Iterative Sparsification-Projection (ISP): Fast and Robust Sparse Signal Approximation

Mostafa Sadeghi, *Student Member, IEEE*, Massoud Babaie-Zadeh, *Senior Member, IEEE*

***Abstract*—In this paper, we address recovery of sparse signals from compressed measurements, and sparse signal approximation, which have received considerable attention over the last decade. First, we revisit smoothed L0 (SL0), a well-known sparse recovery algorithm, and give some insights into it that have not been noticed previously. Specifically, we re-derive the SL0 algorithm based on proximal methods, and using recent results in solving non-convex problems by proximal algorithms, we provide a convergence guarantee for it. In addition, inspired by this derivation, we propose a general family of algorithms, which we call iterative sparsification-projection (ISP), having SL0 as a special member. Our algorithmic framework starts with an initial guess for the unknown sparse vector, and then iteratively sparsifies it (using a fixed threshold) followed by projecting the result onto the admissible solution set. The threshold is then decreased and the same process is repeated. The algorithm terminates when the threshold becomes sufficiently small, or another stopping criterion is satisfied. We also propose a robust projection to handle the situations with observation noise or model uncertainties. Our extensive simulations confirm the promising performance of the ISP algorithms compared with some well-known algorithms.**

***Index Terms*—Sparse signal approximation, compressed sensing, SL0, proximal algorithms, gradient projection, ADMM**

I. Introduction

A. Sparse signal recovery

Solving least squares problems under some sparsity constraints has been extensively studied over the last decade. This problem arises in, for example, *sparse signal approximation* [1] and *regression and variable selection* [2], with many applications, including compressed sensing (CS) [3], image enhancement and compression [1], signal separation [4], medical image reconstruction [5], and pattern recognition [6].

In sparse signal approximation or representation, given a signal $\mathbf{y} \in \mathbb{R}^m$ and a dictionary $\mathbf{D} \in \mathbb{R}^{m \times n}$, which is a collection of n atoms with $n > m$, the goal is to represent $\mathbf{y}$ as a linear combination of the atoms of $\mathbf{D}$ in a parsimonious way; *i.e.*, by using as few atoms as possible. On the other hand, in CS, it is intended to recover a sparse signal from a number of linear measurements, far fewer than the signal dimension. In both cases, the main task is to find the sparsest solution of the underdetermined system of linear equations $\mathbf{y} = \mathbf{D}\mathbf{x}$. To this end, the following problem has to be solved

$$(P_0): \quad \min_{\mathbf{x}} \ \|\mathbf{x}\|_0 \quad \text{s.t.} \quad \mathbf{y} = \mathbf{D}\mathbf{x},$$

where $\|.\|_0$ is the so-called ℓ_0 (pseudo) norm, defined as the number of non-zero entries. The ℓ_0 norm function is highly discontinuous and non-differentiable, making the above problem hard to solve[1]. To remedy this issue, alternative sparsity promoting functions have been introduced. The most well-known approximating function is the ℓ_1 norm, which is the closest convex norm to the ℓ_0 function [1]. Using the ℓ_1 norm, the sparse signal recovery problem becomes

$$(P_1): \quad \min_{\mathbf{x}} \ \|\mathbf{x}\|_1 \quad \text{s.t.} \quad \mathbf{y} = \mathbf{D}\mathbf{x}\cdot$$

In practical applications, where there is usually observation noise or model mismatch, the noise-aware variants of the above problems are used [8], [9], in which, the equality constraint is replaced with $\|\mathbf{y}-\mathbf{D}\mathbf{x}\|_2 \leq \epsilon$, where ϵ is an upper-bound on the noise power. We denote the robust versions of P_0 and P_1 by P_0^ϵ and P_1^ϵ, respectively.

Numerous algorithms have been proposed for solving the above-mentioned sparse recovery problems; see *e.g.* [10]. One broad family of algorithms targets the P_1 and P_1^ϵ problems, due to their convexity. Well-known examples of this group include iterative shrinkage-thresholding (IST) [11], [12] and its variants [13], [14], message passing algorithms [15], [16], [17], gradient projection for sparse reconstruction (GPSR) [18], spectral projection gradient method (SPGL1) [19], and NESTA [20], to name only a few. Greedy algorithms, such as orthogonal matching pursuit (OMP) [21], generalized OMP (GOMP) [22], and compressive sampling matching pursuit (CoSAMP) [23] are another family of algorithms. These algorithms do not directly solve any optimization problem, such as P_0, but they sequentially select appropriate atoms out of the dictionary that result in the lowest residual in sparsely representing the target signal.

Although using the ℓ_1 norm leads to a convex problem, which is favorable in optimization, it has been shown that better results are achieved by using non-convex functions, such as ℓ_p (pseudo) norms for $0 \leq p < 1$. For instance, in a recent work, Zheng *et al.* [24] considered the following ℓ_p-penalized least squares problem:

$$\min_{\mathbf{x}} \ \left\{\frac{1}{2}\|\mathbf{y}-\mathbf{D}\mathbf{x}\|_2^2+\lambda\|\mathbf{x}\|_p^p\right\} \tag{1}$$

and showed that in the noiseless setting and with an optimal λ, all values of $p \in [0, 1)$ have the same performance, which

This work has been supported by the Center for International Scientific Studies and Collaboration (CISSC).

The authors are with the Electrical Engineering Department, Sharif University of Technology, Tehran, Iran (e-mail: m.saadeghii@gmail.com; mbzadeh@yahoo.com).

[1] In fact, it has been shown that P_0 is NP-hard [7], [1].

is better than the choice $p = 1$. Moreover, it has been shown that $p = 0$ and $p = 1$ outperform the other values of p for very small and very large amounts of noise, respectively [24].

Examples of the algorithms that use other sparsity promoting functions than the ℓ_1 norm include iterative re-weighted least squares (IRLS) [25], iterative hard thresholding (IHT) [26], and smoothed ℓ_0 (SL0) [27]. Specifically, SL0 approximates the non-smooth ℓ_0 norm by a differentiable function equipped with a smoothing parameter (denoted by σ in [27]). A smaller smoothing parameter results in a better approximation of ℓ_0 norm. The general idea of SL0 is then to use gradient-projection. That is, starting with an initial guess, the solution is iteratively updated by one-step gradient descent of the approximating function (gradient step) followed by projecting the result onto the admissible solution set (projection step). Moreover, noting that the non-convexity of the smooth function increases by decreasing the smoothing parameter, a continuation trick is used, in which, the smoothing parameter is gradually decreased to avoid undesired local minima while getting close to the ℓ_0 norm.

