# Straight-Through meets Sparse Recovery: the Support Exploration Algorithm

Mimoun Mohamed, François Malgouyres, Valentin Emiya, Caroline Chaux

▶ **To cite this version:**

Mimoun Mohamed, François Malgouyres, Valentin Emiya, Caroline Chaux. Straight-Through meets Sparse Recovery: the Support Exploration Algorithm. 2023. hal-03964976v2

**HAL Id: hal-03964976**

**https://hal.science/hal-03964976v2**

Preprint submitted on 7 Feb 2024 (v2), last revised 24 Jun 2024 (v3)

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Public Domain

# Straight-Through meets Sparse Recovery: the Support Exploration Algorithm

Mimoun Mohamed[1,2], François Malgouyres[4], Valentin Emiya[1], and Caroline Chaux[3]

[1]Aix Marseille Univ, CNRS, LIS, Marseille, France
[2]Aix Marseille Univ, CNRS, I2M, Marseille, France
[3]CNRS, IPAL, Singapour
[4]Institut de Mathématiques de Toulouse ; UMR5219 , Université de Toulouse ; CNRS , UPS IMT F-31062 Toulouse Cedex 9, France

February 7, 2024

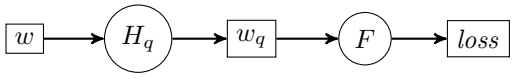
**Abstract**

The *straight-through estimator* (STE) is commonly used to optimize quantized neural networks, yet its contexts of effective performance are still unclear despite empirical successes. To make a step forward in this comprehension, we apply STE to a well-understood problem: *sparse support recovery*. We introduce the *Support Exploration Algorithm* (SEA), a novel algorithm promoting sparsity, and we analyze its performance in support recovery (a.k.a. model selection) problems. SEA explores more supports than the state-of-the-art, leading to superior performance in experiments, especially when the columns of $A$ are strongly coherent. The theoretical analysis considers recovery guarantees when the linear measurements matrix $A$ satisfies the *Restricted Isometry Property* (RIP). The sufficient conditions of recovery are comparable but more stringent than those of the state-of-the-art in sparse support recovery. Their significance lies mainly in their applicability to an instance of the STE.

## 1 Introduction

**Straight-through estimator.** The use of quantized neural networks spares memory, energy, and computing resources during inference, making them essential for embedding neural networks [46, 39]. An effective strategy is to learn the quantized weights. Seminal works [14, 29] rely on full-precision weights $w$ that evolve in the parameter space, while quantized weights $w_q = H_q(w)$ are obtained by applying a piecewise-constant quantization operator $H_q$.

Denoting by $F$ the computational chain from $w_q$ to the loss, the learning procedure $\underset{w}{\text{Minimize}}\, F\left(H_q\left(w\right)\right)$ relies on the computational graph:




Given a step-size $\eta$, the update of $w$ is performed as $w \leftarrow w - \eta \frac{\partial F}{\partial w_q}|_{w_q}$. The motivation, as explained in [28, 5], is that since $\frac{\partial H_q}{\partial w}|_w$ is either undefined or 0, we cannot backpropagate using the chain rule $\frac{\partial F \circ H_q}{\partial w}|_w = \frac{\partial F}{\partial w_q}|_{w_q} \frac{\partial H_q}{\partial w}|_w$. The STE makes the coarse approximation $\frac{\partial F \circ H_q}{\partial w}|_w \approx \frac{\partial F}{\partial w_q}|_{w_q}$ to backpropagate the gradient through the piecewise-constant operator $H_q$. Many subsequent works improve these methods in various aspects [46, 39].

Although STE achieves state-of-the-art performance in training quantized weights for neural networks, it is poorly understood and has not been investigated beyond the context of quantization. We introduce an STE principle for sparsification, leading to a novel algorithm named the Support Exploration Algorithm (SEA) and present experimental evidence of its benefits in challenging, coherent settings such as spike deconvolution, as well as in systematic experiments like the phase transition diagram, see Figure 1. Additionally, we establish theoretical guarantees for the STE-based algorithm.

Figure 1: Overview of the main results. Left: phase transition diagram showing the recovery limits in dimension $n = 500$ while sparsity $k$ and number of observations $m$ varies (the higher, the better, see details in Section 5.1). Right: spike deconvolution in dimension $m = n = 500$ - Average distance between the supports of the solution $x^*$ and the estimations obtained from various algorithms, plotted against the sparsity level $k$ (the lower, the better, see details in Section 5.2).

**Sparse support recovery.** For a sparsity $k \in \mathbb{N}$, we assume $x^* \in \mathbb{R}^n$ is an unknown sparse vector of unknown support $S^* = \text{supp}(x^*)$, of sparsity $|S^*| \leq k$, $A \in \mathbb{R}^{m \times n}$ is a known matrix, and $y = Ax^* + e \in \mathbb{R}^m$ is a linear observation of $x^*$, contaminated with an additive error/noise $e \in \mathbb{R}^m$.

The support recovery objective[1], also coined variable or model selection, searches for a support $S$ with cardinality at most $k$ such that $S^* \subseteq S$. We say that *an algorithm recovers* $S^*$ if it finds such an $S$.

Using a least-square criterion $F(x) = \frac{1}{2}\|Ax - y\|_2^2$, a famous model for support recovery is the optimization problem with sparsity constraint:

$$\underset{x \in \mathbb{R}^n}{\text{Minimize}}\, F(x) \;\text{ s.t. }\; \|x\|_0 \leq k, \tag{1}$$

where $\|x\|_0$ is the $\ell^0$ pseudo-norm of $x$. Problem (1) is known to be NP-hard [16] and to recover the correct support under mild conditions [23, Chapter 5.2.2].

**Proposed STE-based approach for sparse recovery.** Let us define an unconstrained optimization problem which is equivalent to problem (1) and whose structure is compatible with the STE. We set

$$\underset{\mathcal{X} \in \mathbb{R}^n}{\text{Minimize}}\, F(H(\mathcal{X})) \tag{2}$$

where $H$ is the sparsification operator[2]

$$H(\mathcal{X}) \in \underset{\substack{x \in \mathbb{R}^n \\ \text{supp}(x) \subseteq \text{largest}_k(\mathcal{X})}}{\text{argmin}} \frac{1}{2}\|Ax - y\|_2^2. \tag{3}$$

The equivalence between (1) and (2) is established in Appendix A.

Operator $H$ is piece-wise constant and finds the non-zero values in $x$, the sparse support being induced by dense vector $\mathcal{X}$. Formulation (2) has a similar structure as in the quantization case presented above. It uses the suitable loss $F$ and the sparsification operator $H$ in place of the quantification operator $H_q$. Here, vector $\mathcal{X}$ is dense and $x = H(\mathcal{X})$ is $k$-sparse. Applying the STE to the new formulation (2), we obtain the update $\mathcal{X} \leftarrow \mathcal{X} - \eta \frac{\partial F}{\partial x}|_{H(\mathcal{X})}$, for a step-size $\eta > 0$, where we have approximated $\frac{\partial F \circ H}{\partial \mathcal{X}}|_{\mathcal{X}} = \frac{\partial F}{\partial x}|_{H(\mathcal{X})} \frac{\partial H}{\partial \mathcal{X}}|_{\mathcal{X}} \approx \frac{\partial F}{\partial x}|_{H(\mathcal{X})}$. This leads to an original algorithm for sparse recovery, based on the STE, which we analyze in this article.

**Contributions.** The first contribution of the article is to adapt the STE to the sparse support recovery problem (as explained above). Doing so, we obtain a new sparsity-inducing algorithm that we call *Support Exploration Algorithm (SEA)*. It uses the full gradient history over iterations as a heuristic in order to select the next support to optimize over. SEA is supported by support recovery guarantees. In Theorem 4.1 and Corollary 4.2, the sufficient hypotheses guaranteeing the support recovery are on the Restricted Isometry Property (RIP) constants of $A$ and $x^*$. These conditions are comparable to those in the state-of-the-art, albeit slightly more stringent. Their interest mainly lies in the fact that they apply to an instance of STE, for which very few guarantees of convergence exist. However, the successes of SEA observed in the experiments extend to coherent problems where the RIP hypothesis is no longer satisfied. Additional support recovery statements are in Theorem C.4 of Appendix C and in Corollary D.1 of Appendix D. The proofs are based on the interpretation of SEA as a noisy version of an 'Oracle algorithm'which is analyzed in Appendix C.1.

The performances of SEA are compared to those of state-of-the-art algorithms on: 1/ phase-transition synthetic experiments for Gaussian matrices; 2/ spike deconvolution problems; 3/ classification and

[1]The adaptation of the article to "signed support recovery" is possible and is straightforward. We chose to simplify the presentation and not discuss sign recovery.

[2]In scenarios of interest, the minimization problem (3) has a unique and easy to compute solution.

regression problems for real datasets. An important feature of SEA is that it can be used as a post-processing to improve the results of existing algorithms, as shown in the experiments. Also, because SEA has the ability to explore more supports, it performs remarkably well when the matrix $A$ is coherent. The code is available in the git repository of the project. [3]

**Organization of the article.** Related works are detailed in Section 2. Then, SEA is described in Section 3. The theoretical analysis is in Section 4. The experiments are in Section 5. Conclusions and perspectives are in Section 6.

In Appendices, a thorough comparison between SEA and the most similar algorithms of the state-of-the-art is detailed in Appendix B. Appendix B also details an efficient implementation of SEA. The proofs of the theoretical statements as well as complementary support-recovery statements are in Appendices C and D. Complementary experimental results are in Appendices E, F, and G.

# 2 Related works

**On the STE.** Although STE achieves performances defining the state-of-the-art, it is poorly understood. In [33], the authors show that STE behaves well on convex problems and that a stochastic variant of STE does not on non-convex ones. For a two-layer linear neural network with a quantized activation function, a well-chosen STE converges to a critical point of the population risk [45] or reproduces a teacher network [34]. These are the only known formal guarantees for STE. Other works study the large dimension geometry of binary weights [1]. In [27] the authors interpret $w$ as an inertia variable and design a new (related) algorithm. In [13], the authors view the STE as a projected Wasserstein gradient flow.

**On sparse prior and support recovery.** Sparse representations and sparsity-inducing algorithms are widely used in statistics and machine learning [26], as well as in signal processing [23]. For instance, in machine learning, sparse representations are used to select relevant variables. They are also sought to interpret trained models. In signal processing, linear inverse problems have a wide array of applications. The sparsity assumption is ubiquitous since most real signals can be exactly or approximately represented as sparse signals in some domains, e.g., communication signals in Fourier space, natural images in wavelet space. While sparse models are appealing, they are hard to estimate due to the underlying combinatorial difficulty of identifying the correct sparse support.

Our algorithm has been designed in a support recovery context. In the noisy case $e \neq 0$, support recovery is a stronger guarantee than the one in the most standard compressed sensing setting, initiated in [10, 19], when the goal is to upper-bound $\|x - x^*\|_2$, for a well-chosen $x$. The first particularity of support recovery is to assume $x^*$ is truly $k$-sparse – not just compressible. Also, support recovery guarantees always involve a hypothesis on $\min_{i \in S^*} |x_i^*|$, in addition of the incoherence hypothesis on $A$ [44, 36, 48, 9, 47]. We cannot indeed expect to recover an element $i \in S^*$ if $|x_i^*|$ is negligible when compared to all the other quantities involved in the problem [44].

**Support recovery models and algorithms.** Beyond (1), various algorithms were investigated. There are three main families of algorithms: relaxation, combinatorial approaches, and greedy algorithms.

The most famous relaxed model uses the $\ell^1$ norm and is known as the LASSO [41] or Basis Pursuit Algorithm [12]. Combinatorial approaches like Branch and Bound algorithms [4], find the global minimum of (1) but lack scalability. Greedy algorithms can be divided into two categories. Greedy Pursuits like Matching Pursuit (MP) [35] and Orthogonal Matching Pursuit (OMP) [38] are algorithms that start from an empty support and build up an estimate of $x^*$ by iteratively adding components to the current support and optimizing the components. As for thresholding algorithms like Iterative Hard Thresholding (IHT) [8], Hard Thresholding Pursuit (HTP) [24], Compressive Sampling Matching Pursuit (CoSaMP) [37], OMP with Replacement (OMPR) [31], Exhaustive Local Search (ELS) [2] (a.k.a. Fully Corrective Forward Greedy Selection with Replacement [40]) and Subspace Pursuit (SP) [15], they start from any vector and add a replacement step in the iterative process. It allows them to explore various supports before stopping at a local optimum.

[3] For the double-blind review, the anonymized code is in a zipped file in the supplementary materials. This will be replaced by the repository link in the final version of the paper.

**Position of the article.** In this work, we take a different approach and apply the STE to a well-understood problem to compare its behavior, empirical performances, and theoretical guarantees to those of the well-established state-of-the-art.

Compared to other sparse support recovery algorithms, the algorithm introduced in this article may belong to the family of greedy algorithms. A clear difference is the introduction of a non-sparse vector $\mathcal{X}^t \in \mathbb{R}^n$, which evolves during the iterative process and indicates which support should be tested at iteration $t$. We call $\mathcal{X}^t$ the *support exploration variable*. It is the analog of the full-precision weights – used by BinaryConnect that also instantiates the STE – to optimize binary weights of neural networks [14, 30]. We exhibit that the adaptation of STE for sparsification enables a different exploration/exploitation trade-off compared to the state-of-the-art. It explores more. This permits to obtain better performances than the state-of-the-art on very difficult –coherent– problems. We establish that it is possible, in the sparse support recovery context, to obtain theoretical guarantees for the STE.

# 3 Method

After clarifying the notations in Section 3.1, SEA is described in detail in Section 3.2 and its computational complexity is discussed in Section 3.3.

## 3.1 Notations

For any $a, b \in \mathbb{R}$ ($a$ and $b$ can be real numbers), the set of integers between $a$ and $b$ is denoted by $[\![a, b]\!]$ and $\lfloor a \rfloor$ denotes the floor of $a$. For any set $S \subseteq [\![1, n]\!]$, we denote the cardinality of $S$ by $|S|$. The complement of $S$ in $[\![1, n]\!]$ is denoted by $\overline{S}$.

Given $x \in \mathbb{R}^n$ and $i \in [\![1, n]\!]$, the $i^{th}$ entry of $x$ is denoted by $x_i$. The support of $x$ is denoted by $\text{supp}(x) = \{i : x_i \neq 0\}$. The $\ell^0$ pseudo-norm of $x$ is defined by $\|x\|_0 = |\text{supp}(x)|$. The set containing the indices of the $k$ largest absolute entries of $x$ is denoted by $\text{largest}_k(x)$. When ties lead to multiple possible choices for $\text{largest}_k(x)$, we assume $\text{largest}_k(x)$ arbitrarily chooses one of the possible solutions.

For any $S \subseteq [\![1, n]\!]$, $A \in \mathbb{R}^{m \times n}$, and $x \in \mathbb{R}^n$, we define $x_{|S} \in \mathbb{R}^{|S|}$, the restriction of the vector $x$ to the indices in $S$, and $A_S \in \mathbb{R}^{m \times |S|}$, the restriction of the matrix $A$ to the set $S$ as the matrix composed of the columns of $A$ whose indexes are in $S$. The transpose of $A$ is denoted by $A^T \in \mathbb{R}^{n \times m}$. The pseudoinverse of $A$ is denoted by $A^\dagger \in \mathbb{R}^{n \times m}$ and the pseudoinverse of $A_S$ by $A_S^\dagger = (A_S)^\dagger \in \mathbb{R}^{|S| \times m}$. For any $d \in \mathbb{N}$, the identity matrix of size $d$ is denoted by $I_d$. The symbol $\odot$ denotes the Hadamard product.

## 3.2 The Support Exploration Algorithm

The proposed Support Exploration Algorithm (SEA) is given in Algorithm 1. In terms of pseudocode, SEA resembles many state-of-the-art algorithms and is close to HTP and IHT (see comparison between SEA, HTP, and IHT in Appendix B). However, it stands out from the others for its exploratory behavior, which stems from the STE principle behind it.

**Algorithm 1** Support Exploration Algorithm

**Input:** noisy observation $y$, sampling matrix $A$, sparsity $k$, step size $\eta$
**Output:** sparse vector $x$
Initialize $\mathcal{X}^0$
$t \leftarrow 0$
**repeat**
  $S^t \leftarrow \text{largest}_k(\mathcal{X}^t)$
  $\begin{cases} x_i^t & \leftarrow 0 \text{ for } i \in \overline{S^t} \\ x_{S^t}^t & \leftarrow A_{S^t}^\dagger y \end{cases}$
  $\mathcal{X}^{t+1} \leftarrow \mathcal{X}^t - \eta A^T(Ax^t - y)$
  $t \leftarrow t + 1$
**until** halting criterion is *true*
$t_{BEST} \leftarrow \underset{t' \in [\![0,t]\!]}{\text{argmin}} \|Ax^{t'} - y\|_2^2$
**return** $x^{t_{BEST}}$

Each iteration begins by the foward pass given by (3), in which the current sparse solution $x^t$ is computed by applying the sparsification operator $H$ to a dense vector $\mathcal{X}^t$: after selecting the support $S^t$

from $\mathcal{X}^t$ at line 6, the sparse solution $x^t = H(\mathcal{X}^t)$ is computed at line 7. The backward pass uses the STE principle $\frac{\partial F \circ H}{\partial \mathcal{X}}|_{\mathcal{X}^t} \approx \frac{\partial F}{\partial x}|_{x^t} = A^T(Ax^t - y)$ at line 8. To illustrate with the analogy with BinaryConnect [14], the non-sparse vector $\mathcal{X}$ is the analog of the full-precision weights and $H(\mathcal{X})$ is the analog of the quantized weights. To the best of our knowledge, this is the first use of the STE to solve a sparse linear inverse problem.

