N]$, $s = [3s/4, s/4]$ *and* $M = [N/4, N/2, 3N/4, N]$, $s = [s/2, 0, s/2, 0]$ *respectively, and each trial performing 1.5s iterations instead of s.*

4.3 Results and Discussion

Examining CoSaMP in the first experiment, two main observations come from these results. First, We can see that the in levels based algorithm CoSaMPL performs better than the standard CoSaMP algorithm. Fewer linear measurements m and iterations are required to recover the vector x. Second, we notice that increasing the number of levels improves the performance and probability of success of the algorithms. With more levels, more information is known a priori about the underlying structure of the vector's sparsity, which - in turn - improves the algorithm's efficiency.

The difference between the experiments using standard OMP and OMPL with 2 levels is marginal, and arguably slightly worse, but the diagrams show an improvement when using 4 levels. We can also see that the algorithm OMP is much less sharp than CoSaMP. By increasing the number of iterations from s to $1.5s$, the vector x has a higher recovery probability. If the vector x is recovered after s iterations, the algorithm will keep choosing indices that are already in the running support of $x^{(n)}$, making no difference. If the vector x is not recovered after s iterations, the additional iterations allow the output vector $x^\sharp$ to be $1.5s$-sparse because subsequent iterations select indices that are not in the current running support of $x^{(n)}$. When the running support of $x^{(n)}$ is larger than the initial support of x, the least square step minimizes the residual $\|y - Ax\|_2$, allowing the indices that were incorrectly selected to be updated to a negligible value (ie less than 10^{-10}), and the residual will be numerically 0. This works as long as the actual support of x is contained in the running support of $x^{(n)}$, because $x^{(n)} - x$ will still be in the kernel nullspace of A.

Running OMP with the maximum number of iterations, N, would not make sense because at this point, the output vector may not be sparse at all. The least square minimization does not promote sparsity, so giving it the flexibility of a very large support of indices to choose from, decreases the probability of recovery. This no longer is compressed sensing, but simply a least square minimization problem. This is why we chose $1.5s$ iterations, as it provides significant improvements to the results without allowing too much flexibility in the least square minimization.

An important point to note, is that initially, a sparsity of 64 was chosen in the second experiment for CoSaMP, and it did not make sense to use this for the following reason. With $N = 256$, and 4 levels used, each level contains 64 entries. If the total sparsity is 64, with half the sparsity in the 1st and 3rd level, then we have 32 non-zero coefficients in each level (half the size of that level). In the first step of the algorithm, when the $2s$ largest entries of each level are retained, the full level is retained in the case when $s = 64$, saturating the level. We then perform a least square problem using every possible index set of each level, and use the thresholding operator to enforce the sparsity s. This simply becomes a matrix multiplication and inverse problem, and the vector x is simply recovered in 1 iteration. This is why $s = 60$ was chosen instead, so the efficacy of the algorithm could truly be tested.

5 Recovery guarantees

This section will solely be discussing the recovery guarantees behind the algorithm CoSaMP, as well as the extension of these to the levels case. First, the sensing matrix A that is used in the algorithm must satisfy the *Restricted Isometry Property (RIP)* with a restricted isometry constant δ_s of order s, where $0 < \delta_s < 1$. A number of matrices satisfy the RIP with high probability such as Gaussian random matrices (used in the numerical experiments), Bernoulli random matrices, or subgaussian random matrices. For the levels case, much of the recovery guarantees are based on the Restricted Isometry Property in Levels (RIPL), defined as follows [1].

Definition 5.1 *[1, Definition 3.3] Let the sparsity levels be $M = (M_1, ..., M_r)$ and local sparsities $\mathbf{s} = (s_1, ...s_r)$. The (s, M)-th Restricted Isometry Constant in levels (RICL) $\delta_{(s,M)}$ of a matrix $A \in \mathbb{C}^{m\times N}$ is the smallest $\delta \leq 0$ such that*

$$(1-\delta)\|x\|_2^2 \leq \|Ax\|_2^2 \leq \|x\|_2^2(1+\delta) \quad \forall x \in \Sigma_{s,M}$$

If $0 < \delta_{s,M} < 1$, then the matrix is said to have the RIPL of order (s, M).

The RIP and RIPL measure the quality of the sensing matrix A by ensuring that A doesn't change the length of the vector x too much as long as x is at most s-sparse or (s, M)-sparse. The goal is to have A behave like an isometry. The theoretical recovery results for CoSaMP for stability and robustness can now be expressed as,

Theorem 5.1 *[6, Theroem 6.28] Suppose that the $8s$th restricted isometry constant of the matrix $A \in \mathbb{C}^{m\times N}$ satisfies*

$$\delta_{8s} < \frac{\sqrt{\frac{11}{3}} - 1}{4} \approx 0.478$$

Then, for $x \in \mathbb{C}^N$ and $e \in \mathbb{C}^m$ the sequence $x^{(n)}$ defined by CoSaMP$(A, y, 2s)$ with $y = Ax + e$ for any $n \geq 0$ satisfies

$$\|x - x^{(n)}\|_1 \leq C\sigma_s(x)_1 + D\sqrt{s}\|e\|_2 + 2\rho^n\sqrt{s}\|x\|_2,$$

$$\|x - x^{(n)}\|_2 \leq \frac{C}{\sqrt{s}}\sigma_s(x)_1 + D\|e\|_2 + 2\rho^n\sqrt{s}\|x\|_2,$$

where the constants $C, D > 0$ and $0 < \rho < 1$ depend solely on δ_{8s}.