A noticeable remark regarding the sparse recovery algorithms is that, most of them, including IST, IHT, and IRL1 solve the regularized (unconstrained) versions of P_0^ϵ or P_1^ϵ given in (1). It can be shown that under appropriate selection of λ, the unconstrained problems become equivalent to their constrained counterparts. Although solving the unconstrained problems are easier, there is no exact recipe for choosing the regularization parameter λ. Moreover, in most practical applications, including signal denoising and compression, there exists a good estimate of ϵ. These findings have motivated some researchers to propose algorithms for directly solving P_0^ϵ or P_1^ϵ. SPGL1, NESTA, and SL0 are well-known examples of these attempts.

B. Contributions

In this paper, we consider the constrained sparse recovery problem, and by focusing on the SL0 algorithm, we make the following contributions:

1) We provide new insights into SL0 by developing it using the idea of *proximal algorithms* [28]. This leads to a better understanding of the algorithm. Alongside, we establish a convergence guarantee for the sequence generated by SL0, corresponding to any fixed σ. As a by-product, the convergence bound for the step-size parameter in the gradient step of SL0 is derived. We also give some other insights into the final value and the decreasing sequence of the smoothing parameter σ.
2) It is shown that the gradient-projection approach used in SL0 is in fact equivalent to sparsification (thresholding)-projection, based on which, we derive a family of iterative sparsification-projection (ISP) algorithms that work as follows. Starting with an initial guess, the two steps of sparsification and projection are repeatedly performed. The sparsification step can be realized using the well-known *hard/soft thresholding* functions, or by one-step gradient descent of a smooth sparsity promoting function. Moreover, the threshold is gradually decreased along iterations, similar to the σ parameter in SL0. This is actually the idea used in *deterministic annealing* (DA) methods in optimization, where for solving a non-convex problem, a sequence of sub-problems are solved; each in a lower temperature than its previous one. Here, the threshold plays the role of temperature in DA methods[2].
3) Although a robust version of SL0 has already been proposed in [30], we experimentally show that it is sub-optimal and its performance deteriorates dramatically in some situations. To address this issue, we propose a new algorithm to implement the projection step of the ISP algorithms.

C. Structure of the Paper

The rest of the paper is organized as follows. In Section II, the notations used throughout the paper are introduced. This section is then followed by a review on the basics of proximal algorithms. In Section III, we review the main steps of the SL0 algorithm. Section IV presents our own works, including the new derivation of SL0, and the ISP algorithms. Section V is devoted to simulation results, in which, the ISP algorithms are compared with some well-known algorithms, including GOMP [22], expectation-maximization Gaussian-mixture approximate message passing (EM-GM-AMP) [17], and NESTA [20].

II. Preliminaries and notations

Throughout the paper, small and capital bold face characters are used for vector- and matrix-valued quantities, respectively. The notation dom$f$ denotes the domain of the function f. The superscript T denotes matrix or vector transposition. The identity matrix is denoted by $\mathbf{I}$.

In what follows, the basics of the proximal algorithms are reviewed from [28]. First note the following definitions:

Definition 1. *A function f : $domf \longrightarrow \mathbb{R}$ is called Lipschitz continuous if there exists a constant $L > 0$ such that*

$$\forall x, y \in dom f : \quad |f(x) - f(y)| \leq L|x - y| \cdot$$

L is referred to as a Lipschitz constant of f. The smallest constant is sometimes called the (best) Lipschitz constant. A Lipschitz continuous function is almost everywhere differentiable and has a bounded derivative [31].

Definition 2 ([28])**.** *The proximal mapping (prox-operator) of a convex function g : $domg \longrightarrow \mathbb{R}$ is defined as*

$$prox_g(\mathbf{x}) \triangleq \underset{\mathbf{u} \in dom g}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{x}\|_2^2 + g(\mathbf{u}) \right\} \cdot$$

As an example, which will be used in the remaining of the paper, consider $g(\mathbf{x}) = \lambda \|\mathbf{x}\|_1$. Its prox-operator is the so-called *soft-thresholding* function [1]:

$$\operatorname{prox}_g(\mathbf{x}) = \underset{\mathbf{u}}{\operatorname{argmin}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{x}\|_2^2 + \lambda \|\mathbf{u}\|_1 \right\} = \mathcal{T}_\lambda^s(\mathbf{x}),$$

[2] It is worth mentioning that this idea is called graduated non-convexity (GNC) in the vision literature [29].

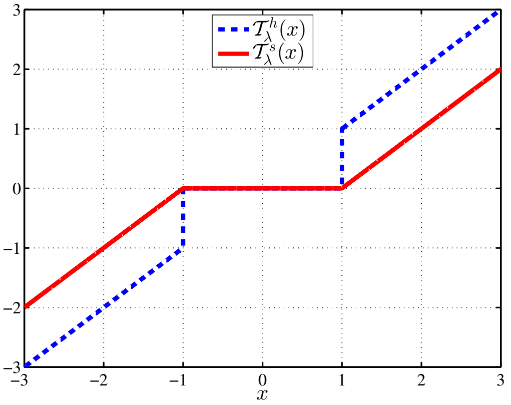


Fig. 1: Soft *vs.* hard thresholdings ($\lambda = 1$).

where the component-wise soft-thresholding function $\mathcal{T}^s_\lambda$ is defined as

$$\mathcal{T}^s_\lambda(x) \triangleq \begin{cases} x-\lambda & x>\lambda \\ 0 & |x|\leq\lambda \\ x+\lambda & x<-\lambda \end{cases}.$$

Another important example is the ℓ_0 norm: $g(\mathbf{x}) = \lambda\|\mathbf{x}\|_0$, whose prox-operator leads to *hard-thresholding* [1]:

$$\operatorname{prox}_g(\mathbf{x}) = \underset{\mathbf{u}}{\operatorname{argmin}} \left\{\frac{1}{2}\|\mathbf{u}-\mathbf{x}\|_2^2+\lambda\|\mathbf{u}\|_0\right\} = \mathcal{T}^h_{2\lambda}(\mathbf{x}),$$

where the component-wise hard-thresholding function $\mathcal{T}^h_\lambda$ is defined as

$$\mathcal{T}^h_\lambda(x) \triangleq \begin{cases} x & |x|>\sqrt{\lambda} \\ 0 & |x|\leq\sqrt{\lambda} \end{cases}.$$

These two thresholding functions are depicted in Fig. 1.