The key idea is that support $S^t$ is designated at line 6 by a non-sparse variable $\mathcal{X}^t$ called the *support exploration variable*. It offers an original mechanism to explore supports in a more diverse way than existing algorithms. Variable $\mathcal{X}^{t+1} = \mathcal{X}^0 - \eta \sum_{t'=0}^{t} A^T(Ax^{t'} - y)$ is actually an accumulation of gradients taken in the sparse iterates and is used to designate the support of the next sparse iterate $x^{t+1}$. Consequently, unlike other descent-based algorithms, $\mathcal{X}^t$ is not confined to the neighborhood of $k$-sparse vectors. Its evolution is not intended to make the objective function decrease at each iteration. In this regard, since the algorithm explores supports in a way that allows the functional to sometimes increase, the retained solution is the best one encountered along the iterations (line 11). Illustrations of this phenomenon are given in Appendix F.2 where one can see that the behavior of the loss along the iterations shows important variations when a new support is explored. This is an important difference with the aforementioned state-of-the-art algorithms, resulting in increased exploration.

An important feature of SEA is that it can be used as a post-processing of the solution $\hat{x}$ of another algorithm. This is simply done by initializing $\mathcal{X}^0 = \hat{x}$. In this case $S^0 = \text{supp}(\hat{x})$ (line 6) and $x^0$ improves or is equal to $\hat{x}$ (line 7). In the experiments, we have investigated the initialization with the result of OMP [38], ELS [2, 40] and the initialization $\mathcal{X}^0 = 0$. We observe that the initialization with ELS is generally preferable except for difficult problems, when columns of $A$ are very coherent (see Section 5.2).

Finally, as often, there are many possible strategies to design the halting criterion of the 'repeat' loop of Algorithm 1. It is clear that a more permissive criterion allows for more exploration and better results, at the expense of computation time. We have not investigated this aspect in the experiments and leave this study for the future. We preferred to focus our experiments on the illustration of the potential benefits of SEA and, as a consequence, we always used a large fixed number of passes in the 'repeat' loop of Algorithm 1.

Similarly, since we have $\mathcal{X}^t = \mathcal{X}^0 - \eta \sum_{t'=0}^{t-1} A^T(Ax^{t'} - y)$ for all $t > 1$, $\eta$ has no impact on $S^t$ and $x^t$ when $\mathcal{X}^0 = 0$ and therefore on the output $x^{t_{BEST}}$ of Algorithm 1. In this case, indeed, the whole trajectory $(\mathcal{X}^t)_{t\in\mathbb{N}}$ is dilated by $\eta > 0$ and the dilation has no effect on the selected supports $S^t$. When $\mathcal{X}^0 \neq 0$, the initial support exploration variable is forgotten as the iterations progress. It is forgotten more rapidly when $\eta$ increases. We have not studied the tuning of this parameter in depth, leaving it for future research.

### 3.3 Computational complexity

An efficient implementation of SEA is described in Appendix B.2. The analysis of the computational complexity of SEA is based on two facts. First, if the support $S^t$ obtained at line 6 has already been explored, then the sparse vector $x^t$ and the gradient $\eta A^T(Ax^t - y)$ have already been computed. So, if these quantities have been memorized (as in Algorithm 5, Appendix B.2), the cost of the iteration is negligible. The overall cost thus depends on the number of explored supports rather than on the number of iterations. Second, each time a new support is extracted, the cost of the iteration is dominated by solving the (unconstrained) linear system $A_{S^t}^T A_{S^t} x_{S^t} = A_{S^t}^T y$. While the pseudo-inverse is a convenient notation at line 7, the solution may be obtained more efficiently, e.g. in $\mathcal{O}(k^2 n)$ to compute $A_{S^t}^T A_{S^t}$ and $A_{S^t}^T y$, and apply the conjugate gradient algorithm. The overall complexity is thus in $\mathcal{O}(n_{\text{supp}} k^2 n)$ where $n_{\text{supp}}$ is the number of supports actually explored.

The complexity of HTP, OMP, OMPR and ELS is also dominated by the number of times $A_S^T A_S x_S = A_S^T y$ is solved for $S$ such that $|S| = k$. As for SEA, efficient implementations of HTP, OMPR and ELS can save computations by storing all the explored support and related iterates. The HTP, OMP and OMPR then depend on the number of explored supports in a similar way as SEA. The OMP solves $k$ instances of them which results in less exploration and less computational cost. The ELS is much more demanding since it explores $(n - k)$ supports at each iteration, many of which are irrelevant. IHT has a lower complexity than SEA since it never inverses the system $A_S^T A_S x_S = A_S^T y$.

As we will see in the deconvolution experiments in Section 5.2, SEA outperforms ELS (see Figure 4) while exploring two times less supports (see Appendices F.2 and F.3).

# 4 Theoretical analysis

In this section, we provide the theorem stating that SEA recovers the correct support for some[4] $x^*$ when the matrix $A$ satisfies a RIP constraint. Then, we compare the conditions with existing support recovery conditions for the LASSO, OMP, and HTP. In addition to the statements in this section, recovery statements are given in Appendices C and D. The interest of the theorems lies mainly in the fact that they apply to an instance of STE, for which guarantees are rare. From the practitioner's point of view, the theoretical analysis is not useful since SEA mostly shows promises in coherent scenarios in which the RIP hypothesis is not satisfied.

In this section, we assume that columns of $A$ are normalized: for any $i \in [\![1, n]\!]$, $\|A_i\|_2 = 1$. As has been standard practice since Candès and Tao first proposed it in [11], we define for all $l \in [\![1, n]\!]$ the $l$th Restricted Isometry Constant of $A$ as the smallest non-negative number $\delta_l$ such that for any $x \in \mathbb{R}^n$, such that $\|x\|_0 \leq l$,

$$(1 - \delta_l)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta_l)\|x\|_2^2. \tag{4}$$

If $\delta_l < 1$, $A$ is said to satisfy the Restricted Isometry Property of order $l$ or the $l$-RIP.

In this section, we assume that $A$ satisfies the $(2k+1)$-RIP. In the scenarios of interest, $\delta_{2k+1}$ is small. We define

$$\alpha_k^{RIP} = \delta_{2k+1}\left(\frac{\delta_{2k}}{1-\delta_k} + 1\right) \in \mathbb{R}_+^*$$

$$\text{and} \qquad \gamma_k^{RIP} = \delta_{2k+1}\frac{\sqrt{1+\delta_k}}{1-\delta_k} + 1 \in \mathbb{R}_+^*. \tag{5}$$

As soon $\delta_k$ is far from 1 (for example $\delta_k \leq \frac{1}{2}$), $\alpha_k^{RIP}$ has the order of magnitude of $\delta_{2k+1}$ (in the example $\delta_{2k+1} \leq \alpha_k^{RIP} \leq 3\delta_{2k+1}$) and $\gamma_k^{RIP}$ has the order of magnitude of $1 + \delta_{2k+1}$ (in the example $(1 + \delta_{2k+1}) \leq \gamma_k^{RIP} \leq \sqrt{6}(1 + \delta_{2k+1})$).

As is typical of support recovery statements, the next theorem includes a condition on $x^*$. We call this condition *the Recovery Condition for the RIP case ($RC_{RIP}$)*. It is defined by

$$\gamma_k^{RIP}\|e\|_2 < \frac{\min_{i\in S^*}|x_i^*|}{2k} - \alpha_k^{RIP}\|x^*\|_2. \tag{$RC_{RIP}$}$$

**Theorem 4.1** (Recovery - RIP case)**.** *Assume $A$ satisfies the $(2k+1)$-RIP and[5] for all $i \in [\![1, n]\!]$, $\|A_i\|_2 = 1$. Assume moreover that $x^*$ satisfies ($RC_{RIP}$).*

*Then for all initializations $\mathcal{X}^0$ and all $\eta > 0$, there exists $t_s \leq T_{RIP}$ such that $S^* \subseteq S^{t_s}$, where*

$$T_{RIP} = \frac{2k\frac{\|\mathcal{X}^0\|_\infty}{\eta} + (k+1)\min_{i\in S^*}|x_i^*|}{\min_{i\in S^*}|x_i^*| - 2k\left(\alpha_k^{RIP}\|x^*\|_2 + \gamma_k^{RIP}\|e\|_2\right)}. \tag{6}$$

*If moreover, $x^*$ is such that*

$$\min_{i\in S^*}|x_i^*| > \frac{2}{\sqrt{1-\delta_{2k}}}\|e\|_2 \tag{7}$$

*and SEA performs more than $T_{RIP}$ iterations, then $S^* \subseteq S^{t_{BEST}}$ and $\|x^{t_{BEST}} - x^*\|_2 \leq \frac{2}{\sqrt{1-\delta_k}}\|e\|_2$.*

The proof is in Appendix C. To introduce the proof and provide the main intuition, we first detail in Appendix C.1 a theorem and its proof stating that the support exploration algorithm using an Oracle Update Rule, see Algorithm 6, always recovers the true support $S^*$. In Theorem C.4 of Appendix C.2, we provide a sufficient condition on the discrepency between the oracle-update and the STE-update guaranteeing that SEA visits the true support $S^*$. Then, in Appendix C.3, we establish that the hypotheses of Theorem 4.1 ensure that the discrepency is sufficiently small to satisfy the hypothesis of Theorem C.4.

As a sanity check, we establish in Corollary D.1 of Appendix D that when the columns of $A$ are orthonormal and $e = 0$, SEA recovers $x^*$ in less than $k + 1$ iterations.

[4] In particular it is necessary that $\min_{i\in S^*}|x_i^*|$ is sufficiently large.

[5] The normalization aims at simplifying formulas by guaranteeing that $\delta_1 = 0$. It is done at no expense since, if $A$ is not normalized but satisfies (4) for $l > 1$, its normalization only has a small impact on $\delta_l$. Indeed, considering $\Delta \in \mathbb{R}^{n\times n}$ diagonal such that $\Delta_{i,i} = \|A_i\|_2$, $A\Delta^{-1}$ is normalized and for all $l$-sparse vector $x$

$$(1 - \delta_l)\|\Delta^{-1}x\|_2^2 \leq \|A\Delta^{-1}x\|_2^2 \leq (1 + \delta_l)\|\Delta^{-1}x\|_2^2.$$

Using $1 - \delta_1 \leq \|A_i\|_2^2 \leq 1 + \delta_1$, we can derive $l$-RIP constants for the normalized matrix $A\Delta^{-1}$.

When $(RC_{RIP})$ holds, $T_{RIP}$ increases as $\min_{i\in S^*}|x_i^*| - 2k\left(\alpha_k^{RIP}\|x^*\|_2 + \gamma_k^{RIP}\|e\|_2\right)$ decreases. In particular, the number of iterations required by the algorithm to provide the correct solution increases when the information on some of the columns of $S^*$ diminishes, i.e. when $\min_{i\in S^*}|x_i^*|$ decreases. Also, the initializations $\mathcal{X}^0 \neq 0$ have an apparent negative impact on the number of iterations required in the worst case. This is because in the worst-case $\mathcal{X}^0$ is poorly chosen and SEA needs iterations to correct this poor choice.

When possible, any $\mathcal{X}^0$ and $\eta$ permit the recovery of $S^*$. $\mathcal{X}^0$ and $\eta$ only influence $T_{RIP}$. In this regard, since the larger $\eta$, the faster SEA overrides the initialization $\mathcal{X}^0$, the choice of $\eta$ is very much related to the question of the quality of the initialization. The latter is often beneficial in practice.

To illustrate $(RC_{RIP})$, we provide below a simplified condition which is shown in Corollary 4.2 to be stronger than $(RC_{RIP})$ in the noiseless scenario. We say $x^*$ satisfies the Simplified Recovery Condition in the RIP case if there exists $\Lambda \in (0,1)$ such that

$$2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x_i^*|} \leq \Lambda. \qquad (RC_{SRIP})$$

**Corollary 4.2** (Noiseless recovery - simplified RIP case)**.** *Assume $\|e\|_2 = 0$, $A$ satisfies the $(2k+1)$-RIP and for all $i \in [\![1,n]\!]$, $\|A_i\|_2 = 1$.*

*If moreover $x^*$ satisfies $(RC_{SRIP})$, then $x^*$ satisfies $(RC_{RIP})$. As a consequence, for $\mathcal{X}^0 = 0$ and for all $\eta > 0$, if SEA performs more than $T_{SRIP} = \frac{k+1}{1-\Lambda}$ iterations, we have $S^* \subseteq S^{t_{BEST}}$ and $x^{t_{BEST}} = x^*$.*

The proof is in Appendix C.4.

Compared to the support recovery guarantees for the LASSO [44, 36, 48], the OMP [9], the HTP [24, 47] and the ARHT [2] the recovery conditions provided in Theorem 4.1 and Corollary 4.2 for SEA are stronger. All conditions involve a condition on the incoherence of $A$ and a condition similar to (7). In the case of the LASSO algorithm, the latter is not very explicit. However, none of the support recovery conditions involve a condition like $(RC_{RIP})$ and $(RC_{SRIP})$. A clear drawback of these conditions is that the support of an $x^*$ such that $\max_{i\in S^*}|x_i^*| \gg \min_{i\in S^*}|x_i^*|$ is not guaranteed to be recovered. This is because, if $i \notin S^t$ and $|x_i^*| \gg \min_{i\in S^*}|x_i^*|$, the discrepancy between the Oracle update, described in Appendix C.1, and the STE update can be large. However, it is possible to get around this problem since SEA inherits the support recovery properties of any well-chosen initialization. Also, we have not observed this phenomenon in the experiments of Section 5. Additional comments on $(RC_{RIP})$ and $(RC_{SRIP})$ are provided in Appendix C.5.

As will be seen later in Section 5.2, SEA performs well even when $A$ is coherent. This is not explained by Theorem 4.1 and Corollary 4.2 which use the RIP assumption. The main interest of the above theoretical results lies in the fact that they apply to an instance of the STE. Another theory needs to be developed to explain the good behavior of SEA for sparse support recovery when $A$ is coherent.

# 5 Experimental analysis

We compare SEA to state-of-the-art algorithms on two tasks in the noisy setting: phase transition diagrams (Section 5.1 and Appendix E) and spike deconvolution problems for signal processing (Section 5.2 and Appendix F). For completeness, additional comparisons between SEA and state-of-the-art algorithms for linear and logistic regression tasks in supervised learning settings are provided in Appendix G.

The tested algorithms are Iterative Hard Thresholding (IHT) [8], Hard Thresholding Pursuit (HTP) [24], Orthogonal Matching Pursuit [35, 38], OMP with Replacement (OMPR) [31] and Exhaustive Local Search (ELS) [2]. OMPR and ELS are initialized with the solution of OMP. Three versions of SEA are studied: the cold-start version $\text{SEA}_0$, where SEA is initialized with the null vector, and the warm-start versions $\text{SEA}_{\text{ELS}}$ and $\text{SEA}_{\text{OMP}}$, where SEA is initialized with the solutions of ELS and OMP, respectively. We have also studied HTP and IHT initialized with OMP and ELS. They are called $\text{HTP}_{\text{OMP}}$, $\text{HTP}_{\text{ELS}}$, $\text{IHT}_{\text{OMP}}$ and $\text{IHT}_{\text{ELS}}$.

For all algorithms, each least-square projection for a fixed support, as in Line 7 of Algorithm 1, is solved using the conjugate gradient descent of SciPy [43]. For all algorithms, $256k$ iterations are performed. The results of HTP and to a lesser extent IHT and SEA depend on the choice of the stepsize. For the sake of fairness of the comparison with OMP, OMPR and ELS, we did not optimize the choice of the stepsize. The stepsize of SEA, HTP, and IHT is arbitrarily[6] fixed to $\eta = \frac{1.8}{L}$, where $L$ is the

[6]We do not report further experiments for $\eta \in \{\frac{2^l}{L} \mid \text{for } l = [\![-3,+3]\!]\}$ that do not significantly alter the results in terms of running time, stability, performance, and do not impact our conclusions.

Figure 2: Phase transition diagram: each curve is the threshold below which the related algorithm recovers at least 95% of the supports. $\zeta$ denotes the ratio between the number of rows and the number of columns in $A$ while $\rho$ denotes the ratio between the sparsity and the number of rows in $A$. Matrix $A$ have i.i.d. standard Gaussian entries and non-zero entries in $x^*$ are drawn uniformly in $[-2,-1]\cup[1,2]$. $n = 500$ is fixed and results are obtained from 1000 runs.

spectral radius of $A$. The columns of $\mathbf{A}$ are normalized before solving the problem. The sparse vector $x^* \in \mathbb{R}^n$ is random. Indexes of the support are randomly picked, uniformly without replacement. The non-zero entries of $x^*$ are drawn uniformly in $[-2,-1]\cup[1,2]$ as in [23]. The noise $e$ is drawn uniformly using the same method as described in [7]. Their detailed descriptions are in the next two sections. For each experiment, the metrics used for performance evaluation are defined in the corresponding subsection. The code is implemented in Python 3 and is available in the git repository of the project [7]. As explained in Section 3.3 and in Appendix B.2, the computational cost of SEA mainly depends on the number of explored supports. The illustration related to the number of explored supports for a fixed number of iterations and the efficiency of the exploration can be found in Appendix F.3.