This theorem states the guarantees for stability and robustness of the algorithm, with constants C and D providing a scaling between the $l1$ and $l2$ errors, and the best s-term approximation and robustness. The constant ρ establishes the stable Null Space Property of order s relative to any set $S \subset [N]$, with card$(S) \leq s$ [6]. Now as n goes to infinity and if $x^\sharp$ denotes a cluster point of the sequence $x^{(n)}$, ρ^n goes to 0, and the last term vanishes in both upper bounds [6].

To generalize these results to the in levels model, the inclusion of weights must be considered in the best s-term approximation and the $l1$ norm. Assuming that the weights are constant on each level, an ideal choice would be $w^{(k)} = \sqrt{s/s_k}$, where s_k is the local sparsity on the kth level, and s is the global sparsity [1]. We can define the following constants as

$$\zeta = \sum_{i=1}^{r} (w^{(i)})^2 s_i \qquad \xi = \min_{i=1\ldots r} (w^{(i)})^2 s_i$$

to state the recovery results for CoSaMPL, which are

Theorem 5.2 *[1, Therorm 5.2] Suppose that the $(8s, M)$th restricted isometry constant of the matrix $A \in \mathbb{C}^{m\times N}$ satisfies*

$$\delta_{8s,M} < \frac{\sqrt{\frac{11}{3}} - 1}{4} \approx 0.478$$

Then, for $x \in \mathbb{C}^N$ and $e \in \mathbb{C}^m$ the sequence $x^{(n)}$ defined by CoSaMPL$(A, y, 2s, M)$ with $y = Ax + e$ for any $n \geq 0$ satisfies

$$\|x - x^{(n)}\|_{l_w^1} \leq C\frac{\sqrt{\zeta}}{\sqrt{\xi}}\sigma_{s,M}(x)_{l_w^1} + D\sqrt{\zeta}\|e\|_2 + 2\sqrt{\zeta}\rho^n\|x\|_2$$

$$\|x - x^{(n)}\|_2 \leq \frac{E}{\sqrt{\xi}}\sigma_{s,M}(x)_{l_w^1} + F\|e\|_2 + \rho^n\|x\|_2$$

where ρ, C, D, E, F are > 0 and depend only on $\delta_{8s,M}$.

Examining the trivial case $w = 1$ with a single level, one will simply find the recovery guarantees for CoSaMP aforementioned. Now, with the weights selected as mentioned, the constants obtained simplify to $\zeta = rs$, and $\xi = s$, and the recovery guarantees for CoSaMPL for a large enough n are as follows,

$$\|x - x^{(n)}\|_{l_w^1} \lesssim \sqrt{r}\sigma_{s,M}(x)_{l_w^1} + \sqrt{rs}\|e\|_2$$

$$\|x - x^{(n)}\|_2 \lesssim \frac{1}{\sqrt{s}}\sigma_{s,M}(x)_{l_w^1} + \|e\|_2$$

The stability and robustness of this algorithm now scales simply with the total sparsity s and the number of levels r. It is also important to note that the inclusion of weights is not employed in the algorithms, but only as a tool to demonstrate theoretical results [1].

6 Conclusion

This paper has shown the benefits of using an in level structured sparsity with the OMP and CoSaMP decoders to recover a vector x through a number of numerical experiments. This type of model has shown that in some cases, less measurements are required, less iterations are necessary for a successful recovery, and the probability of success increases. Using more levels provides the algorithms with more information about where the sparse coefficients are located, resulting in higher recovery probabilities. This is why the increase in performance is observed mainly when using 4 levels instead of 2. Increasing the number of iterations from s to $1.5s$ in OMP and OMPL provides much sharper results with a higher recovery probability, while still providing a sparse solution.

Some future experiments using the in levels model could include using levels having a similar structure as wavelet scales, where the level size increases exponentially (factor of 2^n), and using a combination of a Discrete Wavelet Transform and Random Gaussian as the sensing matrix A to recover a signal of interest.

References

[1] B. Adcock, S. Brugiapaglia, and M. King-Roskamp, *The benefits of acting locally: Reconstruction algorithms for sparse in levels signals with stable and robust recovery guarantees*, 2020. arXiv: 2006.13389 [cs.IT].

[2] D. Needell and J. A. Tropp, *Cosamp: Iterative signal recovery from incomplete and inaccurate samples*, 2008. arXiv: 0803.2392 [math.NA].

[3] Y. Pati, R. Rezaiifar, and P. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition", in *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, 1993, 40–44 vol.1. DOI: 10.1109/ACSSC.1993.342465.

[4] M. King-Roskamp, "Global guarantees from local knowledge: Stable and robust recovery of sparse in levels vectors", M.S. thesis, Simon Fraser University, 2018.

[5] B.Adcock and A. C. Hansen, *Compressive Imaging: Structure, Sampling, Learning*. Cambridge University Press, 2021.

[6] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*. Birkhauser, 2013.