Now, we are ready to review the proximal algorithms. These methods target the following minimization problem:

$$\min_{\mathbf{x}} \left\{h(\mathbf{x}) \triangleq f(\mathbf{x}) + g(\mathbf{x})\right\}, \tag{2}$$

where, $f:\ \operatorname{dom} f \longrightarrow \mathbb{R}$ is a convex and continuously differentiable function (hence its gradient is Lipschitz continuous), and $g:\ \mathbb{R}^n \longrightarrow \mathbb{R}$ is a non-smooth convex function. Let L be the Lipschitz constant of ∇f. It can be shown that [31]

$$\forall x,y \in \operatorname{dom} f: \quad f(\mathbf{x}) \leq \tilde{f}(\mathbf{x},\mathbf{y}),$$

where,

$$\tilde{f}(\mathbf{x},\mathbf{y}) \triangleq f(\mathbf{y}) + \nabla f(\mathbf{y})^T(\mathbf{x}-\mathbf{y}) + \frac{1}{2\mu_l}\|\mathbf{x}-\mathbf{y}\|_2^2 \tag{3}$$

is a quadratic upper-bound of f at $\mathbf{y}$, and $\mu_l \in (0, 1/L]$. The idea of the proximal methods is then to iteratively minimize h, with f being replaced with its upper-bound at the current estimate. That is,

$$\mathbf{x}_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{\tilde{h}(\mathbf{x}) \triangleq \tilde{f}(\mathbf{x},\mathbf{x}_k) + g(\mathbf{x})\right\}\cdot \tag{4}$$

This is very similar to *majorization-minimization algorithms* [28], in which, for minimizing a gradient Lipschitz function f, the following iterations are performed

$$\mathbf{x}_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}}\ \bar{f}(\mathbf{x},\mathbf{x}_k),$$

where $\bar{f}(\mathbf{x},\mathbf{x}_k)$ is a so-called *majorizing* function of f; a convex upper-bound to f that is tight at $\mathbf{x}_k$, *i.e.*, for all $\mathbf{x}$, $\bar{f}(\mathbf{x},\mathbf{x}_k) \geq f(\mathbf{x})$ and $\bar{f}(\mathbf{x}_k,\mathbf{x}_k) = f(\mathbf{x}_k)$. The quadratic upper-bound in (3) for $\mu_l \in (0, 1/L]$ is such a majorized function. In proximal algorithms, the same trick is used, but it is applied to only the differentiable part of the objective function.

By simple calculations, it can be shown that (4) is equivalent to

$$\mathbf{x}_{k+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \left\{\frac{1}{2}\|\mathbf{x}-\bar{\mathbf{x}}_k\|_2^2+\mu_l g(\mathbf{x})\right\}, \tag{5}$$

where $\bar{\mathbf{x}}_k \triangleq \mathbf{x}_k - \mu_l\nabla f(\mathbf{x}_k)$. Comparing (5) with the definition of the proximal mapping, we have finally the following iterative algorithm

$$\mathbf{x}_{k+1} = \operatorname{prox}_{\mu_l g}(\mathbf{x}_k - \mu_l\nabla f(\mathbf{x}_k))\cdot$$

This algorithm is shown to converge with rate $\mathcal{O}(1/k)$ when a fixed step-size $\mu_l \in (0, 1/L]$ is used [32]. It is worth mentioning that, as said in [28] and discussed in [32], this algorithm actually converges for step-sizes smaller than $2/L$, not just $1/L$. However, the method is no longer majorization-minimization for step-sizes larger than $1/L$. Notice also that this method has been generalized to the case in which f is non-convex [33].

III. SL0: A brief review

A. *Noise-free version*

As said previously, the main idea of SL0 is to approximate the non-smooth ℓ_0 norm with a differentiable function. We denote this function by $\|\cdot\|_\sigma$, and it is defined as

$$\|\mathbf{x}\|_\sigma \triangleq n - \sum_{i=1}^{n} \exp(-\frac{x_i^2}{\sigma^2})\cdot \tag{6}$$

This smooth approximation approaches the ℓ_0 norm function as $\sigma \to 0$. With this choice, the SL0 problem is

$$\min_{\mathbf{x}}\ \|\mathbf{x}\|_\sigma \quad \text{s.t.} \quad \mathbf{y} = \mathbf{D}\mathbf{x}\cdot \tag{7}$$

The strategy of the SL0 algorithm to solve this problem is to use gradient-projection as follows. Starting with an initial estimate, which is chosen as the minimum ℓ_2 norm solution of $\mathbf{y} = \mathbf{D}\mathbf{x}$, the estimate is iteratively updated by performing one-step gradient descent on $\|\mathbf{x}\|_\sigma$ (gradient step), and then projecting the result onto the admissible set $\mathcal{A} \triangleq \{\mathbf{x}:\ \mathbf{y} = \mathbf{D}\mathbf{x}\}$ (projection step). A key point is that the non-convexity of the SL0 function (6) increases by decreasing σ. Consequently, as $\sigma \to 0$, the possibility of getting trapped into unwanted local minima is increased. To overcome this problem, in SL0 a sequence of sub-problems of the form (7) is solved, in which, σ is gradually decreased and the final solution of each sub-problem is used as a starting point for the next one. This idea is known as *warm-starting* or *continuation* in homotopy methods [34].

The SL0 algorithm is summarized in Algorithm 1. The step-size of the gradient descent is decreased along the outer-loop iterations as $\mu_\sigma \triangleq \mu\cdot\sigma^2$, in which, μ is a constant scaling factor with a default value of $\mu = 1$ [35]. Moreover, σ_0 and

Algorithm 1 SL0

```
Require: $\mathbf{y}$, $\mathbf{D}$, $\sigma_0$, $\sigma_{\min}$, $0 < c < 1$, $\mu$, $I$
Initialization: $\mathbf{x} = \mathbf{D}^{\dagger}\mathbf{y}$, $\sigma = \sigma_0$, $\mu_\sigma = \mu \cdot \sigma^2$
while $\sigma > \sigma_{\min}$ do
    for $i = 1, 2, \ldots, I$ do
        $\tilde{\mathbf{x}} = \mathbf{x} - \mu_\sigma \nabla \|\mathbf{x}\|_\sigma$          ▷ Gradient step
        $\mathbf{x} = \tilde{\mathbf{x}} - \mathbf{D}^{\dagger}(\mathbf{D}\tilde{\mathbf{x}} - \mathbf{y})$          ▷ Projection step
    end for
    $\sigma \leftarrow c \cdot \sigma$
    $\mu_\sigma = \mu \cdot \sigma^2$
end while
Output: $\mathbf{x}$
```

$\sigma_{\min}$ are the initial and the final values of σ, respectively, c is a decreasing factor for the sequence of σ's, I is the total number of iterations of the inner-loop corresponding to a particular value of σ, and $\mathbf{D}^{\dagger}$ denotes the Moore-Penrose pseudoinverse of $\mathbf{D}$.