## 5.1 Phase transition diagram experiment

Introduced by Donoho and Tanner [18] and used in compressed sensing [20, 25], phase transition diagrams show the recovery limits of an algorithm depending on the undersampling/indeterminacy $\zeta = \frac{m}{n}$ of $A$, and the sparsity/density $\rho = \frac{k}{m}$ of $x^*$. We fix $n = 500$, $m$ takes 18 values in $[\![1, n]\!]$ and $k$ all values in $[\![1, 0.5m]\!]$. For each triplet $(m, n, k)$ and each algorithm, we run $r = 1000$ experiments (described below) to assess the success rate $\frac{s_{\zeta,\rho}}{r}$ of the algorithm, where $s_{\zeta,\rho}$ is the number of problems successfully solved. A problem is considered successfully solved if the support of the output of the algorithm is equal to $S^*$. For each run, the entries of $A \in \mathbb{R}^{m\times n}$ are drawn independently from the standard normal distribution. The restricted isometry constants are poor when $\zeta = \frac{m}{n}$ is small and improve when $m$ grows [3]. The noise $e$ is drawn uniformly from the sphere of radius $0.01\|Ax^*\|_2$ in $\mathbb{R}^m$. Figure 2 shows results from this experiment. Each curve indicates the threshold below which the algorithm has a success rate larger than 95%. The higher the curve, the better. We see that OMP, HTP and IHT achieve poor recovery successes. The smooth, decreasing part of the HTP and IHT curves on the left is an artifact due to the discrete values of $(m, n, k)$ and actually corresponds to a phase transition located at $k = 1$. $\text{SEA}_0$ outperforms OMP, HTP and IHT when $\frac{m}{n} < 0.6$. All the OMP-initialized algorithms (in blue) improve OMP performance except in the most coherent cases ($\frac{m}{n} < 0.2$) where $\text{HTP}_{\text{OMP}}$ and $\text{IHT}_{\text{OMP}}$ fail while $\text{SEA}_{\text{OMP}}$ exhibits the best improvement. Contrary to $\text{HTP}_{\text{ELS}}$ and $\text{IHT}_{\text{ELS}}$, $\text{SEA}_{\text{ELS}}$ (in red) improves further ELS performances and outperforms the other algorithms for all $\frac{m}{n}$. The main improvements are when $\frac{m}{n}$ is small ($\frac{m}{n} < 0.4$), i.e., for the most coherent matrices $A$. Thus, SEA refines a good support candidate into a better one by exploring new supports and achieves recovery for higher values of sparsity $k$ than competitors. The actual superiority of $\text{SEA}_{\text{ELS}}$ and $\text{SEA}_{\text{OMP}}$ for coherent matrices ($\frac{m}{n} < 0.3$) is a major conclusion from this experiment and illustrates its ability to successfully explore supports in difficult problems where competitors fail. We study the noiseless setup (i.e., $e = 0$) in Appendix E.

## 5.2 Deconvolution experiment

Deconvolution purposes arise in many signal processing areas such as microscopy or remote sensing. Of particular interest is the deconvolution of sparse signals, also known as point source deconvolution [6] or spike deconvolution [22, 21], assuming the linear operator is known (contrary to blind approaches [32]). The objective is to recover spike positions and amplitudes.

We set $n = 500$, a convolution matrix $A$ corresponding to a Gaussian filter with a standard deviation equal to 3. The coherence of matrix $A$ is $\max_{i\neq j} |A_i^T A_j| = 0.97$, resulting in very difficult problems for which the support recovery theorems do not apply. For each sparsity level $k \in [\![1, 50]\!]$, every algorithm is tested on $r = 200$ distinct problems corresponding to different $k$-sparse vectors $x^*$. The maximal number of iterations is 1000, for all algorithms. The noise $e$ is drawn uniformly from the sphere of radius $0.1\|Ax^*\|_2$ of $\mathbb{R}^m$, aiming for a signal-to-noise ratio of 20dB.

Figure 3 illustrates the results for a 20-sparse vector $x^*$ restricted to a crowded area of the full signal (the later being depicted in Appendix F.1). Generally speaking, isolated spikes are recovered by almost all algorithms. However, algorithms often fail to accurately identify spikes when they are close to each

[7] For the double-blind review, the anonymized code is in a zipped file in the supplementary materials.

Figure 3: Spike deconvolution: representation of an instance of $x^*$ and $y$ with the solutions provided by the algorithms when $k = 20$. This is a cropped version of a crowded area (spikes are close).

Figure 4: Spike deconvolution: average support distance between $S^*$ and the support of the solutions provided by several algorithms as a function of the sparsity level $k$.

other. For instance, ELS, OMP and OMPR falsely detect entries in the highest energetic part of the signal (around position 400) and are trapped in a local minimum. $\text{SEA}_0$, $\text{SEA}_{\text{OMP}}$, and $\text{SEA}_{\text{ELS}}$ recover the original signal with a better precision than its competitors. It is worth mentioning that only SEA recovers perfectly this signal in the noiseless settings (see Appendix F.5.2). To illustrate the exploratory behavior of SEA, we show in Appendix F.2, the evolution of $\|Ax^t - y\|_2$ when $t$ and the number of explored supports varies, for the experiment of Figure 3.

On Figure 4, for each algorithm and for all $k \in [\![1, 50]\!]$, we display the support distance metric [23] averaged over $r = 200$ runs and defined by $\text{dist}_{\text{supp}}(x) = \frac{k-|S^* \cap \text{ supp}(x)|}{k}$ (the lower the distance, the better). For all considered sparsity values, $\text{SEA}_0$, $\text{SEA}_{\text{OMP}}$, and $\text{SEA}_{\text{ELS}}$ outperform the other algorithms. SEA improves OMP and ELS results while they are never enhanced by HTP nor IHT (curves are superimposed). Note that for small $k$, IHT shows poor performance because it assigns several neighboring elements of the support to the largest peak of $y$ and fails to correct this error afterward. As $k$ increases, due to the increasing difficulty of the problem, the algorithms are gradually becoming unable to recover $S^*$. Using a cold-start strategy, $\text{SEA}_0$ is here the best performing algorithm. The analyses conducted in Appendix F.3 indicate that the exploration carried out by SEA can be more efficient than the support element swaps performed by ELS. These experiments also suggest that a warm-start strategy, such as $\text{SEA}_{\text{ELS}}$ or $\text{SEA}_{\text{OMP}}$, may lead the algorithm to get trapped in a local minimum. The choice of the best strategy appears to depend on the quality of the initialization. We recommend selecting it based on empirical performance. The same conclusions are drawn when using additional metrics (Appendix F.4) and in the noiseless case (Appendix F.5).

# 6 Conclusions and perspectives

In this article, we proposed SEA: a new principled algorithm for sparse support recovery, based on STE. Experiments show that SEA supplements state-of-the-art algorithms and outperforms them in particular when $A$ is coherent, thanks to its better exploration ability. Indeed, SEA initialized with the output of ELS is generally a good strategy to try to improve recovery results. Nonetheless, the cold-start strategy where SEA is initialized at 0 may also be profitable: it is the best setting in problems with very coherent matrices like in the deconvolution experiment. Understanding which strategy should be preferred remains an open question.

We established guarantees when the matrix $A$ satisfies the RIP, which we hope gives new insight on the STE. The theoretical guarantees involve conditions on $x^*$ that are not present for similar statements for other algorithms and that might restrict their applicability. Improving the theoretical analysis in the following directions are promising perspective. The algorithm perform well when $A$ is coherent: this is not explained by the current theoretical analysis which only applies to matrices satisfying the RIP. Checking explicitly RIP conditions being NP-hard [42], we will investigate theoretical guarantees based on mutual incoherence [17]. Also it would be interesting to adapt the strategy developed for obtaining the theoretical guarantees to other contexts, such as the optimization of quantized neural networks.

Finally, this paper opens up broader perspectives. The proposed STE is a deterministic approach for support exploration and may also be compared to or extended by the use of stochastic heuristics. There are many perspectives of SEA and STE applications to sparse inverse problems such as sparse matrix factorization, tensor problems, as well as real-world applications such as in biology and astronomy.

# 7 Acknowledgement

This work has benefited from the AI Interdisciplinary Institute ANITI, which is funded by the French "Investing for the Future – PIA3" program under the Grant agreement ANR-19-P3IA-0004. F. Malgouyres gratefully acknowledges the support of IRT Saint Exupéry and the DEEL project[8] and thanks Franck

[8] `https://www.deel.ai/`

Mamalet for all the discussions on the STE.

M. Mohamed was suppported by a PhD grant from "Emploi Jeunes Doctorants (EJD)" plan which is funded by the French institution "Région Sud - Provence-Alpes-Côte d'Azur" and Euranova France. M. Mohamed gratefully acknowledges their financial support.

# References

[1] Alexander G. Anderson and Cory P. Berg. The high-dimensional geometry of binary neural networks. In *International Conference on Learning Representations*, 2018.

[2] Kyriakos Axiotis and Maxim Sviridenko. Sparse convex optimization via adaptively regularized hard thresholding. In *Proc. Int. Conf. Mach. Learn.*, volume 119 of *Proceedings of Machine Learning Research*, pages 452–462. PMLR, 13–18 Jul 2020.

[3] Bubacarr Bah and Jared Tanner. Bounds of restricted isometry constants in extreme asymptotics: formulae for Gaussian matrices. *Lin. Algebra Appl.*, 441:88–109, 2014.

[4] Ramzi Ben Mhenni, Sébastien Bourguignon, and Jordan Ninin. Global optimization for sparse solution of least squares problems. *Optim. Methods Softw.*, 37(5):1740–1769, 2022.

[5] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, arXiv:1308.3432, 2013.

[6] Brett Bernstein and Carlos Fernandez-Granda. Deconvolution of point sources: A sampling theorem and robustness guarantees. *Comm. Pure Appl. Math.*, 72(6):1152–1230, 2019.

[7] Jeffrey D Blanchard and Jared Tanner. Performance comparisons of greedy algorithms in compressed sensing. *Numerical Linear Algebra with Applications*, 22(2):254–282, 2015.

[8] Thomas Blumensath and Mike E. Davies. Iterative hard thresholding for compressed sensing. *Appl. Comput. Harmon. Analysis*, 27(3):265–274, 2009.

[9] T Tony Cai and Lie Wang. Orthogonal matching pursuit for sparse signal recovery with noise. *IEEE Trans. Inform. Theory*, 57(7):4680–4688, 2011.

[10] Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory*, 52(2):489–509, 2006.

[11] Emmanuel J Candes and Terence Tao. Decoding by linear programming. *IEEE Trans. Inform. Theory*, 51(12):4203–4215, 2005.

[12] Scott Shaobing Chen, David L Donoho, and Michael A Saunders. Atomic decomposition by basis pursuit. *SIAM Rev.*, 43(1):129–159, 2001.

[13] Pengyu Cheng, Chang Liu, Chunyuan Li, Dinghan Shen, Ricardo Henao, and Lawrence Carin. Straight-through estimator as projected wasserstein gradient flow. In *Third workshop on Bayesian Deep Learning (NeurIPS)*, 2018.

[14] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Adv. Neural Inf. Process. Syst.*, volume 28, pages 3123–3131, Montreal, Quebec, Canada, Dec. 7–12 2015.

[15] Wei Dai and Olgica Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *IEEE Trans. Inform. Theory*, 55(5):2230–2249, 2009.

[16] Geoff Davis, Stephane Mallat, and Marco Avellaneda. Adaptive greedy approximations. *Constr. Approx.*, 13(1):57–98, 1997.

[17] David Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *IEEE Trans. Inform. Theory*, 47(7):2845–2862, 2001.

[18] David Donoho and Jared Tanner. Observed universality of phase transitions in high-dimensional geometry, with implications for modern data analysis and signal processing. *Phil. Trans. R. Soc. A*, 367(1906):4273–4293, 2009.

[19] David L Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52(4):1289–1306, 2006.

[20] David L Donoho, Arian Maleki, and Andrea Montanari. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919, 2009.

[21] Vincent Duval and Gabriel Peyré. Sparse spikes super-resolution on thin grids I: the LASSO. *Inverse Problems*, 33(5):055008, 2017.

[22] Vincent Duval and Gabriel Peyré. Exact support recovery for sparse spikes deconvolution. *Found. Comput. Math.*, 15(5):1315–1355, 2015.

[23] Michael Elad. *Sparse and Redundant Representations*. Springer New York, NY, 1 edition, 2010.

[24] Simon Foucart. Hard thresholding pursuit: an algorithm for compressive sensing. *SIAM J. Numer. Anal.*, 49(6):2543–2563, 2011.

[25] Simon Foucart, Holger Rauhut, Simon Foucart, and Holger Rauhut. *An invitation to compressive sensing*. Springer, 2013.

[26] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical Learning with Sparsity*. Chapman and Hall/CRC, May 2015.

[27] Koen Helwegen, James Widdicombe, Lukas Geiger, Zechun Liu, Kwang-Ting Cheng, and Roeland Nusselder. Latent weights do not exist: Rethinking binarized neural network optimization. *Advances in neural information processing systems*, 32, 2019.

[28] Geoffrey E. Hinton. Neural networks for machine learning. Coursera, video lectures, 2012. Lecture 15b.

[29] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. *Advances in neural information processing systems*, 29, 2016.

[30] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. *Adv. Neural Inf. Process. Syst.*, 29, Dec. 5–10 2016.

[31] Prateek Jain, Ambuj Tewari, and Inderjit Dhillon. Orthogonal matching pursuit with replacement. *Adv. Neural Inf. Process. Syst.*, 24, Dec. 12–17 2011.

[32] Han-Wen Kuo, Yenson Lau, Yuqian Zhang, and John Wright. Geometry and symmetry in short-and-sparse deconvolution. In *Proc. Int. Conf. Mach. Learn.*, volume 97 of *Proceedings of Machine Learning Research*, pages 3570–3580. PMLR, 09–15 Jun 2019.

[33] Hao Li, Soham De, Zheng Xu, Christoph Studer, Hanan Samet, and Tom Goldstein. Training quantized nets: A deeper understanding. In *Advances in Neural Information Processing Systems*, pages 5811–5821, 2017.

[34] Ziang Long, Penghang Yin, and Jack Xin. Learning quantized neural nets by coarse gradient method for nonlinear classification. *Research in the Mathematical Sciences*, 8:1–19, 2021.

[35] Stéphane G Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Trans. Signal Process.*, 41(12):3397–3415, 1993.

[36] Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the Lasso. *Ann. Statist.*, 34(3):1436–1462, 2006.

[37] Deanna Needell and Joel A Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comput. Harmon. Analysis*, 26(3):301–321, 2009.

[38] Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proc. Asilomar Conf. Signal Syst. Comput.*, volume 1, pages 40–44, Pacific Grove, CA, USA, 1993. IEEE.

[39] Ratshih Sayed, Haytham Azmi, Heba A. Shawkey, Alaa Hussein Khalil, and Mohamed Refky. A systematic literature review on binary neural networks. *IEEE Access*, 11:27546–27578, 2023.

[40] Shai Shalev-Shwartz, Nathan Srebro, and Tong Zhang. Trading accuracy for sparsity in optimization problems with sparsity constraints. *SIAM J. Optim.*, 20(6):2807–2832, 2010.

[41] Robert Tibshirani. Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 58(1):267–288, 1996.

[42] Andreas M. Tillmann and Marc E. Pfetsch. The computational complexity of the restricted isometry property, the nullspace property, and related concepts in compressed sensing. *IEEE Trans. Inform. Theory*, 60(2):1248–1259, 2014.

[43] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[44] Martin J Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using $\ell_1$-constrained quadratic programming (Lasso). *IEEE Trans. Inform. Theory*, 55(5):2183–2202, 2009.

[45] Penghang Yin, Jiancheng Lyu, Shuai Zhang, Stanley J. Osher, Yingyong Qi, and Jack Xin. Understanding straight-through estimator in training activation quantized neural nets. In *International Conference on Learning Representations*, 2019.

[46] Chunyu Yuan and Sos S. Agaian. A comprehensive review of binary neural network. *Artificial Intelligence Review*, pages 1–65, 2023.

[47] Xiaotong Yuan, Ping Li, and Tong Zhang. Exact recovery of hard thresholding pursuit. In *Adv. Neural Inf. Process. Syst.*, volume 29, pages 3558–3566, Barcelona, Spain, Dec. 5–10 2016.

[48] Peng Zhao and Bin Yu. On model selection consistency of Lasso. *J. Mach. Learn. Res.*, 7:2541–2563, 2006.

# A Problem statement

The equivalence between problem (1) and problem (2) is established by the following proposition. Before stating the proposition, let us remind

$$F(x) = \frac{1}{2}\|Ax - y\|_2^2 \qquad , \forall x \in \mathbb{R}^n \tag{8}$$

and

$$H(\mathcal{X}) \in \underset{\substack{x \in \mathbb{R}^n \\ \text{supp}(x) \subseteq \text{largest}_k(\mathcal{X})}}{\text{argmin}} \frac{1}{2}\|Ax - y\|_2^2 \qquad , \forall \mathcal{X} \in \mathbb{R}^n. \tag{9}$$

Let us also recall the optimization problem (1)

$$\underset{x \in \mathbb{R}^n}{\text{Minimize}}\, F(x) \text{ s.t. } \|x\|_0 \leq k \tag{10}$$

and the optimization problem (2)

$$\underset{\mathcal{X} \in \mathbb{R}^n}{\text{Minimize}}\, F(H(\mathcal{X})). \tag{11}$$

**Proposition A.1** (Optimization problem equivalence)**.** *For all $m, n, k \in \mathbb{N}$, $A \in \mathbb{R}^{m \times n}$ and $y \in \mathbb{R}^m$. Problem* (10) *is equivalent to problem* (11)*, in the sense that*

1. *for any solution $\mathcal{X}^* \in \text{argmin}_{\mathcal{X} \in \mathbb{R}^n} F(H(\mathcal{X}))$ of* (11)*, $H(\mathcal{X}^*)$ is solution of* (10)*.*
2. *for any minimizer $x'$ of* (10)*, we have $x' \in \text{argmin}_{\mathcal{X} \in \mathbb{R}^n} F(H(\mathcal{X}))$, i.e., $x'$ is solution of* (11)*.*

*Proof.* To establish the first item, we consider a solution $\mathcal{X}^* \in \text{argmin}_{\mathcal{X} \in \mathbb{R}^n} F(H(\mathcal{X}))$ of (11). By definition of $H$, in (9), $H(\mathcal{X}^*)$ is $k$-sparse. To prove that it minimizes (10), consider $x \in \mathbb{R}^n$ such that $\|x\|_0 \leq k$, we have

$$F(H(\mathcal{X}^*)) \leq F(H(x)) \leq F(x), \tag{12}$$

where the first inequality is due to the hypothesis on $\mathcal{X}^*$, and the last inequality to the definition of $H$. Finally, since $H(\mathcal{X}^*)$ is $k$-sparse and (12) holds for all $k$-sparse vector $x$, we conclude that $H(\mathcal{X}^*)$ is solution of (10).

To prove the second item, consider a minimizer $x'$ of (10) and $\mathcal{X} \in \mathbb{R}^n$. By definition of $H$, $H(\mathcal{X})$ is $k$-sparse. Using that $x'$ is solution of (10), we therefore have

$$F(x') \leq F(H(\mathcal{X})). \tag{13}$$

Moreover, since $x'$ is $k$-sparse, we have $\text{supp}(x') \subseteq \text{largest}_k(x')$, and by the definition of $H$,

$$F(H(x')) \leq F(x').$$

Combining with (13), we obtain $F(H(x')) \leq F(H(\mathcal{X}))$, for all $\mathcal{X} \in \mathbb{R}^n$, and conclude that $x'$ is solution of (11).

□

# B Additional algorithms

In this appendix, more details are given about SEA pseudo-code: the main differences with state-of-the-art algorithms HTP and IHT are discussed in Section B.1 and tricks for an efficient implementation of SEA are given in Section B.2.

## B.1 State-of-the-art algorithms

In terms of pseudo-code, SEA looks similar to Hard Thresholding Pursuit (Algorithm 3, [24]) and to a less extent to Iterative Hard Thresholding (Algorithm 4, [8]). In this section , we highlight the differences between these algorithms. In particular, in Algorithm 2, Algorithm 3 and Algorithm 4 distinctions are pointed out in red.

Both HTP and IHT are projected descent algorithms that alternate a gradient step at a sparse estimate $x^t$ and a projection of the resulting variable $\mathcal{X}^t$ onto the set of sparse vectors. The whole difference with SEA lies in the introduction of the support exploration variable $\mathcal{X}^t$ and its interaction

with the sparse vector $x^t$. HTP and IHT perform a regular gradient step $\mathcal{X}^{t+1} \leftarrow x^t - \eta A^T(Ax^t - y)$ (where $\mathcal{X}$ denotes an intermediate variable here, not a support exploration variable) while SEA uses an STE update $\mathcal{X}^{t+1} \leftarrow \mathcal{X}^t - \eta A^T(Ax^t - y)$ of the support exploration variable itself ($\mathcal{X}^t$) with a gradient computed at $x^t$. As a consequence, the vector $\mathcal{X}^t$ in HTP or IHT is always one gradient step away from sparse vector $x^t$. They do not explore much. This is not the case with SEA. The support exploration variable $\mathcal{X}^t$ is not expected to minimize the objective: it rather accumulates all the gradient iterates, where the gradient is computed at $x^t$. This is the whole point of the STE. In particular, $(\mathcal{X}^t)_{t\in\mathbb{N}}$ is not restricted to a small portion of $\mathbb{R}^n$ in the vicinity of sparse vectors. It can explore much more than in HTP and IHT. This explains why SEA has a different exploration/exploitation trade-off. It explores more. As can also be seen from the experiments in Appendix F.5.2, the loss oscillates a lot during SEA's iterative process, but SEA retains the best solution $x^{t_{BEST}}$ encountered during the exploration. SEA is not based on a descent principle as IHT, HTP and such.

Finally, one may also notice that HTP stops as soon as the gradient is small enough such that the support does not change during two successive iterations. On the contrary, SEA keeps accumulating gradients so that the support may remain unchanged for many iterations before a new support is explored. This is clearly visible in the illustrations of Appendix F.5.2.

**Algorithm 2** SEA
(copy of Algorithm 1)

**Inputs:**
noisy observation $y$,
sampling matrix $A$,
sparsity $k$,
step size $\eta$
**Output:** sparse vector $x$
Initialize $\mathcal{X}^0$
$t \leftarrow 0$
**repeat**
  $S^t \leftarrow \text{largest}_k(\mathcal{X}^t)$
  $\begin{cases} x^t_i & \leftarrow 0 \text{ for } i \in \overline{S^t} \\ x^t_{S^t} & \leftarrow A^\dagger_{S^t} y \end{cases}$
  $\mathcal{X}^{t+1} \leftarrow \mathcal{X}^t - \eta A^T(Ax^t - y)$
  $t \leftarrow t + 1$
**until** halting criterion is *true*
$t_{BEST} \leftarrow \underset{t' \in [\![0,t]\!]}{\text{argmin}} \|Ax^{t'} - y\|_2^2$
**return** $x^{t_{BEST}}$

**Algorithm 3** HTP
[24]

**Inputs:**
noisy observation $y$,
sampling matrix $A$,
sparsity $k$,
step size $\eta$
**Output:** sparse vector $x$
Initialize $\mathcal{X}^0$
$t \leftarrow 0$
**repeat**
  $S^t \leftarrow \text{largest}_k(\mathcal{X}^t)$
  $\begin{cases} x^t_i & \leftarrow 0 \text{ for } i \in \overline{S^t} \\ x^t_{S^t} & \leftarrow A^\dagger_{S^t} y \end{cases}$
  $\mathcal{X}^{t+1} \leftarrow x^t - \eta A^T(Ax^t - y)$
  $t \leftarrow t + 1$
**until** halting criterion is *true*
**return** $x^t$

**Algorithm 4** IHT
[8]

**Inputs:**
noisy observation $y$,
sampling matrix $A$,
sparsity $k$,
step size $\eta$
**Output:** sparse vector $x$
Initialize $\mathcal{X}^0$
$t \leftarrow 0$
**repeat**
  $S^t \leftarrow \text{largest}_k(\mathcal{X}^t)$
  $\begin{cases} x^t_i & \leftarrow 0 \text{ for } i \in \overline{S^t} \\ x^t_{S^t} & \leftarrow \mathcal{X}^t_{S^t} \end{cases}$
  $\mathcal{X}^{t+1} \leftarrow x^t - \eta A^T(Ax^t - y)$
  $t \leftarrow t + 1$
**until** halting criterion is *true*
**return** $x^t$

## B.2 Efficient implementation of SEA

Algorithm 1 is presented in a way that favors clarity and simplifies the theoretical analysis. In practice, one can notice that if the support $S^t$ does not change (line 6), then the sparse vector $x^t$ and the gradient $\eta A^T(Ax^t - y)$ do not change either. Algorithm 5 is an equivalent pseudo-code for a computationally-efficient implementation. The most expensive computations —the sparse projection at line 10 and the gradient at line 12— are only required when the support has never been explored before. Also, the best sparse vector can be memorized on the fly (line 16). Hence, the remaining operations, that are performed at each iteration, have a low computational cost: support extraction (line 8), search for a previous, identical support (line 9) and STE update (line 21). This computationally-efficient version of SEA has a larger spatial complexity due to the memorization of all the supports and gradients seen along the iterations. However, this overhead is limited since 1/ for each explored support, only two vectors are memorized, one of them being sparse; and 2/ the number of explored supports is generally much lower than the number of iterations.

In addition, solving the (unconstrained) linear system $A^T_{S^t} A_{S^t} x_{S^t} = A^T_{S^t} y$ can also be performed efficiently. While the pseudo-inverse is a convenient notation at line 10, the solution may be obtained more efficiently, e.g. in $\mathcal{O}(k^2 n)$ to compute $A^T_{S^t} A_{S^t}$ and $A^T_{S^t} y$, and apply the conjugate gradient algorithm. This complexity is a worst-case scenario so in practice, the solution is generally obtained more quickly.

**Algorithm 5** Support Exploration Algorithm: efficient implementation

**Input:** noisy observation $y$, sampling matrix $A$, sparsity $k$, step size $\eta$
**Output:** sparse vector $x$
Initialize $\mathcal{X}^0$
$F^{BEST} \leftarrow +\infty$
$t \leftarrow 0$
$\boldsymbol{\mathcal{S}} \leftarrow \{\}, \boldsymbol{g} \leftarrow \{\}$
**repeat**
  $S \leftarrow \text{largest}_k\left(\mathcal{X}^t\right)$

  {Compute sparse vector and gradient only for unseen supports}
  **if** $S \notin \boldsymbol{\mathcal{S}}$ **then**
    $\begin{cases} x_i^S & \leftarrow 0 \text{ for } i \in \overline{S} \\ x_S^S & \leftarrow A_S^\dagger y \end{cases}$
    $loss^S \leftarrow \frac{1}{2}\|Ax^S - y\|_2^2$
    $g^S \leftarrow \eta A^T\left(Ax^S - y\right)$

    {Memorize support and gradient}
    $\boldsymbol{\mathcal{S}} \leftarrow \boldsymbol{\mathcal{S}} \cup \{S\}$, $\boldsymbol{g} \leftarrow \boldsymbol{g} \cup \left\{g^S\right\}$

    {Memorize best iterate}
    **if** $loss^S < F^{BEST}$ **then**
      $F^{BEST} \leftarrow loss^S$
      $x^{BEST} \leftarrow x^S$
    **end if**
  **else**
    Retrieve $g^S$ in $\boldsymbol{g}$
  **end if**

  {Update support exploration variable}
  $\mathcal{X}^{t+1} \leftarrow \mathcal{X}^t - g^S$
  $t \leftarrow t + 1$
**until** halting criterion is *true*
**return** $x^{BEST}$

# C Proofs and complements of the theoretical analysis

The proof of Theorem 4.1 relies on the fact that when the hypotheses of the theorem are satisfied, the trajectory $(\mathcal{X}^t)_{t\in\mathbb{N}}$ is close to the trajectory of an algorithm that has access to an oracle update. The appendix first contains a description and an analysis of this algorithm in Appendix C.1. Then, in Appendix C.2, we analyze how much the STE-update can deviate from the Oracle Update Rule so that the true support $S^*$ is still recovered. Finally, we prove Theorem 4.1 in Appendix C.3. We prove Corollary 4.2 in Appendix C.4 and conclude with comments on the conditions $(RC_{RIP})$ and $(RC_{SRIP})$ in Appendix C.5.

## C.1 Support Exploration Algorithm using the Oracle Update Rule

The theoretical analysis of SEA and the understanding of the underlying behavior of the algorithm rely on the introduction of an oracle case where the true solution $x^*$and its support $S^*$ are known by the algorithm. In that case, at iteration $t$, we can use the Oracle Update Rule $\mathcal{X}^{t+1} \leftarrow \mathcal{X}^t - u^t$, using the direction $u^t$ defined for any index $i \in [\![1, n]\!]$ by

$$u_i^t = \begin{cases} -\eta x_i^* & i \in S^* \cap \overline{S^t} \\ 0 & i \in \overline{S^*} \cup S^t, \end{cases} \tag{14}$$

where $S^t = \text{largest}_k(\mathcal{X}^t)$ contains the indices of the $k$ largest absolute entries in $\mathcal{X}^t$ and $\eta > 0$ is an arbitrary step size. The resulting pseudo-code is given by Algorithm 6 and we show the important supports in Figure 5.

**Algorithm 6** Support Exploration Algorithm using the Oracle Update Rule

1: **Input:** true solution $x^*$, true support $S^*$, sparsity $k$, step size $\eta$, noisy observation $y$, sampling matrix $A$
2: **Output:** sparse vector $x$
3: Initialize $\mathcal{X}^0$
4: $t \leftarrow 0$
5: **repeat**
6: $\quad S^t \leftarrow \text{largest}_k(\mathcal{X}^t)$
7: $\quad \mathcal{X}^{t+1} \leftarrow \mathcal{X}^t - u^t$
8: $\quad t \leftarrow t + 1$
9: **until** $u^{t-1} = 0$
10: $\begin{cases} x_i & \leftarrow 0 \text{ for } i \in \overline{S^{t-1}} \\ x_{S^{t-1}} & \leftarrow A^\dagger_{S^{t-1}} y \end{cases}$
11: **return** $x$

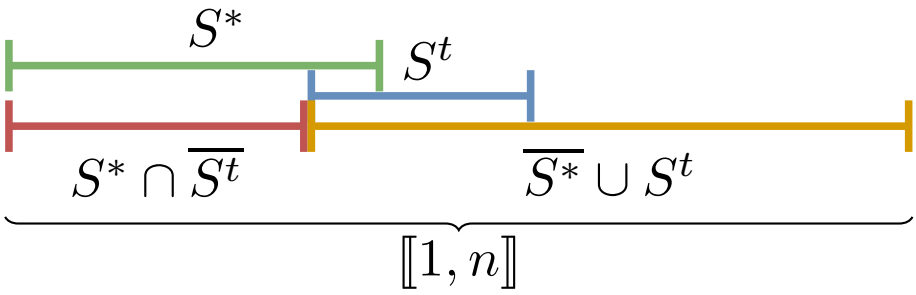



Figure 5: Visual representation of the main sets of indices encountered in the article.

Notice $u_i^t$ is non-zero for indices $i$ from the true support $S^*$ but for which $|\mathcal{X}_i^t|$ is too small to be selected in $S^t$ at line 6. Whatever the initial content of $\mathcal{X}^0$, the oracle update rule always adds the same increment to $\mathcal{X}_i^t$, for $i \in S^* \cap \overline{S^t}$. This guarantees that, at some subsequent iteration $t' \geq t$, the true support $S^*$ is recovered among the $k$ largest absolute entries in $\mathcal{X}^{t'}$, i.e., $S^* \subseteq S^{t'}$, the intersection is empty, $u^{t'} = 0$ and Algorithm 6 stops.

In the following theorem, we formalize this statement and give an upper bound on the number of iterations required by the support exploration algorithm using the Oracle Update Rule.

**Theorem C.1** (Recovery - Oracle Update Rule)**.** *For all matrices $A$, error vectors $e$, initializations $\mathcal{X}^0$ and for all $\eta > 0$, there exists*

$$t_s \leq T_{max}^{oracle} = k\left(1 + \frac{2\|\mathcal{X}^0\|_\infty}{\eta \min_{i \in S^*} |x_i^*|}\right)$$

*such that $S^* \subseteq S^{t_s}$, where $S^t$ is defined in Algorithm 6 line 6.*

*Moreover, $u^{t_s} = 0$ and Algorithm 6 returns* $\underset{\substack{x \in \mathbb{R}^n \\ supp(x) \subseteq S^{t_s}}}{\operatorname{argmin}} \|Ax - y\|_2^2$.

### C.1.1 Proof of Theorem C.1

We denote, for all $t \in \mathbb{N}^*$ and all $i \in [\![1, n]\!]$, and $S^t$ defined in Algorithm 6, line 6

$$c_i^t = |\{t' \in [\![0, t-1]\!] : i \in S^* \cap \overline{S^{t'}}\}|. \tag{15}$$

We extend the definition to $t = 0$ and set, for all $i \in [\![1, n]\!]$, $c_i^0 = 0$.

We can prove by induction on $t$ that, given the definition of $\mathcal{X}^t$ in Algorithm 6 and $u^t$ in (14), for all $t \in \mathbb{N}$,

$$\mathcal{X}^t = \mathcal{X}^0 + \eta\ c^t \odot x^*, \tag{16}$$

where $\odot$ denotes the Hadamard product.

The following lemma states that if $c_i^t$ is large then $i$ is always selected by Algorithm 6.

**Lemma C.2.** *For all $i \in S^*$ and all $t \in \mathbb{N}^*$*

$$\text{if} \quad c_i^t > \frac{2\|\mathcal{X}^0\|_\infty}{\eta |x_i^*|} \quad \text{then} \quad \forall t' \geq t, \quad i \in S^{t'}.$$

*Proof.* Let $i \in S^*$ and $t \in \mathbb{N}^*$ be such that $c_i^t > \frac{2\|\mathcal{X}^0\|_\infty}{\eta |x_i^*|}$. Consider $t' \geq t$.

Since $t \mapsto c_i^t$ is non-decreasing, we have

$$c_i^{t'} \geq c_i^t > \frac{2\|\mathcal{X}^0\|_\infty}{\eta |x_i^*|}.$$

Therefore, for all $j \in \overline{S^*}$,

$$|\mathcal{X}_i^{t'}| = |\mathcal{X}_i^0 + \eta c_i^{t'} x_i^*| \geq |\eta c_i^{t'} x_i^*| - |\mathcal{X}_i^0| > 2\|\mathcal{X}^0\|_\infty - |\mathcal{X}_i^0| \geq \|\mathcal{X}^0\|_\infty \geq |\mathcal{X}_j^0| = |\mathcal{X}_j^{t'}|.$$

Therefore $|\mathcal{X}_i^{t'}|$ is larger than at least $n - k$ elements of $\{|\mathcal{X}_j^{t'}| : j \in [\![1, n]\!]\}$. Said differently, $i \in \text{largest}_k(\mathcal{X}^{t'}) = S^{t'}$.

This concludes the proof of Lemma C.2. □

This leads to the following upper bound.

**Lemma C.3.** *For all $i \in S^*$ and all $t \in \mathbb{N}^*$*

$$c_i^t \leq \frac{2\|\mathcal{X}^0\|_\infty}{\eta |x_i^*|} + 1.$$

*Proof.* If Lemma C.3 is false, there exists $i \in S^*$ and $t \in \mathbb{N}^*$ such that $c_i^t > \frac{2\|\mathcal{X}^0\|_\infty}{\eta |x_i^*|} + 1$.

We denote

$$t' = \min\{t \in \mathbb{N}^* : c_i^t > \frac{2\|\mathcal{X}^0\|_\infty}{\eta |x_i^*|} + 1\}.$$

We have $c_i^{t'} > \frac{2\|\mathcal{X}^0\|_\infty}{\eta |x_i^*|} + 1 \geq 1$ and therefore $t' > 1$. As a consequence, $t' - 1 \in \mathbb{N}^*$ and $c_i^{t'-1}$ is defined by (15). Because of the definitions of $t'$ and $c_i^{t'}$, we must have $c_i^{t'} = c_i^{t'-1} + 1$. Therefore, $c_i^{t'-1} > \frac{2\|\mathcal{X}^0\|_\infty}{\eta |x_i^*|}$. Since $i \in S^*$ and $t' - 1 \in \mathbb{N}^*$, using Lemma C.2, we conclude that $i \in S^{t'-1}$ and, using the definition of $c_i^{t'}$ in (15), that $c_i^{t'} = c_i^{t'-1}$.

This is impossible and we conclude that Lemma C.3 holds. □

**Proof of Theorem C.1:** We denote

$$t_s = \min\{t \in \mathbb{N} : S^* \subseteq S^t\}. \tag{17}$$

By convention, if for all $t \in \mathbb{N}$, $S^* \not\subseteq S^t$, we set $t_s = +\infty$. The first statement of Theorem C.1 is obvious if $t_s = 0$. We assume below that $t_s \geq 1$.