B. Robust version

The general form of the SL0 problem, which is robust against noise, is

$$\min_{\mathbf{x}} \ \|\mathbf{x}\|_\sigma \quad \text{s.t.} \quad \|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2 \leq \epsilon, \tag{8}$$

where $\epsilon > 0$ is an upper-bound on the noise power. To solve this problem, Eftekhari *et al.* [30] proposed to modify the projection step of the noise-free SL0 (Algorithm 1) in the following way. Let $\tilde{\mathbf{x}}$ be the solution returned by the gradient step, and

$$\mathcal{A}_\epsilon \triangleq \{\mathbf{x} : \ \|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2 \leq \epsilon\} \cdot \tag{9}$$

Then, if $\tilde{\mathbf{x}} \notin \mathcal{A}_\epsilon$, it is projected onto $\mathcal{A} = \{\mathbf{x} : \ \mathbf{y} = \mathbf{D}\mathbf{x}\}$. Otherwise, it remains unchanged. As demonstrated in [30], this new algorithm is more robust than Algorithm 1.

IV. Our work

In this section, our contributions are presented. First, we revisit the SL0 algorithm by formulating the SL0 problem as the general form of a proximal problem, and deriving the SL0 algorithm using the proximal approach. In this way, the exact form of the step-size, μ_σ, involved in SL0 is derived. Moreover, using the recent works on non-convex proximal problems, the convergence of the inner-loop iterations in Algorithm 1 is established. Second, inspired by the mechanism of SL0, we propose a general family of algorithms, which we call iterative sparsification-projection (ISP).

A. SL0: revisited

The general form of the SL0 problem given in (8) can be written in the following equivalent form:

$$\min_{\mathbf{x}} \ \left\{ F_\sigma(\mathbf{x}) \triangleq \|\mathbf{x}\|_\sigma + \mathcal{I}_\epsilon(\mathbf{x}) \right\}, \tag{10}$$

where $\mathcal{I}_\epsilon$ is the indicator function of $\mathcal{A}_\epsilon$ defined as

$$\mathcal{I}_\epsilon(\mathbf{x}) \triangleq \begin{cases} 0 & \mathbf{x} \in \mathcal{A}_\epsilon \\ +\infty & \mathbf{x} \notin \mathcal{A}_\epsilon \end{cases}.$$

Now, define $f_\sigma(\mathbf{x}) \triangleq \|\mathbf{x}\|_\sigma$ and $g(\mathbf{x}) \triangleq \mathcal{I}_\epsilon(\mathbf{x})$. Then, problem (10) becomes in the form of (2), except for the fact that here f_σ is non-convex, but it is gradient Lipschitz, as will be shown by Lemma 1 below. On the other hand, since $\mathcal{A}_\epsilon$ is convex, it follows that g is convex [36]. As a result, these functions meet the requirements of the general proximal algorithms presented in [37]. The Lipschitz constant of ∇f_σ is given by the following lemma:

Lemma 1. *The function $f_\sigma(\mathbf{x}) = \|\mathbf{x}\|_\sigma$, as defined in* (6)*, is gradient Lipschitz with constant $L = \frac{2}{\sigma^2}$.*

Proof: See Appendix A.

So, the iterative proximal algorithm for solving (10) is

$$\mathbf{x}_{k+1} = \operatorname{prox}_{\mu g}(\mathbf{x}_k - \mu_\sigma \nabla f_\sigma(\mathbf{x}_k)) \cdot \tag{11}$$

Note that since g is an indicator function, its prox-operator is simply the projection onto $\mathcal{A}_\epsilon$ [28]:

$$\operatorname{prox}_g(\mathbf{x}) = \underset{\mathbf{u} \in \mathcal{A}_\epsilon}{\operatorname{argmin}} \ \|\mathbf{u} - \mathbf{x}\|_2^2 \cdot$$

The overall algorithm is the same as the SL0 algorithm outlined in Algorithm 1, except for the projection step which is now with respect to $\mathcal{A}_\epsilon$. For the convergence of the iterations in (11), we have the following theorem:

Theorem 1. *Let $\{\mathbf{x}_k\}$ be the sequence generated by* (11)*. Then, the corresponding objective values $\{F_\sigma(\mathbf{x}_k)\}$ in* (10) *are monotonically decreasing. Moreover, the sequence $\{\mathbf{x}_k\}$ converges to a stationary point of F_σ.*

Proof: See Appendix B.

An important result of the proof of the above theorem is the following guarantee for the convergence of the SL0 algorithm.

Corollary 1. *If $\mu_\sigma \in (0, \sigma^2/2]$, or equivalently, $\mu \in (0, 1/2]$ then it is guaranteed that the SL0 algorithm stated in Algorithm 1 converges to a stationary point of F_σ.*

So, the step-size has the same monotonicity behavior as the smoothing parameter σ. In other words, as also said in [27], at initial iterations, where we are far from the true solution, larger steps are taken, and as the algorithm proceeds (σ decreases), the steps become smaller and smaller until the desired solution is reached. It is worth mentioning that, although the default value for the step-size scaling factor in [35][3] is $\mu = 1$, our simulations reveal that using $\mu = 1/2$ (the maximum value to insure having a majorization-minimization like algorithm) leads to much better performance.

B. Iterative Sparsification-Projection (ISP)

Note that the gradient step in SL0 can be written as ($\mu = 1/2$)

$$x \longleftarrow \mathcal{T}_\sigma^0(x) \triangleq x \cdot (1 - \exp(-\frac{x^2}{\sigma^2})),$$

where, due to the separability of the operation, only the scalar case has been considered. The sparsification function $\mathcal{T}_\sigma^0$ along

[3] Note that, in [35], due to its different definition of the gradient term, 2μ has been defined as the scaling factor. So, its default value is actually 2.

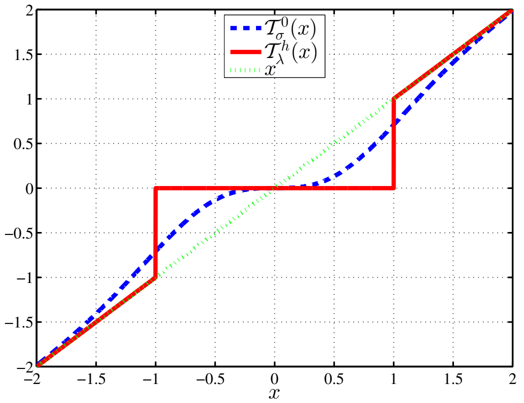


Fig. 2: SL0 sparsification *vs.* hard thresholding.