Consider $t \in [\![0, t_s - 1]\!]$, using of the definition of $t_s$ in (17), there exists $i \in S^* \cap \overline{S^t}$. Using the definition of $c_i^{t+1}$ in (15), we obtain $c_i^{t+1} = c_i^t + 1$. Since for all $j \in [\![1, n]\!]$, $t \mapsto c_j^t$ is non-decreasing, we conclude that

$$\text{for all } t \in [\![0, t_s - 1]\!], \qquad \sum_{i \in S^*} c_i^{t+1} \geq \sum_{i \in S^*} c_i^t + 1.$$

We therefore obtain

$$\begin{aligned}
\sum_{i \in S^*} c_i^{t_s} &= \sum_{i \in S^*} \left( \sum_{i=0}^{t_s - 1} (c_i^{t+1} - c_i^t) \right) \\
&= \sum_{i=0}^{t_s - 1} \left( (\sum_{i \in S^*} c_i^{t+1}) - (\sum_{i \in S^*} c_i^t) \right) \\
&\geq \sum_{i=0}^{t_s - 1} 1 = t_s.
\end{aligned}$$

Using Lemma C.3, we obtain

$$t_s \leq \sum_{i \in S^*} c_i^{t_s} \leq \sum_{i \in S^*} \left( \frac{2\|\mathcal{X}^0\|_\infty}{\eta |x_i^*|} + 1 \right) \leq k \left( 1 + \frac{2\|\mathcal{X}^0\|_\infty}{\eta \min_{i \in S^*} |x_i^*|} \right).$$

To conclude the proof, we simply remark that, since $S^* \subseteq S^{t_s}$, by definition of $u^{t_s}$ in (14),

$$u^{t_s} = 0.$$

□

Since $x^*$ and $S^*$ are not available in practice, we replace in Algorithm 6 the oracle update $u^t$ by the surrogate $\eta A^T(Ax^t - y)$ (line 8). The choice of this surrogate is an application of STE and is natural. For instance, one can show that $u^t = \eta A^T(Ax^t - y)$ in the simple case where $A$ has orthonormal columns and the observation is noiseless (see Corollary D.1 and Lemma D.2 in Appendix D). In the general setting, SEA can be interpreted as a noisy version of the support exploration algorithm using the Oracle update. Theorem 4.1 and its proof in Appendix C are based on the fact that $u^t - \eta A^T(Ax^t - y)$ is small, under suitable hypotheses on $x^*$ and the RIP constants of $A$.

## C.2 If the STE-update is sufficiently close to the Oracle update, SEA visits $S^*$

To prove Theorem 4.1, we first provide a general recovery theorem here. The theorem states that if the discrepancy between the Oracle update and the STE update is sufficiently small, SEA visits $S^*$.

To do so, we define, for all $t \in \mathbb{N}$, the gradient noise: $b^t \in \mathbb{R}^n$ as

$$b^t = u^t - \eta A^T(Ax^t - y). \tag{18}$$

We define the maximal gradient noise norm

$$\varepsilon = \sup_{t \in \mathbb{N}} \|b^t\|_\infty \in \mathbb{R}. \tag{19}$$

We define the Recovery Condition ($RC$) as

$$\varepsilon < \frac{\eta}{2k} \min_{i \in S^*} |x_i^*|. \tag{$RC$}$$

**Theorem C.4** (Recovery - General case)**.** *If (RC) holds, then for all initializations $\mathcal{X}^0$ and all $\eta > 0$, there exists $t_s \leq T_{max}$ such that $S^* \subseteq S^{t_s}$, where $S^t$ is defined in Algorithm 1 line 6 and*

$$T_{max} = \frac{2k\|\mathcal{X}^0\|_\infty + (k+1)\eta \min_{i \in S^*} |x_i^*|}{\eta \min_{i \in S^*} |x_i^*| - 2k\varepsilon}. \tag{20}$$

The proof is in Appendix C.2.1, right after the comments below.

The main interest of Theorem C.4 is to formalize quantitatively that, when $u^t - \eta A^T(Ax^t - y)$ is sufficiently small, SEA visits the correct support. However, the condition $(RC)$ is difficult to use and interpret since it involves both $A$, $x^*$, and all the sparse iterates $x^t$. This is why we provide in Corollary D.1, Theorem 4.1 and Corollary 4.2 sufficient conditions on $A$, $e$ and $x^*$ guaranteeing that $(RC)$ holds.

The conclusion of Theorem C.4 is that the iterative process of SEA visits the correct support at some iteration $t$. We have in general no guarantee that this time-step $t$ is equal to $t_{BEST}$. We are however guaranteed that SEA returns a sparse solution such that $\|Ax^{t_{BEST}} - y\|_2 \leq \|Ax^{t_s} - y\|_2 \leq \|Ax^* - y\|_2$. In machine learning, this upper bound can be used to derive an upper bound of the risk.

Concerning the value of $T_{max}$, a quick analysis shows that $T_{max}$ increases with $\varepsilon$, when $(RC)$ holds. In other words, the number of iterations required by the algorithm to provide the correct solution increases with the discrepancy between $u^t$ and $\eta A^T(Ax^t - y)$.

The initializations $\mathcal{X}^0 \neq 0$ have an apparent negative impact on the number of iterations required in the worst case. This is because in the worst-case $\mathcal{X}^0$ would be poorly chosen and SEA needs iterations to correct this poor choice.

Concerning $\eta$, notice that, since $u^t$ is proportional to $\eta > 0$, $\varepsilon$ is proportional to $\eta > 0$ and therefore $(RC)$ is independent of $\eta$. When possible, any $\eta$ permits the recovery of $S^*$. The only influence of $\eta$ is on $T_{max}$. In this regard, since $\varepsilon$ is proportional to $\eta > 0$, the denominator of (20) is proportional to $\eta$. In the numerator, we see that the larger $\eta$ is, the faster SEA will override the initialization $\mathcal{X}^0$. The choice of $\eta$ is very much related to the question of the quality of the initialization discussed above.

### C.2.1 Proof of Theorem C.4

To prove Theorem C.4, we need to adapt a closed formula for the exploratory variable $\mathcal{X}^t$ already encountered in the proof of Theorem C.1. Then, we will study the properties of this closed formula through the counting vector $c^t$ in Appendix C.2.2. to find a sufficient condition of support recovery. Then we prove Theorem C.4 in Appendix C.2.3.

### C.2.2 Preliminaries

We remind Figure 5 on which the mains supports are represented and we remind, for each iteration $t \in \mathbb{N}$ and $i \in [\![1, n]\!]$, the Oracle Update already defined in (14)

$$u_i^t = \begin{cases} -\eta x_i^* & i \in S^* \cap \overline{S^t} \\ 0 & i \in \overline{S^*} \cup S^t. \end{cases}$$

We also remind the gradient noise, already defined in (18), $b^t = u^t - \eta A^T(Ax^t - y)$.

We first remark that, for any $i \in S^t$,

$$b_i^t = 0. \tag{21}$$

To prove this equality, we remark that, for all $i \in S^t$, $u_i^t = 0$ and prove that $(A^T(Ax^t - y))_i = 0$. Indeed, the latter holds because $i \in S^t$ and $x^t = H(\mathcal{X}^t) = \operatorname{argmin}_{\operatorname{supp}(x) \subseteq S^t} \frac{1}{2}\|Ax - y\|_2^2$ (see (3) and Algorithm 1, line 7). If we denote $F(x) = \frac{1}{2}\|Ax - y\|_2^2$, the Karush-Kuhn-Tucker equations of the latter constrained optimization problem require $\nabla F(x^t)$ to be in the orthogonal complement of the vector space $\{x \in \mathbb{R}^n : \operatorname{supp}(x) \subset S^t\}$. That is $(\nabla F(x^t))_i = (A^T(Ax^t - y))_i = 0$, for all $i \in S^t$. This concludes the proof of (21).

As a consequence of the definition of $b^t$ and SEA, line 8, for any $t \in \mathbb{N}$,

$$\mathcal{X}^{t+1} = \mathcal{X}^t + b^t - u^t. \tag{22}$$

The gradient noise $b^t$ is the error preventing the gradient from being in the direction of the oracle update $u^t$. At each iteration, this error is accumulating in $\mathcal{X}^t$. With $\beta^0 = 0$, for any $t \in \mathbb{N}^*$, we define this accumulated error by

$$\beta^t = \sum_{t'=0}^{t-1} b^{t'} \in \mathbb{R}^n. \tag{23}$$

As already done in the proof of Theorem C.1 for the support sequence defined in Algorithm 6, we define counting vectors. However, this time they are defined for the sequence defined in Algorithm 1. We keep

the same notations for simplicity. We set $c^0 = 0$, for any $t \in \mathbb{N}^*$ and $i \in [\![1, n]\!]$, we also define the counting vector by

$$c_i^t = |\{t' \in [\![0, t-1]\!] : i \in S^* \cap \overline{S^{t'}}\}|. \tag{24}$$

We will use the recursive formula for $c^t$: For any $t \in \mathbb{N}$, $i \in [\![1, n]\!]$

$$c_i^{t+1} = \begin{cases} c_i^t + 1 & \text{if } i \in S^* \cap \overline{S^t} \\ c_i^t & \text{if } i \in \overline{S^*} \cup S^t. \end{cases} \tag{25}$$

For any $i \in [\![1, n]\!]$, the sequence $(c_i^t)_{t\in\mathbb{N}}$ is non-decreasing.
Using (22), (23) and (24), we can establish by induction on $t$ that for any $t \in \mathbb{N}$,

$$\mathcal{X}^t = \mathcal{X}^0 + \eta c^t \odot x^* + \beta^t, \tag{26}$$

where $\odot$ denotes the Hadamard product. This generalizes (16) to the noisy setting.

As can be seen from (26), the error accumulation $\beta^t$ is responsible for the exploration in the wrong directions. While $c^t \odot x^*$ encourages exploration in the direction of the missed components of $x^*$. Below, we provide important properties of $(c^t)_{t\in\mathbb{N}}$.

At each iteration of SEA, using (25) when $S^* \nsubseteq S^t$, there exists at least one $i \in S^*$ such that $c_i^{t+1} = c_i^t + 1$. Using also that, for all $i \in S^*$, $(c_i^t)_{t\in\mathbb{N}}$ is non-decreasing we obtain

$$\text{for all } t \in \mathbb{N} \text{ such that } S^* \nsubseteq S^t, \qquad \sum_{i\in S^*} c_i^{t+1} \geq \big(\sum_{i\in S^*} c_i^t\big) + 1 \tag{27}$$

We define the first recovery iterate[9] $t_s$ as the smallest iteration $t$ such that $S^* \subseteq S^t$. More precisely,

$$t_s = \min\{t,\, S^* \subseteq S^t\} \in \mathbb{N}. \tag{28}$$

By convention, if $S^*$ is never recovered, $t_s = +\infty$. By induction on $t$, using (27), we obtain a lower bound on $\sum_{i\in S^*} c_i^t$:

$$\text{For all } t \leq t_s, \qquad \sum_{i\in S^*} c_i^t \geq t. \tag{29}$$

Let us now upper bound $\sum_{i\in S^*} c_i^t$. We first remind the definition of $\varepsilon$ in (19). We define the sharp Recovery Condition

$$\varepsilon < \frac{1}{2\sum_{i\in S^*} \frac{1}{\eta|x_i^*|}} \tag{RC'}$$

and

$$T'_{max} = \frac{\sum_{i\in S^*} \frac{\max_{j\notin S^*}|\mathcal{X}_j^0| + |\mathcal{X}_i^0|}{\eta|x_i^*|} + k + 1}{1 - 2\varepsilon\sum_{i\in S^*} \frac{1}{\eta|x_i^*|}}. \tag{30}$$

If $(RC')$ holds, we define for any $i \in S^*$, the $i^{th}$ counting threshold by

$$C_i = \frac{\max_{j\notin S^*}|\mathcal{X}_j^0| + |\mathcal{X}_i^0| + 2T'_{max}\varepsilon}{\eta|x_i^*|}. \tag{31}$$

**Proposition C.5** (Upper bound)**.** *If (RC') holds, for any $i \in S^*$ and any $t \leq T'_{max}$, we have $c_i^t \leq C_i + 1$.*

*Proof.* Assume $(RC')$ holds. We have $T'_{max} > 0$. Let $i \in S^*$, we distinguish two cases:
$\underline{1^{st}\ case}$: If for all $t \leq T'_{max}$, $c_i^t \leq C_i$: Then, obviously, for any $t \leq T'_{max}$, $c_i^t \leq C_i + 1$.
$\underline{2^{nd}\ case}$: If there exists $t \leq T'_{max}$, such that $c_i^t > C_i$:
We define $t_i = \min\{t \in \mathbb{N} :\, c_i^t > C_i\}$. We have $t_i \leq T'_{max}$. The proof follows two steps:

1. We will prove that for all $t \in [\![t_i, T'_{max}]\!]$, $c_i^t = c_i^{t_i}$. (32)
2. We will prove that for all $t \leq T'_{max}$, $c_i^t \leq C_i + 1$. (33)

[9]Again, a similar quantity is defined in the proof of Theorem C.1 for the supports $S^t$ defined in Algorithm 6. We use the same notation although this time the quantity is defined for the sets $S^t$ defined in Algorithm 1. It should not be ambiguous since the notations are used in different proofs and sections.

1. Let $t \in [\![t_i, T'_{max}]\!]$, we have, using (26), the triangle inequality and the fact that $c_i^t \geq c_i^{t_i} > C_i$

$$\begin{aligned}|\mathcal{X}_i^t| &= |\mathcal{X}_i^0 + \eta c_i^t x_i^* + \beta_i^t| \\ &\geq \eta c_i^t |x_i^*| - |\mathcal{X}_i^0| - |\beta_i^t| \\ &> \eta C_i |x_i^*| - |\mathcal{X}_i^0| - |\beta_i^t|.\end{aligned}$$

Using the definition of $C_i$, in (31), we obtain

$$\begin{aligned}|\mathcal{X}_i^t| &> \eta \frac{\max_{j \notin S^*} |\mathcal{X}_j^0| + |\mathcal{X}_i^0| + 2T'_{max}\varepsilon}{\eta |x_i^*|} |x_i^*| - |\mathcal{X}_i^0| - |\beta_i^t| \\ &= \max_{j \notin S^*} |\mathcal{X}_j^0| + 2T'_{max}\varepsilon - |\beta_i^t|.\end{aligned}$$

Since for any $j \in [\![1, n]\!]$, $|\beta_j^t| \leq \sum_{t'=0}^{t-1} |b_j^{t'}| \leq t\varepsilon \leq T'_{max}\varepsilon$, we have

$$\begin{aligned}|\mathcal{X}_i^t| &> \max_{j \notin S^*} |\mathcal{X}_j^0| + \max_{j \notin S^*} |\beta_j^t| + |\beta_i^t| - |\beta_i^t| \\ &\geq \max_{j \notin S^*} |\mathcal{X}_j^0 + \beta_j^t| \\ &= \max_{j \notin S^*} |\mathcal{X}_j^t|, \end{aligned} \tag{34}$$

where the last equality holds because of (26) and for all $j \notin S^*$, all $t \in \mathbb{N}$, $c_j^t = 0$.

Equation (34) implies that $|\mathcal{X}_i^t|$ is larger than $|\{j \notin S^*\}|$ elements of $\{|\mathcal{X}_j^t| \mid j \in [\![1, n]\!]\}$ and, since $|S^*| \leq k$, we have $|\{j \notin S^*\}| = n - |S^*| \geq n - k$. Finally, $|\mathcal{X}_i^t|$ is larger than $n - k$ elements of $\{|\mathcal{X}_j^t| \mid j \in [\![1, n]\!]\}$ and $i \in \text{largest}_k(\mathcal{X}^t) = S^t$.

As a conclusion, for all $t \in [\![t_i, T'_{max}]\!]$, $i \in S^t$. Using (25) , this leads to $c_i^{t+1} = c_i^t$. Therefore, for all $t \in [\![t_i, T'_{max} + 1]\!]$, $c_i^t = c_i^{t_i}$. This concludes the proof of the first step.

2. Since $t_i = \min\{t \in \mathbb{N} : c_i^t > C_i\}$ and since $c_i^0 = 0$, $t_i \geq 1$. Since by definition of $t_i$, $c_i^{t_i - 1} \leq C_i$ and $c_i^{t_i} \neq c_i^{t_i - 1}$; we find that $c_i^{t_i} = c_i^{t_i - 1} + 1 \leq C_i + 1$.

Using (32) , for all $t \in [\![t_i, T'_{max}]\!]$, $c_i^t = c_i^{t_i} \leq C_i + 1$. Finally, since $(c_i^t)_{t \in \mathbb{N}^*}$ is non-decreasing, it follows that for any $t \leq t_i - 1$, $c_i^t \leq c_i^{t_i - 1} \leq C_i$. This concludes the proof of (33).

□

### C.2.3 Proof of Theorem C.4

To prove Theorem C.4, we first prove a sharper, but difficult-to-interpret theorem.

**Theorem C.6** (Recovery - General case)**.** *If (RC') holds, then for all initializations $\mathcal{X}^0$ and all $\eta > 0$, there exists $t_s \leq T'_{max}$ such that $S^* \subseteq S^{t_s}$, where $S^t$ is defined in Algorithm 1 line 6 and*

$$T'_{max} = \frac{\sum_{i \in S^*} \frac{\max_{j \notin S^*} |\mathcal{X}_j^0| + |\mathcal{X}_i^0|}{\eta |x_i^*|} + k + 1}{1 - 2\varepsilon \sum_{i \in S^*} \frac{1}{\eta |x_i^*|}}.$$

*Proof.* We assume ($RC'$) holds and prove Theorem C.6 using the results of Appendix C.2.2.

In order to do this, we first show that $T'_{max} = \sum_{i \in S^*} C_i + k + 1$, then we demonstrate that $t_s \leq T'_{max}$.