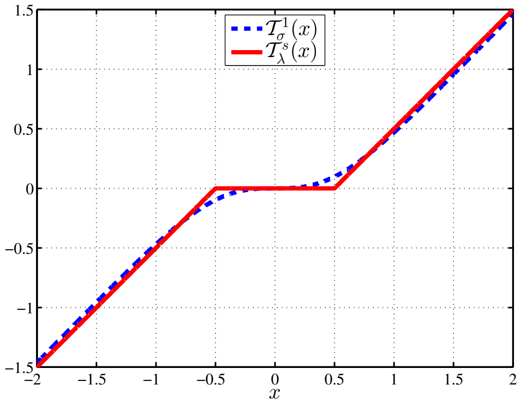


Fig. 3: SL1 sparsification *vs.* soft thresholding.

with the hard-thresholding function is plotted in Fig. 2. This figure describes the shrinkage behavior of the gradient step in SL0. Large enough inputs (compared to the σ parameter) incur no shrinkage, as in hard thresholding, while small enough ones are shrunk toward zero, hence promoting the sparsity. However, the SL0 shrinkage operation is not as crisp as hard thresholding.

Let us give another example. Consider the following scalar function

$$f^1_\sigma(x) = \frac{x^2}{\sqrt{x^2+\sigma^2}}\cdot \qquad (12)$$

When applied element-wise on a vector $\mathbf{x}$ as $F^1_\sigma(\mathbf{x}) \triangleq \sum_i f^1_\sigma(x_i)$, F^1_σ approaches the ℓ_1 norm when $\sigma \to 0$. For simplicity, we call F^1_σ smoothed ℓ_1 (SL1) norm. Using the same approach as in the proof of Lemma 1, it is straightforward to show that f^1_σ is gradient Lipschitz with constant $L = 2/\sigma$. The sparsification function $\mathcal{T}^1_\sigma(x) = x - \mu_\sigma \nabla f^1_\sigma(x)$ with $\mu_\sigma = 1/L$, along with the soft-thresholding function is plotted in Fig. 3. As can be seen, $\mathcal{T}^1_\sigma(x)$ well approximates the soft-thresholding operation. The relation between the σ parameter in $\mathcal{T}^1_\sigma$ and the threshold λ in soft-thresholding can be found by investigating the behaviors of the two functions at infinity: $x - \mu_\sigma = x - \lambda$ so $\lambda = \mu_\sigma = \sigma/2$.

These observations are not accidental. Indeed, a property of prox-operators says that: the prox-operator of a differentiable function f can be interpreted as a kind of gradient descent step for f [28]:

$$\mathrm{prox}_{\mu f}(x) \approx x - \mu \nabla f(x)\cdot \qquad (13)$$

Now, let $f(x) = |x|$. This function is not differentiable, so the above relation cannot be utilized. However, if we use the differentiable approximation of f, *i.e.*, the function $f^1_\sigma(x)$, in the right-hand side of (13), we expect the relation to hold with a good precision, especially for a small enough σ. In other words,

$$\mathrm{prox}_{\mu_\sigma f}(x) = \mathcal{T}^s_{\mu_\sigma}(x) \approx x - \mu_\sigma \nabla f^1_\sigma(x),$$

Again, the quality of this approximation is evident from Fig. 3. So, one-step gradient descent of the differentiable approximation of the ℓ_1 norm acts nearly like soft-thresholding with the threshold being equal to the step-size. The same holds for hard-thresholding. The relation between the thresholding parameter and the step-size is also like before: $\lambda = \mu_\sigma$.

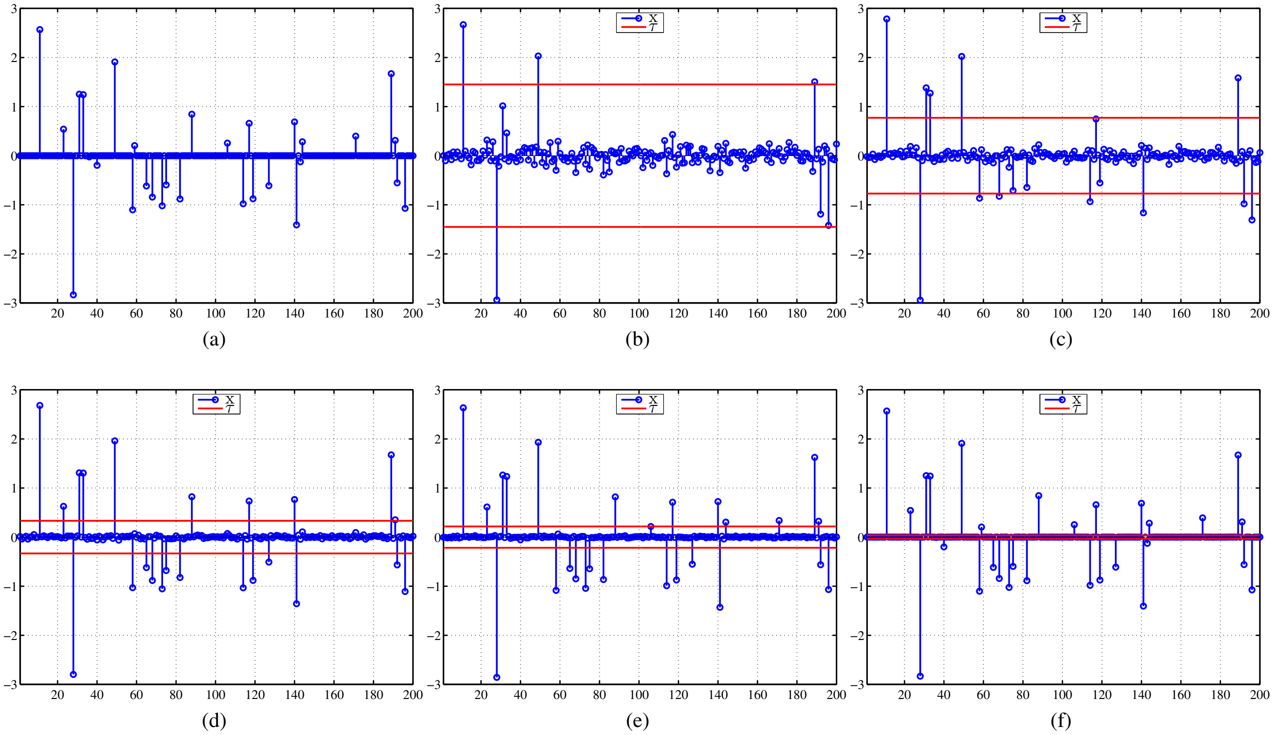
Algorithm 2 ISP

```
Require: $\mathbf{y}$, $\mathbf{D}$, $\mathcal{T}^*_\tau(.)$, $\tau_0$, $\tau_f$, $0 < c < 1$, $I$
Initialization: $\mathbf{x} = \mathbf{D}^\dagger \mathbf{y}$, $\tau = \tau_0$
while $\tau > \tau_f$ do
    for $i = 1, 2, \ldots, I$ do
        $\tilde{\mathbf{x}} = \mathcal{T}^*_\tau(\mathbf{x})$                                ▷ Sparsification step
        $\mathbf{x} = \mathrm{argmin}_{\mathbf{u} \in \mathcal{A}_\epsilon} \|\mathbf{u} - \tilde{\mathbf{x}}\|_2^2$        ▷ Projection step
    end for
    $\tau \leftarrow c \cdot \tau$
end while
Output: $\mathbf{x}$
```

Motivated by the above discussion, a general family of algorithms, dubbed iterative sparsification-projection (ISP), is introduced that follow a similar approach as in SL0. This framework is summarized in Algorithm 2, in which, $\mathcal{T}^*_\tau$ denotes a thresholding function, which can be one-step gradient descent of a smooth sparsity promoting function, *e.g.* (6), or prox-operator of a non-smooth sparsity promoting function, *e.g.* $\mathcal{T}^s_\lambda$ and $\mathcal{T}^h_\lambda$. Moreover, τ_0 and τ_f are the initial and the final values of the threshold τ, $0 < c < 1$ is a decreasing factor for τ, and I denotes the total number of inner-loop iterations corresponding to a particular value of τ.

Depending on the sparsification function used in Algorithm 2, different instances of the ISP algorithms with competing performances can be realized. Specifically, ISP-Hard, ISP-Soft, ISP-ℓ_0, and ISP-ℓ_1 correspond to $\mathcal{T}^h_\tau, \mathcal{T}^s_\tau, \mathcal{T}^0_\tau$, and $\mathcal{T}^1_\tau$ sparsifications, respectively. Gradient-based algorithms such as ISP-ℓ_0 have an additional step-size parameter to tune, while proximity-based algorithms do not have such a free parameter.

Remark 1. For the initialization of SL0, it is proposed in [27] to start with the minimum ℓ_2 norm solution. However, for the general ISP algorithms, the zero initialization has the same effect, regardless of the type of sparsification. This is because that, $\mathbf{x} = \mathbf{0}$ is not affected by the sparsification step in the first iteration, while the result after applying the projection step is exactly the minimum ℓ_2 norm solution. The threshold in the next iteration can be set to $\tau = 5 \cdot \max_i \; |x_i|$, as suggested in [27]. The threshold is then decreased along the outer-loop iterations similar to the σ parameter in SL0. The final value of



(a) (b) (c) (d) (e) (f)

Fig. 4: A few iterations of the ISP-Hard algorithm in recovery of a 25-sparse Bernoulli-Gaussian signal of length $n = 200$ from $m = 80$ Gaussian measurements. The original signal is depicted in (a), while (b)-(f) show the progress of the algorithm along iterations. The threshold level for each iteration is drawn as two horizontal red lines.

the threshold depends on the minimum absolute value of the non-zero entries of the true solution. About this final value, it is said in [27] that “In applications where the zeros in the sparsest $\mathbf{x}$ are exactly zero, σ can be decreased arbitrarily.” However, in practice, a very small σ does not affect the progress of the algorithm, because, when σ is small enough, the sparsification function becomes an identical map: $\mathcal{T}^0_\sigma(x) \approx x$.

Remark 2. The decreasing behavior of the threshold makes the ISP algorithms similar to deterministic annealing (DA) approach for solving non-convex problems [38]. DA is inspired by statistical physics, where there is an annealing process that avoids many shallow local minima of a specified cost by gradually decreasing the temperature. Similarly, DA algorithms avoid undesired local minima by gradually decreasing a problem parameter, analogues to the temperature in statistical physics processes. In ISP, the threshold plays the role of temperature in DA methods.

When there is a limitation of fixed number of iterations for the algorithm, the threshold sequence can be set as

$$\tau(j) = \tau_0 \left(\frac{\tau_f}{\tau_0}\right)^{\frac{j}{J}},$$

where j denotes the (outer-loop) iteration index, and J is the total number of iterations. In this way, the threshold starts (approximately) at τ_0 and finishes at τ_f.

Figure 4 shows the solutions of a few iterations of the ISP algorithm using hard-thresholding (ISP-Hard) for reconstructing a 25-sparse Bernoulli-Gaussian signal of length 200 from 80 random measurements. It is observed that as the threshold decreases along the iterations, the estimates’ quality is improved.

C. Sparse recovery in presence of noise

In the projection step of the ISP algorithms, the following problem has to be solved:

$$\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x}-\tilde{\mathbf{x}}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{y}-\mathbf{D}\mathbf{x}\|_2 \le \epsilon, \tag{14}$$

where $\tilde{\mathbf{x}}$ is the result of the sparsification step, and ϵ denotes an error tolerance. The idea of Eftekhari *et. al* explained in Subsection III-B for solving the robust version of SL0 is not a good method, especially when m is close to n. We will see this behavior in Section V. Therefor, we propose an alternative solution that has better performance. To this aim, we first consider the special case of $\mathbf{D}$ being a tight-frame, *i.e.*, $\mathbf{D}\mathbf{D}^T = \alpha\mathbf{I}$ for some $\alpha > 0$, and then we study the general case.

1) Tight-frame $\mathbf{D}$*:* To solve (14), first the Lagrangian function is formed as

$$L(\mathbf{x},\lambda) = \frac{1}{2}\|\mathbf{x}-\tilde{\mathbf{x}}\|_2^2 + \lambda(\|\mathbf{y}-\mathbf{D}\mathbf{x}\|_2^2 - \epsilon^2), \tag{15}$$

in which, λ is the Lagrangian multiplier. Karush-Kuhn-Tucker (KKT) conditions are used to derive the following optimality conditions:

$$\begin{cases} \mathbf{x}^* = (\mathbf{I} + \lambda^* \cdot \mathbf{D}^T\mathbf{D})^{-1}(\tilde{\mathbf{x}} + \lambda^* \cdot \mathbf{D}^T\mathbf{y}) \\ \|\mathbf{y}-\mathbf{D}\mathbf{x}^*\|_2^2 = \epsilon^2 \\ \lambda^* \ge 0 \end{cases} \tag{16}$$