Since ($RC'$) holds, using the definition of $T'_{max}$, we calculate

$$\begin{aligned}T'_{max} &= \frac{1}{1 - 2\varepsilon \sum_{i \in S^*} \frac{1}{\eta |x_i^*|}} \left( \sum_{i \in S^*} \frac{\max_{j \notin S^*} |\mathcal{X}_j^0| + |\mathcal{X}_i^0|}{\eta |x_i^*|} + k + 1 \right) \\ \left(1 - 2\varepsilon \sum_{i \in S^*} \frac{1}{\eta |x_i^*|}\right) T'_{max} &= \sum_{i \in S^*} \frac{\max_{j \notin S^*} |\mathcal{X}_j^0| + |\mathcal{X}_i^0|}{\eta |x_i^*|} + k + 1 \\ T'_{max} &= \sum_{i \in S^*} \frac{\max_{j \notin S^*} |\mathcal{X}_j^0| + |\mathcal{X}_i^0|}{\eta |x_i^*|} + k + 1 + 2T'_{max}\varepsilon \sum_{i \in S^*} \frac{1}{\eta |x_i^*|} \\ T'_{max} &= \sum_{i \in S^*} \frac{\max_{j \notin S^*} |\mathcal{X}_j^0| + |\mathcal{X}_i^0| + 2T'_{max}\varepsilon}{\eta |x_i^*|} + k + 1.\end{aligned}$$

Using (31), we obtain $T'_{max} = \sum_{i\in S^*} C_i + k + 1$.

We finally prove Theorem C.4 by contradiction. Assume by contradiction that $t_s > T'_{max}$, where $t_s$ is defined in (28). Using (29) with $t = \lfloor T'_{max} \rfloor < t_s$, we have

$$\sum_{i\in S^*} c_i^{\lfloor T'_{max}\rfloor} \geq \lfloor T'_{max} \rfloor = \lfloor \sum_{i\in S^*} C_i + k + 1 \rfloor > \sum_{i\in S^*} C_i + k. \tag{35}$$

However, using $|S^*| \leq k$ and Proposition C.5 for $t = \lfloor T'_{max} \rfloor$ , we find

$$\sum_{i\in S^*} C_i + k \geq \sum_{i\in S^*} (C_i + 1) \geq \sum_{i\in S^*} c_i^{\lfloor T'_{max}\rfloor}$$

This contradicts (35) and we can conclude that $t_s \leq T'_{max}$. This proves Theorem C.6. □

**Proof of Theorem C.4:**

If $(RC)$ holds, that is $\varepsilon < \frac{\eta}{2k} \min_{i\in S^*} |x_i^*|$, since $\sum_{i\in S^*} \frac{1}{|x_i^*|} \leq \frac{k}{\min_{i\in S^*} |x_i^*|}$, we have

$$\varepsilon < \frac{1}{2\sum_{i\in S^*} \frac{1}{\eta |x_i^*|}}.$$

Therefore, $(RC')$ holds, and we can apply Theorem C.6. It ensures that for all initializations $\mathcal{X}^0$ and all $\eta > 0$, there exists $t_s \leq T'_{max}$ such that $S^* \subseteq S^{t_s}$.

To prove Theorem C.4, it suffices to prove that $T'_{max} \leq T_{max}$. Using $\sum_{i\in S^*} \frac{1}{|x_i^*|} \leq \frac{k}{\min_{i\in S^*} |x_i^*|}$, we obtain

$$1 - 2\varepsilon \sum_{i\in S^*} \frac{1}{\eta |x_i^*|} \geq 1 - 2\varepsilon \frac{k}{\eta \min_{i\in S^*} |x_i^*|}$$

and using $\min_{j\notin S^*} |\mathcal{X}_j^0| + |\mathcal{X}_i^0| \leq 2\|\mathcal{X}^0\|_\infty$ we have

$$T'_{max} = \frac{\sum_{i\in S^*} \frac{\max_{j\notin S^*} |\mathcal{X}_j^0| + |\mathcal{X}_i^0|}{\eta |x_i^*|} + k + 1}{1 - 2\varepsilon \sum_{i\in S^*} \frac{1}{\eta |x_i^*|}} \leq \frac{\frac{2k\|\mathcal{X}^0\|_\infty}{\eta \min_{i\in S^*} |x_i^*|} + k + 1}{1 - 2\varepsilon \frac{k}{\eta \min_{i\in S^*} |x_i^*|}} = T_{max}.$$

## C.3 Proof of Theorem 4.1

We remind in Appendix C.3.1 known properties of RIP matrices. We bound in Appendix C.3.2 the error made when approximating $x^*$ on a specific support $S$. This permits us to bound $b^t$ and apply Theorem C.4 to prove Theorem 4.1 in Appendix C.3.3. We finally apply Theorem 4.1 in Appendix C.4 to prove Corollary 4.2. Before that and to illustrate and quantify that the STE-update $\mathcal{X}^{t+1} \leftarrow \mathcal{X}^t - \eta A^T(Ax^t - y)$ is a noisy version of the Oracle update $\mathcal{X}^{t+1} \leftarrow \mathcal{X}^t - u^t$, we provide in the following theorem an upper bound on the discrepancy between the two updates. This bound is pivotal in the proof of Theorem 4.1. The statement of Theorem 4.1 is, up to the additional upper-bound (36), the same as the statement of the following theorem, which we prove in this section.

**Theorem C.7** (Recovery - RIP case)**.** *Assume $A$ satisfies the $(2k+1)$-RIP and for all $i \in [\![1, n]\!]$, $\|A_i\|_2 = 1$. Then, for all $t \in \mathbb{N}$,*

$$\|\frac{u^t}{\eta} - A^T(Ax^t - y)\|_\infty \leq \alpha_k^{RIP} \|x^*\|_2 + \gamma_k^{RIP} \|e\|_2. \tag{36}$$

*If moreover $x^*$ satisfies $(RC_{RIP})$, then for all initializations $\mathcal{X}^0$ and all $\eta > 0$, there exists $t_s \leq T_{RIP}$ such that $S^* \subseteq S^{t_s}$, where*

$$T_{RIP} = \frac{2k\frac{\|\mathcal{X}^0\|_\infty}{\eta} + (k+1)\min_{i\in S^*} |x_i^*|}{\min_{i\in S^*} |x_i^*| - 2k\left(\alpha_k^{RIP}\|x^*\|_2 + \gamma_k^{RIP}\|e\|_2\right)}. \tag{37}$$

*If moreover, $x^*$ is such that*

$$\min_{i\in S^*} |x_i^*| > \frac{2}{\sqrt{1-\delta_{2k}}} \|e\|_2 \tag{38}$$

*and SEA performs more than $T_{RIP}$ iterations, then $S^* \subseteq S^{t_{BEST}}$ and $\|x^{t_{BEST}} - x^*\|_2 \leq \frac{2}{\sqrt{1-\delta_k}} \|e\|_2$.*

### C.3.1 Reminders on properties of RIP matrices

We first remind the definition of Restricted Isometry Constant in (4) and a few properties of RIP matrices.

**Fact 1:** For any $k, k' \in [\![1, n]\!]$, such that $k \leq k'$, we have

$$\delta_k \leq \delta_{k'}. \tag{39}$$

**Fact 2:** For any $R, S \subseteq [\![1, n]\!]$, such that $R \cap S = \emptyset$ and $A$ satisfies the $(|R| + |S|)$-RIP. We remind Lemma 1 of [15] (see also [11]): For any $x \in \mathbb{R}^{|S|}$

$$\|A_R^T A_S x\|_2 \leq \delta_{|R|+|S|} \, \|x\|_2. \tag{40}$$

For completeness, we prove this inequality below. Let $A$, $R$, $S$ and $x$ be as above, we have,

$$\|A_R^T A_S \frac{x}{\|x\|_2}\|_2 = \max_{x' : \|x'\|_2 = 1} \langle x', A_R^T A_S \frac{x}{\|x\|_2} \rangle.$$

Using $\langle x', A_R^T A_S \frac{x}{\|x\|_2} \rangle = \langle A_R x', A_S \frac{x}{\|x\|_2} \rangle \leq \frac{1}{2} \|A_R x' + A_S \frac{x}{\|x\|_2}\|_2^2$, the fact that $R \cap S = \emptyset$, and that $A$ satisfies the $(|R| + |S|)$-RIP defined in (4), we obtain for all $x' \in \mathbb{R}^{|R|}$ such that $\|x'\|_2 = 1$

$$\langle x', A_R^T A_S \frac{x}{\|x\|_2} \rangle \leq \frac{1}{2}(1 + \delta_{|R|+|S|}) \left( \|x'\|_2^2 + \| \frac{x}{\|x\|_2} \|_2^2 \right) = (1 + \delta_{|R|+|S|}).$$

This concludes the proof of (40).

**Fact 3:** Let us assume that $A$ satisfies the $|S|$-RIP. Taking inspiration of Proposition 3.1 of [37], for any singular value $\lambda \in \mathbb{R}$ of $A_S$, and the corresponding right singular vector $x_\lambda \in \mathbb{R}^{|S|}$, we have $\|A_S x_\lambda\|_2 = \lambda$. Using (4), $1 - \delta_{|S|} \leq \lambda^2 \leq 1 + \delta_{|S|}$. All singular values of $A_S$ and $A_S^T$ lie between $\sqrt{1 - \delta_{|S|}}$ and $\sqrt{1 + \delta_{|S|}}$.

As a consequence, for any $u \in \mathbb{R}^m$, we have

$$\|A_S^T u\|_2 \leq \sqrt{1 + \delta_{|S|}} \, \|u\|_2. \tag{41}$$

**Fact 4:** Let us assume that $A$ satisfies the $|S|$-RIP. Using the same reasoning, we find that the eigenvalues of $A_S^T A_S$ lie between $1 - \delta_{|S|}$ and $1 + \delta_{|S|}$. This implies that $A_S^T A_S$ is non-singular and that the eigenvalues of $(A_S^T A_S)^{-1}$ lie between $\frac{1}{1+\delta_{|S|}}$ and $\frac{1}{1-\delta_{|S|}}$. Then $A_S$ is full column rank and for any $x \in \mathbb{R}^{|S|}$

$$\|(A_S^T A_S)^{-1} x\|_2 \leq \frac{1}{1 - \delta_{|S|}} \|x\|_2. \tag{42}$$

**Fact 5:** Let us assume that $A$ satisfies the $|S|$-RIP. By using one last time the same reasoning, we find that the eigenvalues of $A_S^T A_S - I_{|S|}$ lie between $-\delta_{|S|}$ and $\delta_{|S|}$. Finally, for any $x \in \mathbb{R}^{|S|}$,

$$\|(A_S^T A_S - I_{|S|})x\|_2 \leq \delta_{|S|} \|x\|_2. \tag{43}$$

### C.3.2 Preliminaries

In this section, the facts from Appendix C.3.1 are used to bound from above the error $\|(x^t - x^*)_{|S^t}\|_2$, where $(.)_{|S^t}$ is the restriction of the vector to the support $S^t$ and $S^t$ is defined in Algorithm 1, line 6. This bound will lead to an upper bound on $\|b^t\|_2$. Throughout the section, we assume $A$ satisfies the $(2k + 1)$-RIP. Figure 5 might help visualize the different sets of indices considered in the proof.

**Lemma C.8.** *If $A$ satisfies the $(2k + 1)$-RIP, for any $t \in \mathbb{N}$,*

$$\|(x^t - x^*)_{|S^t}\|_2 \leq \frac{\delta_{2k}}{1 - \delta_k} \frac{\|u^t\|_2}{\eta} + \frac{\sqrt{1 + \delta_k}}{1 - \delta_k} \|e\|_2.$$

*Proof.* For any $t \in \mathbb{N}$, using the definition of $x^t$ in Algorithm 1 and (14), we find

$$\begin{aligned} x^t_{|S^t} &= A^\dagger_{S^t} y \\ &= A^\dagger_{S^t}(A_{S^*} x^*_{|S^*} + e) \\ &= A^\dagger_{S^t} A_{S^* \cap S^t} x^*_{|S^* \cap S^t} - \frac{1}{\eta} A^\dagger_{S^t} A_{S^* \cap \overline{S^t}} u^t_{|S^* \cap \overline{S^t}} + A^\dagger_{S^t} e. \end{aligned} \tag{44}$$

We also have

$$\begin{aligned} A^\dagger_{S^t} A_{S^* \cap S^t} x^*_{|S^* \cap S^t} &= A^\dagger_{S^t} \begin{bmatrix} A_{S^* \cap S^t} & A_{S^t \setminus S^*} \end{bmatrix} \begin{bmatrix} x^*_{|S^* \cap S^t} \\ 0 \end{bmatrix} \\ &= A^\dagger_{S^t} A_{S^t} x^*_{|S^t}. \end{aligned} \tag{45}$$

Since $\delta_{2k+1} < 1$, (39) implies that $\delta_k \leq \delta_{2k+1} < 1$ and the smallest singular value of $A_{S^t}$ is larger than $\sqrt{1-\delta_k} \geq \sqrt{1-\delta_{2k+1}} > 0$. Therefore $A_{S^t}$ is full column rank and

$$A^\dagger_{S^t} = (A^T_{S^t} A_{S^t})^{-1} A^T_{S^t}. \tag{46}$$

Combining (44), (45) and (46), we obtain

$$x^t_{|S^t} = x^*_{|S^t} - \frac{1}{\eta} A^\dagger_{S^t} A_{S^* \cap \overline{S^t}} u^t_{|S^* \cap \overline{S^t}} + A^\dagger_{S^t} e.$$

Using (46), we find

$$\begin{aligned} \|(x^t - x^*)_{|S^t}\|_2 &= \|\frac{1}{\eta} A^\dagger_{S^t} A_{S^* \cap \overline{S^t}} u^t_{|S^* \cap \overline{S^t}} - A^\dagger_{S^t} e\|_2 \\ &\leq \frac{1}{\eta} \|(A^T_{S^t} A_{S^t})^{-1} A^T_{S^t} A_{S^* \cap \overline{S^t}} u^t_{|S^* \cap \overline{S^t}}\|_2 + \|(A^T_{S^t} A_{S^t})^{-1} A^T_{S^t} e\|_2. \end{aligned}$$

Finally, using (42), then (40), (39), (41) and (14), we finish the proof

$$\begin{aligned} \|(x^t - x^*)_{|S^t}\|_2 &\leq \frac{1}{1-\delta_k} \left( \frac{1}{\eta} \|A^T_{S^t} A_{S^* \cap \overline{S^t}} u^t_{|S^* \cap \overline{S^t}}\|_2 + \|A^T_{S^t} e\|_2 \right) \\ &\leq \frac{1}{1-\delta_k} \left( \frac{\delta_{2k}}{\eta} \|u^t_{|S^* \cap \overline{S^t}}\|_2 + \sqrt{1+\delta_k} \|e\|_2 \right) \\ &= \frac{\delta_{2k}}{1-\delta_k} \frac{\|u^t\|_2}{\eta} + \frac{\sqrt{1+\delta_k}}{1-\delta_k} \|e\|_2. \end{aligned}$$

□

We have the following upper bound on $\|b^t\|_2$. This bound is given in Theorem C.7.

**Lemma C.9** (Bound of $b^t$ - RIP case)**.** *If $A$ satisfies the $(2k+1)$-RIP, for any $t \in \mathbb{N}$,*

$$\|\frac{u^t}{\eta} - A^T(Ax^t - y)\|_\infty = \frac{1}{\eta} \|b^t\|_\infty \leq \alpha^{RIP}_k \|x^*\|_2 + \gamma^{RIP}_k \|e\|_2,$$

*where $\alpha^{RIP}_k$ and $\gamma^{RIP}_k$ are defined in* (5)*.*

*Proof.* Let $t \in \mathbb{N}$ and $i \in [\![1, n]\!]$, reminding the definition of $b^t$ in (18), we have

$$\begin{aligned} |b^t_i| &= |u^t_i - \eta (A_i)^T (Ax^t - y)| \\ &= |u^t_i - \eta (A_i)^T A(x^t - x^*) + \eta (A_i)^T e| \\ &= |u^t_i - \eta (A_i)^T A_{S^* \cup S^t} (x^t - x^*)_{|S^* \cup S^t} + \eta (A_i)^T e|. \end{aligned} \tag{47}$$

We distinguish three cases: $i \in S^t$, $i \in \overline{S^*} \cap \overline{S^t}$ and $i \in S^* \cap \overline{S^t}$. We prove that in the three cases

$$|b^t_i| \leq \eta \left( \delta_{2k+1} \|x^t - x^*\|_2 + \|e\|_2 \right). \tag{48}$$

<u>$1^{st}$ *case*</u>: If $i \in S^t$, because of the definitions of $u^t$ and $x^t$, $b^t_i = 0$ and (48) holds.

$\underline{2^{nd}\ case}$: If $i \in \overline{S^*} \cap \overline{S^t}$, using the definition of $u^t$ in (14), (40), (39) and the fact that $\|A_i\|_2 = 1$ we obtain

$$\begin{aligned}
|b_i^t| &= |-\eta (A_i)^T A_{S^* \cup S^t}(x^t - x^*)_{|S^* \cup S^t} + \eta (A_i)^T e| \\
&\leq \eta \left( \|(A_i)^T A_{S^* \cup S^t}(x^t - x^*)_{|S^* \cup S^t}\|_2 + \|(A_i)^T e\|_2 \right) \\
&\leq \eta \left( \delta_{2k+1} \|(x^t - x^*)_{|S^* \cup S^t}\|_2 + \|e\|_2 \right) \\
&= \eta \left( \delta_{2k+1} \|x^t - x^*\|_2 + \|e\|_2 \right)
\end{aligned}$$

$\underline{3^{rd}\ case}$: If $i \in S^* \cap \overline{S^t}$, reminding that $\overline{\{i\}}$ is the complement of $\{i\} \subseteq [\![1, n]\!]$, (47) becomes

$$\begin{aligned}
|b_i^t| &= |-\eta x_i^* - \eta (A_i)^T A (x^t - x^*) + \eta (A_i)^T e| \\
&= \eta |-x_i^* - (A_i)^T A_{\{i\}}(x^t - x^*)_{|\{i\}} - (A_i)^T A_{\overline{\{i\}}}(x^t - x^*)_{|\overline{\{i\}}} + (A_i)^T e| \\
&= \eta |-x_i^* - (x_i^t - x_i^*) - (A_i)^T A_{\overline{\{i\}}}(x^t - x^*)_{|\overline{\{i\}}} + (A_i)^T e| \\
&= \eta |-(A_i)^T A_{\overline{\{i\}}}(x^t - x^*)_{|\overline{\{i\}}} + (A_i)^T e| \\
&\leq \eta \left( |(A_i)^T A_{\overline{\{i\}}}(x^t - x^*)_{|\overline{\{i\}}}| + |(A_i)^T e| \right).
\end{aligned}$$

Using (40), (39) and $\|A_i\|_2 = 1$, we obtain

$$\begin{aligned}
|b_i^t| &\leq \eta \left( \delta_{2k} \|x^t - x^*\|_2 + \|e\|_2 \right) \\
&\leq \eta \left( \delta_{2k+1} \|x^t - x^*\|_2 + \|e\|_2 \right).
\end{aligned}$$

Regrouping the three cases, we conclude that for all $i \in [\![1, n]\!]$, (48) holds. We now finish the proof.