This leads to the following nonlinear equation for λ^*:

$$\|\mathbf{y}-\mathbf{D}(\mathbf{I}+\lambda^*\mathbf{D}^T\mathbf{D})^{-1}(\tilde{\mathbf{x}}+\lambda^*\mathbf{D}^T\mathbf{y})\|_2^2=\epsilon^2\cdot \qquad (17)$$

This equation does not have a closed-form solution in general, but in the special case where $\mathbf{D}$ is a tight-frame, we can find a closed-form solution. Tight-frame matrices are often of interest in compressed sensing applications for their fast computations. For instance, submatrices of the discrete Fourier transform (DFT), the discrete cosine transform (DCT), the Hadamard transform, and the noiselet transform are tight-frames [20]. Moreover, it has been shown in [39] that unit-norm tight-frames (tight-frames that have unit-norm columns) lead to good mean squared error (MSE) performance in compressed sensing. Using the tight-frame assumption for $\mathbf{D}$ and applying the Woodbury matrix inversion lemma [40], the matrix inversion term in $\mathbf{x}^*$ can be rewritten as

$$(\mathbf{I}+\lambda^*\mathbf{D}^T\mathbf{D})^{-1}=\mathbf{I}-\frac{\lambda^*}{(1+\lambda^*\alpha)}\mathbf{D}^T\mathbf{D}, \qquad (18)$$

which by inserting into (17) and solving for λ^*, while considering $\mathbf{D}\mathbf{D}^T=\alpha\mathbf{I}$, leads to the following formulas:

$$\begin{cases}\lambda^*=\frac{1}{\alpha}\max(\frac{\|\mathbf{y}-\mathbf{D}\tilde{\mathbf{x}}\|_2}{\epsilon}-1,0)\\ \mathbf{x}^*=\tilde{\mathbf{x}}+\frac{\lambda^*}{1+\lambda^*\alpha}\mathbf{D}^T(\mathbf{y}-\mathbf{D}\tilde{\mathbf{x}})\end{cases}. \qquad (19)$$

2) General $\mathbf{D}$*:* In the general case, we use alternating direction method of multipliers (ADMM) [41] to solve problem (14). Toward this goal, consider the following equivalent form of (14)

$$\min_{\mathbf{x},\mathbf{z}}\ \frac{1}{2}\|\mathbf{x}-\tilde{\mathbf{x}}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{z}\|_2\leq\epsilon,\ \mathbf{z}=\mathbf{y}-\mathbf{D}\mathbf{x}, \qquad (20)$$

in which $\mathbf{z}$ is an auxiliary variable. Then, the augmented Lagrangian function of the above problem is formed as[4]

$$L(\mathbf{x},\mathbf{z},\boldsymbol{\lambda})=\frac{1}{2}\|\mathbf{x}-\tilde{\mathbf{x}}\|_2^2-\boldsymbol{\lambda}^T(\mathbf{z}-\mathbf{y}+\mathbf{D}\mathbf{x})+\frac{\gamma}{2}\|\mathbf{z}-\mathbf{y}+\mathbf{D}\mathbf{x}\|_2^2, \qquad (21)$$

in which, $\gamma>0$ is a penalty parameter which controls the convergence rate of the algorithm. This function is iteratively minimized over $\mathbf{x}$ and $\mathbf{z}$, in a Gauss-Seidel manner, followed by update of the Lagrange multiplier vector $\boldsymbol{\lambda}$. The update problems are as follows

$$\begin{cases}\mathbf{z}_{k+1}=\operatorname{argmin}_{\mathbf{z}:\ \|\mathbf{z}\|_2\leq\epsilon}\ L(\mathbf{x}_k,\mathbf{z},\boldsymbol{\lambda}_k)\\ \mathbf{x}_{k+1}=\operatorname{argmin}_{\mathbf{x}}\ L(\mathbf{x},\mathbf{z}_{k+1},\boldsymbol{\lambda}_k)\\ \boldsymbol{\lambda}_{k+1}=\boldsymbol{\lambda}_k-\gamma(\mathbf{z}_{k+1}-\mathbf{y}+\mathbf{D}\mathbf{x}_{k+1})\end{cases}. \qquad (22)$$

It can be easily shown that the $\mathbf{z}$-update problem is

$$\mathbf{z}_{k+1}=\underset{\mathbf{z}:\ \|\mathbf{z}\|_2\leq\epsilon}{\operatorname{argmin}}\ \frac{1}{2}\|\mathbf{z}-\mathbf{y}+\mathbf{D}\mathbf{x}_k-\frac{1}{\gamma}\boldsymbol{\lambda}_k\|_2^2, \qquad (23)$$

which admits the following solution

$$\mathbf{z}_{k+1}=\mathcal{P}_{\mathcal{A}_\mathbf{z}}(\mathbf{y}-\mathbf{D}\mathbf{x}_k+\frac{1}{\gamma}\boldsymbol{\lambda}_k), \qquad (24)$$

where, $\mathcal{P}_{\mathcal{A}_\mathbf{z}}(.)$ is the projection onto $\mathcal{A}_\mathbf{z}=\{\mathbf{z}:\ \|\mathbf{z}\|_2\leq\epsilon\}$. That is,

$$\mathcal{P}_{\mathcal{A}_\mathbf{z}}(\mathbf{z})\triangleq\begin{cases}\mathbf{z} & \mathbf{z}\in\mathcal{A}_\mathbf{z}\\ \frac{\epsilon}{\|\mathbf{z}\|_2}\cdot\mathbf{z} & \text{oth.}\end{cases}$$

Algorithm 3 Projection onto $\mathcal{A}_\epsilon$ using ADMM

```
Require: $\mathbf{y}$, $\mathbf{D}$, $\tilde{\mathbf{x}}$, $\gamma>0$
Initialization: $\mathbf{x}=\tilde{\mathbf{x}}$, $\boldsymbol{\lambda}=\mathbf{0}$, set $\mathbf{A}=(\mathbf{I}+\gamma\mathbf{D}^T\mathbf{D})^{-1}$
while $\|\mathbf{y}-\mathbf{D}\mathbf{x}\|_2>\epsilon$ do
    $\mathbf{z}=\mathcal{P}_{\mathcal{A}_\mathbf{z}}(\mathbf{y}-\mathbf{D}\mathbf{x}+\frac{1}{\gamma}\boldsymbol{\lambda})$
    $\mathbf{x}=\mathbf{A}(\tilde{\mathbf{x}}+\gamma\mathbf{D}^T(\mathbf{y}-\mathbf{z}+\frac{1}{\gamma}\boldsymbol{\lambda}))$
    $\boldsymbol{\lambda}=\boldsymbol{\lambda}-\gamma(\mathbf{z}-\mathbf{y}+\mathbf{D}\mathbf{x})$
end while
Output: $\mathbf{x}$
```

The update problem for $\mathbf{x}$ has also the following closed-form solution:

$$\mathbf{x}_{k+1}=(\mathbf{I}+\gamma\mathbf{D}^T\mathbf{D})^{-1}(\tilde{\mathbf{x}}+\gamma\mathbf{D}^T(\mathbf{y}-\mathbf{z}_{k+1}+\frac{1}{\gamma}\boldsymbol{\lambda}_k))\cdot \qquad (25)$$

The final projection algorithm is summarized in Algorithm 3.