Using (14) followed by Lemma C.8, we find

$$\begin{aligned}
|b_i^t| &\leq \eta \left( \delta_{2k+1} \left( \|(x^t - x^*)_{|S^t}\|_2 + \frac{\|u^t\|_2}{\eta} \right) + \|e\|_2 \right) \\
&\leq \eta \left( \delta_{2k+1} \left( \frac{\delta_{2k}}{1 - \delta_k} + 1 \right) \frac{\|u^t\|_2}{\eta} + \left( \delta_{2k+1} \frac{\sqrt{1 + \delta_k}}{1 - \delta_k} + 1 \right) \|e\|_2 \right) \\
&\leq \eta \left( \alpha_k^{RIP} \|x^*\|_2 + \gamma_k^{RIP} \|e\|_2 \right),
\end{aligned}$$

where the last inequality holds because $\frac{\|u^t\|_2}{\eta} \leq \|x^*\|_2$. □

### C.3.3 End of the proof of Theorem C.7

We now resume to the proof of Theorem C.7 and assume $A$ satisfies the $(2k+1)$-RIP and $x^*$ satisfies $(RC_{RIP})$. We remind the definitions of $T_{max}$ in (20) and $T_{RIP}$ in (6).

Using (19) and Lemma C.9, we have

$$\varepsilon = \sup_{t \in \mathbb{N}} \|b^t\|_\infty \leq \eta \left( \alpha_k^{RIP} \|x^*\|_2 + \gamma_k^{RIP} \|e\|_2 \right). \tag{49}$$

Combined with $(RC_{RIP})$, that is $\gamma_k^{RIP} \|e\|_2 < \frac{\min_{i \in S^*} |x_i^*|}{2k} - \alpha_k^{RIP} \|x^*\|_2$, this implies that

$$\varepsilon < \frac{\eta}{2k} \min_{i \in S^*} |x_i^*|.$$

Therefore $(RC)$ holds and Theorem C.4 implies that there exists $t_s \leq T_{max}$ such that $S^* \subseteq S^{t_s}$, with

$$T_{max} = \frac{2k \|\mathcal{X}^0\|_\infty + (k+1) \eta \min_{i \in S^*} |x_i^*|}{\eta \min_{i \in S^*} |x_i^*| - 2k\varepsilon}.$$

Using (49), we obtain

$$T_{max} \leq \frac{2k \|\mathcal{X}^0\|_\infty + (k+1) \eta \min_{i \in S^*} |x_i^*|}{\eta \min_{i \in S^*} |x_i^*| - 2k\eta \left( \alpha_k^{RIP} \|x^*\|_2 + \gamma_k^{RIP} \|e\|_2 \right)} = T_{RIP}.$$

We still need to prove that, when $\min_{i \in S^*} |x_i^*| > \frac{2}{\sqrt{1 - \delta_{2k}}} \|e\|_2$, $t_{BEST}$ satisfies $S^* \subseteq S^{t_{BEST}}$, as well as the last upper-bound of Theorem C.7 .

Assume by contradiction that

$$\min_{i\in S^*} |x_i^*| > \frac{2}{\sqrt{1-\delta_{2k}}}\|e\|_2 \tag{50}$$

holds but $S^* \not\subset S^{t_{BEST}}$. The construction of $t_{BEST}$, in line 11 of Algorithm 1, and the existence $t_s$ such that $S^* \subseteq S^{t_s}$ guarantee that

$$\|Ax^{t_{BEST}} - y\| \leq \|Ax^{t_s} - y\| \leq \|Ax^* - y\| = \|e\|_2.$$

Therefore, using the left inequality in (4), we obtain

$$\begin{aligned}\sqrt{1-\delta_{2k}}\|x^{t_{BEST}} - x^*\|_2 &\leq \|A(x^{t_{BEST}} - x^*)\|_2 \\ &\leq \|Ax^{t_{BEST}} - y\|_2 + \|Ax^* - y\|_2 \\ &\leq 2\|e\|_2.\end{aligned}$$

On the other hand, since we assumed $S^* \not\subset S^{t_{BEST}}$ we have

$$\|x^{t_{BEST}} - x^*\|_2 \geq \min_{i\in S^*} |x_i^*|.$$

We conclude that $\min_{i\in S^*} |x_i^*| \leq \frac{2}{\sqrt{1-\delta_{2k}}}\|e\|_2$ which contradicts (50).

As a conclusion, when $\min_{i\in S^*} |x_i^*| > \frac{2}{\sqrt{1-\delta_{2k}}}\|e\|_2$, we have $S^* \subseteq S^{t_{BEST}}$.

In this case, since the support of $x^{t_{BEST}} - x^*$ is of size smaller than $k$, we can redo the above calculation and obtain

$$\sqrt{1-\delta_k}\|x^{t_{BEST}} - x^*\|_2 \leq \|A(x^{t_{BEST}} - x^*)\|_2 \leq 2\|e\|_2.$$

This leads to the last inequality of Theorem C.7 and concludes the proof.

## C.4 Proof of Corollary 4.2

We assume that $x^*$ satisfies $(RC_{SRIP})$ and that $\|e\|_2 = 0$. Let us first prove that $x^*$ satisfies $(RC_{RIP})$. Using $(RC_{SRIP})$ we have

$$\begin{aligned}0 < 1 - \Lambda \leq 1 - 2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x_i^*|} \\ = \frac{2k}{\min_{i\in S^*}|x_i^*|}\left(\frac{\min_{i\in S^*}|x_i^*|}{2k} - \alpha_k^{RIP}\|x^*\|_2\right).\end{aligned}$$

As a consequence, since $2k > 0$ and $\min_{i\in S^*}|x_i^*| > 0$,

$$0 < \frac{\min_{i\in S^*}|x_i^*|}{2k} - \alpha_k^{RIP}\|x^*\|_2. \tag{51}$$

We conclude that $x^*$ satisfies the $(RC_{RIP})$ for $A$.

Applying Theorem 4.1 and since $\|e\|_2 = 0$ and $\mathcal{X}^0 = 0$, we know that there exists $t \leq T_{RIP} = \frac{(k+1)}{1-2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x_i^*|}}$ such that $S^* \subseteq S^t$. It is not difficult to check that the function $f : \mathbb{R} \longrightarrow \mathbb{R}$ defined for all $u \in \mathbb{R}$ by $f(u) = \frac{k+1}{1-u}$ is increasing on $[0, 1)$. By applying $f$ to

$$0 \leq 2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x_i^*|} \leq \Lambda < 1,$$

we obtain

$$T_{RIP} = f\left(2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x_i^*|}\right) \leq f(\Lambda) = T_{SRIP},$$

where $T_{SRIP}$ is defined in Corollary 4.2.

Therefore $t \leq T_{SRIP}$ and we conclude that there exists $t \leq T_{SRIP}$ such that $S^* \subseteq S^t$.

The last statement of Corollary 4.2 is a direct consequence of Theorem 4.1 and $x^*$ satisfies (7) with $\|e\| = 0$.

This concludes the proof of Corollary 4.2.

### C.5 Comments on $(RC_{RIP})$ and $(RC_{SRIP})$

The hypotheses of Theorem 4.1 are on the RIP of $A$ and there are two hypotheses on $x^*$: $(RC_{RIP})$ and (7). The condition $(RC_{RIP})$ is difficult to interpret. Below, we give an example to illustrate it.

When for all $i \in S^*$, $x_i^* = c$ for some constant $c \in \mathbb{R}$, condition $(RC_{RIP})$ becomes

$$\gamma_k^{RIP}\|e\|_2 < |c| \left( \frac{1 - 2k\alpha_k^{RIP}|S^*|^{\frac{1}{2}}}{2k} \right).$$

The existence of $c \in \mathbb{R}$ such that this condition holds depends on the sign of $1 - 2k\alpha_k^{RIP}|S^*|^{\frac{1}{2}}$. If $1 - 2k\alpha_k^{RIP}|S^*|^{\frac{1}{2}} \leq 0$, there does not exist any $c$ satisfying the condition; if $1 - 2k\alpha_k^{RIP}|S^*|^{\frac{1}{2}} > 0$, any $c \in \mathbb{R}$ satisfying

$$|c| \geq \left( \frac{2k\gamma_k^{RIP}}{1 - 2k\alpha_k^{RIP}|S^*|^{\frac{1}{2}}} \right) \|e\|_2 \tag{52}$$

leads to an $x^*$ that satisfies $(RC_{RIP})$. Therefore the condition $\alpha_k^{RIP} < \frac{1}{2}k^{-\frac{3}{2}}$ is sufficient to guarantee the existence of $x^*$ satisfying $(RC_{RIP})$. We remind that $\alpha_k^{RIP}$ has the order of magnitude of $\delta_{2k+1}$. This shows that the conditions of Theorem 4.1 are not vacuous and that the theorem can be applied at least in very incoherent settings.

Notice also that the condition (52) is stronger than (7) and that the latter might be useless[10].

Similarly, the condition $(RC_{SRIP})$ in Corollary 4.2 is not vacuous under a similar constraint on $\alpha_k^{RIP}$. If $\alpha_k^{RIP}$ is too large, there does not exist any $x^*$ satisfying $(RC_{SRIP})$. It is for instance the case if $\alpha_k^{RIP} \geq 0.5$. On the contrary, a sufficient condition of existence of vectors $x^*$ satisfying $(RC_{SRIP})$ is that the constant $\alpha_k^{RIP}$ satisfies $2k^{\frac{3}{2}}\alpha_k^{RIP} \leq \Lambda < 1$. In this case, when all non-zero entries of $x^*$ are equal, we have $\|x^*\|_2 = \sqrt{|S^*|}\min_{i\in S^*}|x_i^*|$ and $2k\alpha_k^{RIP}\frac{\|x^*\|_2}{\min_{i\in S^*}|x_i^*|} = 2k\alpha_k^{RIP}\sqrt{|S^*|} \leq 2k^{\frac{3}{2}}\alpha_k^{RIP} \leq \Lambda < 1$. Summarizing, when $2k^{\frac{3}{2}}\alpha_k^{RIP} < 1$, there exist vectors $x^*$ satisfying $(RC_{SRIP})$ and the condition of Corollary 4.2 is not vacuous.

As a conclusion, when $\alpha_k^{RIP} < \frac{1}{2}k^{-\frac{3}{2}}$, both Theorem 4.1 and Corollary 4.2 can be applied for a non-empty set of vectors $x^*$. Moreover, the sets of $x^*$ satisfying respectively $(RC_{RIP})$ and $(RC_{SRIP})$ are cones whose interior is not empty. The two sets grow as $\alpha_k^{RIP}$ decreases.

## D SEA recovers the correct support when the columns of $A$ are orthonormal

The following corollary particularizes Theorem C.4 to the noiseless and orthogonal case. In practice, a complicated algorithm like SEA is of course useless in such a case, and the state-of-the-art algorithms mentioned in the introduction have similar recovery properties. We give this corollary mostly to illustrate the diversity of links between the properties of the triplet $(A, x^*, e)$ and $\varepsilon$ and the behavior of SEA, where we remind the definitions of $b^t$ and $\varepsilon$ in (18) and (19).

**Corollary D.1** (Recovery - Orthogonal case)**.** *If $A$ is a tall (or square) matrix with orthonormal columns ($A^TA = I_n$) and $\|e\|_2 = 0$, then $\varepsilon = 0$. As a consequence, for all $x^*$, for initialization $\mathcal{X}^0 = 0$ and all $\eta > 0$, if SEA performs more than $k+1$ iterations, we have $S^* \subseteq S^{t_{BEST}}$ and $x^{t_{BEST}} = x^*$.*

To prove Corollary D.1, we first show in Lemma D.2 that the gradient noise $b^t$ is null for all $t \in \mathbb{N}$. Then, we apply Theorem C.4 and prove that $S^* \subseteq S^{t_{BEST}}$ and $x^{t_{BEST}} = x^*$.

**Lemma D.2.** *If $A$ is a tall (or square) matrix with orthonormal columns ($A^TA = I_n$) and $\|e\|_2 = 0$, then for any $t \in \mathbb{N}$ and any $\eta > 0$,*

$$\eta A^T\left(Ax^t - y\right) = u^t,$$

*i.e. $b^t = 0$.*

*Proof.* Let $t \in \mathbb{N}$. Notice first that since $\|e\|_2 = 0$ and $A$ is a tall (or square) matrix with orthonormal columns

$$A^T\left(Ax^t - y\right) = A^TA\left(x^t - x^*\right) = x^t - x^*. \tag{53}$$

To prove the Lemma, we distinguish three cases: $i \in S^t$, $i \in \overline{S^*} \cap \overline{S^t}$ and $i \in S^* \cap \overline{S^t}$.

[10] We have not investigated this question.

$\underline{1^{st}\ case}$: If $i \in S^t$, $\eta \left(A^T \left(Ax^t - y\right)\right)_i = 0 = u_i^t$. The first equality is a consequence of the definition of $x^t$ in Algorithm 1, line 7. The second is due to the definition of $u^t$, in (14).

$\underline{2^{nd}\ case}$: If $i \in \overline{S^*} \cap \overline{S^t}$, taking the $i$th entry of (53) and using the support constraints of $x^t$ and $x^*$, we find

$$\eta \left(A^T \left(Ax^t - y\right)\right)_i = 0 = u_i^t,$$

where the second equality is due to the definition of $u^t$, in (14).

$\underline{3^{rd}\ case}$: If $i \in S^* \cap \overline{S^t}$, the $i$th entry of (53) becomes

$$\eta \left(A^T \left(Ax^t - y\right)\right)_i = -\eta x_i^* = u_i^t,$$

where again the second equality is due to the definition of $u^t$, in (14).

□

*Proof.* We now resume the proof of Corollary D.1 and assume that $A$ is a tall (or square) matrix with orthonormal columns ($A^T A = I_n$), $\|e\|_2 = 0$ and $\mathcal{X}^0 = 0$. We remind the definition of $T_{max}$ in (20).

Using Lemma D.2, (19) and (18), we find that $\varepsilon = 0$. Therefore ($RC$) holds for all $x^*$ and Theorem C.4 implies that there exists $t_s \leq T_{max}$ such that $S^* \subseteq S^{t_s}$. Since $\mathcal{X}^0 = 0$ and $\varepsilon = 0$, we find $T_{max} = k + 1$.

Since $\|e\|_2 = 0$, we know from Theorem C.4 and the definitions of $t_{BEST}$ and $x^t$ in Algorithm 1 that

$$\|Ax^{t_{BEST}} - y\|_2 \leq \|Ax^{t_s} - y\|_2 \leq \|Ax^* - y\|_2 = 0.$$

Using that $A$ is a tall (or square) matrix with orthonormal columns ($A^T A = I_n$), and $\|e\|_2 = 0$, this leads to

$$\begin{aligned} 0 =& Ax^{t_{BEST}} - y \\ =& A^T A(x^{t_{BEST}} - x^*) \\ =& x^{t_{BEST}} - x^*. \end{aligned}$$

Therefore, $S^* = \text{supp}(x^*) = \text{supp}(x^{t_{BEST}}) \subseteq S^{t_{BEST}}$.

This concludes the proof of Corollary D.1. □

# E Additional results for phase transition diagram experiment

We consider the same experiment as in Section 5.1 but in the noiseless setting. The analog of the curves of Figure 2 are in Figure 6. Again, an artifact stemming from the discrete values of $(m, n, k)$ is responsible for the smooth and decreasing part observed on the left side of the phase transition curves, in a region where $k = 1$. Without noise, all algorithms exhibit a similar phase transition curve and maintain the same ranking as in the noisy setting. The conclusions that are drawn in the noiseless setting from Figure 6 are analog to those in Section 5.1 in the noisy setting.

Figure 6: Phase transition diagram (noiseless setting). Problems below each curve are solved by the related algorithm with a success rate larger than 95%. $\zeta = m/n$ denotes the ratio between the number of rows and the number of columns in $A$ while $\rho = k/m$ denotes the ratio between the sparsity and the number of rows in $A$. Matrix $A$ have i.i.d. standard Gaussian entries and non-zero entries in $x^*$ are drawn uniformly in $[-2, -1] \cup [1, 2]$. $n = 500$ is fixed and results are obtained from 1000 runs.

# F Additional results in deconvolution

To supplement Section 5.2, we present additional results for the initial experimental setup. In Appendix F.1, we provide the results for the full signal of Figure 3. In Appendix F.2, we display the loss along the iterative process for the experiment shown in Figure 3. In Appendix F.3 (resp. Appendix F.4), we show the average number of explored supports (resp. the average loss and Wasserstein distance) over the $r = 200$ problems solved to construct Figure 4 as $k$ varies. Finally, we present results for the noiseless case when $e = 0$ in Appendix F.5.

## F.1 Deconvolution: Examining the specific instance from Figure 3

In Figure 7, we present the full signal corresponding to the cropped instance in Figure 3. For clarity, Figure 8 displays the same crop as Figure 3, presenting the results obtained with all studied algorithms.

In this representation, nearly all algorithms can identify isolated spikes. However, challenges arise when spikes are close, leading algorithms to struggle with precise localization. Notably, IHT and HTP exhibit false detections in the most energetic part of the signal (around positions 140, 180, 260, and 400), getting trapped in local minima. On this experiment (this is not the case in general), initializing SEA with ELS or OMP allows $\text{SEA}_{\text{ELS}}$ and $\text{SEA}_{\text{OMP}}$ to find a better approximation of $S^*$ than $\text{SEA}_0$. These two versions of SEA successfully recover the original signal, except for two spikes between positions 410 and 425. In contrast, other algorithms fail due to the coherence of $A$ and the presence of additive noise.

Figure 7: Full version of Figure 3. Representation of an instance of $x^*$, $y$ and the solutions provided by the algorithms ($k = 20$, $n = 500$).

Figure 8: Crop from the dashed area in Figure 7, matching the location of Figure 3 with results from all analyzed algorithms. This region corresponds to the most densely populated area within the signal.

## F.2 Deconvolution: The loss along the iterative process

Figures 9 and 10 illustrate the behavior of HTP, IHT, ELS, $\text{SEA}_0$, $\text{SEA}_{\text{OMP}}$ and $\text{SEA}_{\text{ELS}}$, for the same 20-sparse vector $x^*$ used in Figure 3 (Section 5.2) and Figures 7 and 8 (Appendix F.1), throughout the iterative process.

Figure 9: Representation of $\ell_{2,\text{rel\_loss}}(x^t)$ (solid lines) and $\ell_{2,\text{rel\_loss}}(x^{t_{BEST}(t)})$ (dashed lines) for each iteration of several algorithms, for the experiment of Figure 3.

Figure 10: Representation of $\ell_{2,\text{rel\_loss}}(x^{t(s)})$ (solid lines) and $\ell_{2,\text{rel\_loss}}(x^{t_{BEST}(t(s))})$ (dashed lines) for each new explored support of several algorithms, for the experiment of Figure 3.

More precisely, in Figure 9 the solid curves represent $\ell_{2,\text{rel\_loss}}(x^t)$ when $t$ varies in $[\![0, 1000]\!]$, where $\ell_{2,\text{rel\_loss}}$ is defined by

$$\ell_{2,\text{rel\_loss}}(x) = \frac{\|Ax - y\|_2}{\|y\|_2}. \tag{54}$$

The dashed lines represent $\ell_{2,\text{rel\_loss}}(x^{t_{BEST}(t)})$ where $t_{BEST}(t) = \underset{t' \in [\![0,t]\!]}{\text{argmin}} \|Ax^{t'} - y\|_2^2$ and $t$ varies in $[\![0, 1000]\!]$.

Overall, no algorithm succeeds in reaching zero error. ELS performs only one iteration before stopping in a local minimum ($\ell_{2,\text{rel\_loss}}(x^1) \approx 0.2$). HTP completes a few iterations before stopping. IHT outperforms HTP by exploring a bit more. One can observe that, due to the exploratory nature of SEA, $\ell_{2,\text{rel\_loss}}(x^t)$ oscillates for both versions of SEA. This exploration enables $\text{SEA}_{\text{ELS}}$ to refine the ELS estimate within 300 iterations. Despite faster decay around the $100^{th}$ iteration, $\text{SEA}_{\text{OMP}}$ finally reaches $\text{SEA}_{\text{ELS}}$ after 620 iterations.

We observe that HTP and IHT exhibit poor performance due to the high coherence of $A$. As demonstrated in Appendix F.1, these algorithms initially make the mistake of erroneously assigning several neighboring atoms to represent the same large bump and fail to correct this error during the iterative process.

Figure 10 illustrates the same iterative process as Figure 9, focusing on support exploration rather than the iteration count for each algorithm. Here, the solid curves represent $\ell_{2,\text{rel\_loss}}(x^{t(s)})$ when $s$ varies from 0 to the number of explored supports, where $t(s)$ is the iteration associated to the $s^{th}$ explored support (without redundancy). As in the previous figure, the dashed lines represent $\ell_{2,\text{rel\_loss}}(x^{t_{BEST}(t(s))})$ where $t_{BEST}(t(s)) = \underset{t' \in [\![0,t(s)]\!]}{\text{argmin}} \|Ax^{t'} - y\|_2^2$. This is the loss associated to the best estimate found while exploring the $s^{th}$ supports.

We observe that HTP explores very little before stopping in a local minimum. Despite performing only one iteration, ELS explores 500 supports within the neighborhood of its OMP initialization for a slight improvement. Here, $\text{SEA}_0$ explores one new support at each iteration, while $\text{SEA}_{\text{ELS}}$ explores fewer, improving upon ELS by exploring less than 250 supports. Again, despite faster decay at the beginning, $\text{SEA}_{\text{OMP}}$ finally reaches $\text{SEA}_{\text{ELS}}$ after exploring around 520 supports. This reveals how efficient each algorithm is at finding relevant supports.

## F.3 Deconvolution: number of explored supports

As discussed in Section 3.3, the overall cost of the algorithms depends on the number of explored supports. In Figure 11, we illustrate the number of explored supports in two different ways. First, in Figure 11 (left), we present the average number of explored supports for the entire problem resolution — representing the overall cost. This includes supports explored before initialization. For instance, $\text{SEA}_{\text{OMP}}$ includes both the supports explored by OMP for its initialization and those explored subsequently in the SEA procedure. Then, in Figure 11 (right), we present the average number of explored supports that actually required computation after initialization. These curves reveal the cost of the algorithms after initialization, where supports seen before the initialization (e.g., those of OMP for $\text{SEA}_{\text{OMP}}$) are not included as they do not incur additional computing time.

Figure 11: Left: Average number of explored supports by algorithms solving the 200 problems in Section 5.2, across sparsity levels $k \in [\![1, 50]\!]$. Right: Average number of explored supports from algorithms initialization in the same setup.

Examining the overall cost of the algorithms on the left, we observe three types of exploration profiles. Some algorithms, such as OMP, OMPR, and HTP, exhibit minimal exploration. Notably, as $k$ increases, HTP explores fewer supports. On the other hand, algorithms like ELS explore extensively. SEA falls in between, exploring a few supports for small $k$ and more as $k$ increases until reaching a threshold. Despite exploring at least two times fewer supports than ELS, SEA's more efficient exploration allows it to achieve better results, as demonstrated in Figure 4.

From this figure, we observe that adding SEA to ELS ($\text{SEA}_{\text{ELS}}$) does not significantly alter the order of magnitude of the cost. Turning our attention to the cost after initialization on the right, we do not observe $\text{HTP}_{\text{ELS}}$ and $\text{HTP}_{\text{OMP}}$ because they do not explore after their initialization, as shown in Figure 10 for $\text{HTP}_{\text{ELS}}$. OMP and $\text{SEA}_0$ curves remain unchanged because they do not have any initializing algorithms.

All SEA variants exhibit a similar order of magnitude of explored supports. However, we conclude that the stronger the initialization (with $0 <$ OMP $<$ ELS), the more challenging the exploration becomes due to the high coherence of $A$ and the local minima in which OMP and ELS end up.

## F.4 Deconvolution: The average loss and Wasserstein distance when $k$ varies

In this section, we complement the analysis of the experiment described in Section 5.2, the results of which are already depicted in Figure 4.

In Figure 12, we present the average – over the $r = 200$ problems – of the relative $\ell_2$ loss ($\ell_{2,\text{rel\_loss}}$), defined in (54), for the outputs of all algorithms and for $k \in [\![1, 50]\!]$. We observe that all versions of SEA achieve the lowest errors for $k < 20$. The largest gap between SEA and its competitors is observed for $k$ between 9 and 13. For clarity, the curves for HTP and IHT are visible for small $k$ only. Due to the high coherence of $A$ and their method of selecting multiple elements of the support estimate at once, both IHT and HTP attempt to reconstruct single peaks with multiple atoms, leading to much larger errors than those of the competitors.

Figure 12: Mean of $\ell_{2,\text{rel\_loss}}(x)$ – defined in (54) – for the outputs of the algorithms on the 200 problems of Section 5.2, for each sparsity level $k \in [\![1, 50]\!]$.

Figure 13: Mean of the Wassertstein distance between the outputs of the algorithms and the solutions $x^*$ of the 200 problems of Section 5.2, for each sparsity level $k \in [\![1, 50]\!]$.

In Figure 13, we show the mean of the Wasserstein-1-distance (also called Earth mover's distance) over the same problems. It illustrates how 'far' the chosen spikes are from the true ones. Again, all the versions of SEA achieve the smallest distances for $k < 18$. As $k$ increases, despite being the best at

finding the exact position of the spikes (see Figure 4), $\text{SEA}_0$ and, to a lesser extent, $\text{SEA}_{\text{OMP}}$ and $\text{SEA}_{\text{ELS}}$, choose spikes 'far' from the true ones when they are mistaken, while IHT improves for the highest $k$.

## F.5 Deconvolution: Results in the noiseless setup

We consider the same experiment as in Section 5.2 but in a noiseless setting ($e = 0$). Thus, we set again n = 500, a convolution matrix $A$ corresponding to a Gaussian filter with a standard deviation equal to 3. We tested every algorithm on $r = 200$ noiseless problems, for different $k$-sparse $x^*$, with $k \in [\![1, 50]\!]$.

### F.5.1 Visualization of a specific instance

The counterparts of the curves in Figures 7 and 8 from Appendix F.2 in the noiseless case are shown in Figures 14 and 15. The algorithms behave in a similar way to the noisy case. However, with no perturbation in the signal, all versions of SEA successfully recover the exact positions of the spikes, whereas no other algorithm achieves such a performance.

Figure 14: Representation of an instance of $x^*$ and $y$ with the solutions provided by the algorithms when $k = 20$ and $n = 500$ in the noiseless case: Full signal.

Figure 15: Crop from the dashed area in Figure 14. This region corresponds to the most densely populated area within the signal.

### F.5.2 The loss along the iterative process

The analogs of Figures 9 and 10 from Appendix F.2 are respectively shown in Figures 16 and 17. The conclusions drawn here in the noiseless setting are similar to those in Appendix F.2.

Similarly to the noisy case, it can be observed in Figure 16 that due to the exploratory nature of SEA, $\ell_{2,\text{rel\_loss}}(x^t)$ oscillates for all versions of SEA. However, this does not prevent $\text{SEA}_0$ from finding a better approximation of $S^*$ than ELS in the first 80 iterations and eventually recovering $S^*$ despite the high coherence of $A$. Indeed, for all SEA versions, once $S^*$ is recovered in the noiseless setting, for $t$ sufficiently large, $x^t = x^*$, and therefore $Ax^t - y = 0$. Using the update rule of $\mathcal{X}^t$ in line 8 of Algorithm 1, we observe that $\mathcal{X}^t$ should no longer evolve, and no new support is explored. This behavior is evident not only from Figure 16 but also from Figure 17. Furthermore, as can be seen in Figure 14, ELS does not improve OMP thus leading to an identical initialization for $\text{SEA}_{\text{OMP}}$ and $\text{SEA}_{\text{ELS}}$. Consequently, these last two algorithms thus follow the same trajectory. From Figure 17, we however observe that$\text{SEA}_{\text{ELS}}$ and $\text{SEA}_{\text{OMP}}$ must explore twice as many supports as $\text{SEA}_0$.

Figure 16: Representation of $\ell_{2,\text{rel\_loss}}(x^t)$ (solid lines) and $\ell_{2,\text{rel\_loss}}(x^{t_{BEST}(t)})$ (dashed lines) for each iteration of several algorithms, for the noiseless experiment.

### F.5.3 Number of explored supports

The analog of Figure 11 from Appendix F.3 is shown in Figure 18. The conclusions drawn here in the noiseless case are similar to those in Appendix F.3.

All the algorithms behave in the same way as in the noisy experiment.

### F.5.4 The average $\text{dist}_{supp}$, loss, and Wasserstein distance when $k$ varies

The analogs of Figures 4, 12 and 13 are respectively displayed in Figures 19 to 21. The results in the noiseless setting closely mirror those in Section 5.2 and Appendix F.4 in the noisy setting.

In Figure 19, for sparsity levels $k < 30$, $\text{SEA}_0$, $\text{SEA}_{\text{OMP}}$, and $\text{SEA}_{\text{ELS}}$ outperform the other algorithms. Across all studied sparsity levels, $\text{SEA}_0$ is reaching the best performances.

Moving to Figure 20, the absence of noise makes the problems easier to solve. Despite overall improvement, $\text{SEA}_0$ still attains the lowest error for the smallest $k$, followed by $\text{SEA}_{\text{ELS}}$. Once again, HTP and IHT exhibit much larger errors than their competitors.

Figure 17: Representation of $\ell_{2,\text{rel\_loss}}(x^{t(s)})$ (solid lines) and $\ell_{2,\text{rel\_loss}}(x^{t_{BEST}(t(s))})$ (dashed lines) for each new explored support of several algorithms, for the noiseless experiment.

Figure 18: Left: Mean of the number of the explored supports by algorithms solving the 200 problems in the noiseless case, across sparsity levels $k \in [\![1, 50]\!]$. Right: Mean of the number of the explored supports from algorithms initialization in the same setup.

In Figure 21, as $k$ increases, $\text{SEA}_0$, followed by $\text{SEA}_{\text{ELS}}$ and eventually IHT, exhibits the lowest Wasserstein distance.

Figure 19: Mean of support distance $\text{dist}_{\text{supp}}$ between $S^*$ and the support of the solutions provided by several algorithms as a function of the sparsity level $k$ in the noiseless setup.

Figure 20: Mean of $\ell_{2,\text{rel\_loss}}(x)$ – defined in (54) – for the outputs of the algorithms on the 200 problems of the noiseless setup.

Figure 21: Mean of the Wassertstein distance between the outputs of the algorithms and the solutions $x^*$ of the 200 problems of the noiseless setup.

# G Supervised Machine Learning experiments

We describe here supervised learning experiments: they confirm that $\text{SEA}_0$ performs well in small dimensions, and performs better in high dimensions when combined with OMP or ELS. They also give evidence that SEA can perform very well in the presence of error/noise and when no perfect sparse vector fits the data.

## G.1 Context

In a supervised learning setting, the rows of matrix $A \in \mathbb{R}^{m \times n}$ (often denoted by $X$) are the $n$-dimensional feature vectors associated with the $m$ training examples, while the related labels are in vector $y \in \mathbb{R}^m$. In the training phase, a sparse vector $x$ (often denoted $\beta$ or $w$) is optimized to fit $y \approx Ax$ using an appropriate loss function. In this context, support recovery is called model selection.

Based on the experimental setup of [2], we compare the training loss for different levels of sparsity, for all the algorithms, on linear regression and logistic regression tasks. We use the preprocessed public datasets[11] provided by [2], following the same preprocessing pipeline: we augment $A$ with an extra column equal to 1 to allow a bias and normalize the columns of $A$.

We present results for regression problems in Appendix G.2 and for classification problems in Appendix G.3.

## G.2 Regression datasets

As we are working with real datasets without ground truth, we use the $\ell_2$ regression loss $\ell_2\_\text{loss}(x) = \frac{1}{2}\|Ax - y\|_2^2$ for $x \in \mathbb{R}^n$ for regression problems.

As shown in Figure 22, $\text{SEA}_0$, $\text{SEA}_{\text{OMP}}$ and $\text{SEA}_{\text{ELS}}$ are at the same level as ELS on a regression dataset with $n$ small as in `comp-activ-harder`. For the higher dimensional regression dataset as `year` (see Figure 23), $\text{SEA}_0$ performs poorly as $k$ increases, but $\text{SEA}_{\text{ELS}}$ can improve ELS performances and outperforms the other algorithms.

Figure 22: Performance on regression dataset `comp-activ-harder` ($m = 8191$ examples, $n = 12$ features).

Figure 23: Performance on regression dataset `year` ($m = 463715$ examples, $n = 90$ features).

In a low-dimensional problem ($n$ is small) as `cal_housing` dataset ($m = 20639$ examples, $n = 8$ features) in Figure 24, we see that $\text{SEA}_{\text{OMP}}$ and $\text{SEA}_{\text{ELS}}$ perform better than ELS, and $\text{SEA}_0$ outperform them for a sparsity $k \in [\![5, 8]\!]$. It is worth mentioning that HTP obtains good performances for this dataset.

The same experiment is reported on Figure 25, but for the dataset `slice` ($m = 53500$ examples, $n = 384$ features). This is an intermediate-dimensional problem. Figure 25 shows that $\text{SEA}_0$ obtains slightly worse results than $\text{SEA}_{\text{ELS}}$, $\text{SEA}_{\text{OMP}}$ and ELS. The non-decreasing curve of HTP comes from its support estimation technique. Since the coherence of the `slice` dataset is 1, HTP selects highly correlated features and fails to correct this mistake.

Figure 24: Performance on the regression dataset `cal_housing` ($m = 20639$ examples, $n = 8$ features).

Figure 25: Performance on the regression dataset `slice` ($m = 53500$ examples, $n = 384$ features).

## G.3 Classification datasets

In these experiments, we consider the logistic regression loss defined by

$$\text{log\_loss}(x) = \sum_{i=1}^{m} \left(-y_i \log(\sigma((Ax)_i)) - (1 - y_i) \log(1 - \sigma((Ax)_i))\right),$$

[11] https://drive.google.com/file/d/1RDu2d46qGLI77AzliBQleSsB5WwF83TF/view

where $\sigma(t) = \frac{1}{1+e^{-t}}$ is the sigmoid function.

We need to adapt SEA to this new loss. In Algorithm 1, line 7 is replaced by $x^t = \underset{\substack{x \in \mathbb{R}^n \\ \text{supp}(x) \subseteq S^t}}{\text{argmin}} \text{log\_loss}(x)$ and line 8 is replaced by $\mathcal{X}^{t+1} = \mathcal{X}^t - \eta \nabla \text{log\_loss}(x^t)$. Similar adaptations are performed on the other algorithms.

The loss log_loss$(x)$, for the `letter` dataset ($m = 20000$ examples, $n = 16$ features), for all $k \in [\![1, 12]\!]$ and for all algorithms is depicted in Figure 26. We depict the same curves obtained for the `ijcnn1` dataset ($m = 24995$ examples, $n = 22$ features) in Figure 27. These two last figures show that $\text{SEA}_0$, $\text{SEA}_{\text{OMP}}$ and $\text{SEA}_{\text{ELS}}$ achieve similar performances to ELS.

Figure 26: Performance on the classification dataset `letter` ($m = 20000$ examples, $n = 16$ features).

Figure 27: Performance on the classification dataset `ijcnn1` ($m = 24995$ examples, $n = 22$ features).