V. Simulation results

A number of numerical experiments on recovery of sparse and compressible signals from their compressed linear measurements (in the compressed sensing application), and sparse approximation of natural image patches were conducted to evaluate the performance of the ISP algorithms, and to compare them with some well-known algorithms, including GOMP[5] [22], EM-GM-AMP[6] [17], and NESTA[7] [20]. GOMP is a generalization of OMP, in the sense that "multiple" atoms are picked up per iteration. As demonstrated in [22], GOMP has a faster convergence and a better performance than OMP. The family of approximate message passing (AMP) algorithms are simple but efficient extensions of iterative shrinkage-thresholding algorithms [11], and are inspired by belief propagation in graphical models [15]. In particular, EM-GM-AMP is a message passing algorithm that first models the distribution of the signal's non-zero coefficients as a Gaussian mixture, and then learns the model parameters through expectation maximization, using generalized AMP (GAMP) [16] to implement the expectation step. The EM-GM-AMP algorithm shows state-of-the-art performance in compressed sensing, as confirmed by the simulations performed in [17]. Finally, NESTA, is an ℓ_1 norm-based sparse recovery algorithm that directly solves P_1^ϵ using Nesterov's smoothing idea [20]. For all the algorithms, their available MATLAB packages were used.

To measure the performances of the algorithms, the following quantities were used:

- The normalized mean squared error (NMSE) between the true sparse signal $\mathbf{x}^*$ and the estimated one $\hat{\mathbf{x}}$:

$$\text{NMSE}(\mathbf{x}^*,\hat{\mathbf{x}})\triangleq\frac{\|\mathbf{x}^*-\hat{\mathbf{x}}\|_2}{\|\mathbf{x}^*\|_2}.$$

- The Gini index (GI) [42]. For a discrete signal $\mathbf{x}$, GI($\mathbf{x}$) is a robust measure of the sparsity of $\mathbf{x}$. In contrast to

[4] Note that the constraint $\|\mathbf{z}\|_2\leq\epsilon$ has not been included in the Lagrangian function. However, when minimizing over $\mathbf{z}$, this constraint will be considered.

[5] http://islab.snu.ac.kr/paper/gOMP.zip

[6] http://www2.ece.ohio-state.edu/∼schniter/EMGMAMP/EMGMAMP.html

[7] http://statweb.stanford.edu/∼candes/nesta/

conventional norm measures, such as ℓ_0 and ℓ_1, GI is normalized between 0 and 1, with 0 corresponding to the least sparse signal comprising from equal energy entries, and 1 for the most sparse signal with all of its energy concentrated in only one entry. Moreover, GI is scale invariant and independent of the length of the signal. For a signal $\mathbf{x} = [x_1, \cdots, x_n]^T$, define its re-ordered version as $\bar{\mathbf{x}} = [\bar{x}_1, \cdots, \bar{x}_n]$ where $|\bar{x}_1| \leq |\bar{x}_2| \leq \cdots |\bar{x}_n|$. The GI of $\mathbf{x}$ is then defined as

$$\mathrm{GI}(\mathbf{x}) \triangleq 1 - \frac{2}{\|\mathbf{x}\|_1} \sum_{i=1}^{n} \frac{n-i+1/2}{n} \cdot |\bar{x}_i|.$$

Moreover, to compare the computational complexities of the algorithms, their runtimes were used as a rough measure. Our simulations were performed on a 64 bit Windows 7 operating system with 8 GB RAM and an intel core i7 CPU.

The parameters of the algorithms were set as follows. For the ISP algorithms, $\tau_0 = 5 \max_i |x_i^0|$, where $\mathbf{x}^0$ is the minimum ℓ_2 norm solution of $\mathbf{y} = \mathbf{D}\mathbf{x}$, which as explained earlier is the result of the second iteration of the algorithms initialized by the zero signal, $\tau_f = 5 \times 10^{-4}$, $c = 0.9$, and $I = 3$. In addition, the penalty parameter γ in Algorithm 3 was set to 0.4. For GOMP, the number of selected atoms in each iteration was set to $N = 4$. Moreover, the algorithm stopped whenever the iteration number reached a maximum, which we set to $K = m$, or the ℓ_2 norm of the residual fell below a given threshold determined by the noise level (for more details, see Table I of [22]). For EM-GM-AMP, the parameters suggested in [17] were used. Finally, in NESTA, except for the final value of the smoothing parameter that was set to $\mu_f = 10^{-5}$, the remaining parameters were set to their default values. These parameters are fixed throughout the simulations.

The rest of this section is organized as follows. In Subsection V-A, the simulation results demonstrating the performances of the ISP algorithms, the effect of the step-size scaling factor μ in SL0, and the performances of Eftekhari's robust projection and Algorithm 3 in noisy settings, are presented. Then, Subsection V-B compares the performances of ISP-Hard, GOMP, EM-GM-AMP, and NESTA.

A. *Comparison of the ISP algorithms*

In this subsection, the results of comparing ISP-Hard, ISP-Soft, ISP-ℓ_0, and ISP-ℓ_1 are reported. As a common practice in the literature, synthetically generated data were used to examine the performances of the algorithms in compressed sensing. To this end, the measurement matrix $\mathbf{D}$ was generated by randomly and independently choosing its entries from the normal distribution $\mathcal{N}(0, 1)$. The sparse signal $\mathbf{x}$, of length $n = 1000$ and with $s \in \{50, 100, 150, 200\}$ non-zero entries, was generated using a Bernoulli-Gaussian distribution as follows. The locations of the non-zero entries were sampled uniformly at random, and their values were selected from $\mathcal{N}(0, 1)$. The measurement vector $\mathbf{y}$, of length $m = 400$, was then generated as $\mathbf{y} = \mathbf{D}\mathbf{x} + \mathbf{e}$, where $\mathbf{e}$ is a Gaussian noise vector with $\mathcal{N}(0, \sigma_{noise}^2)$ distributed entries. Each experiment was repeated 500 times and the average results were reported.

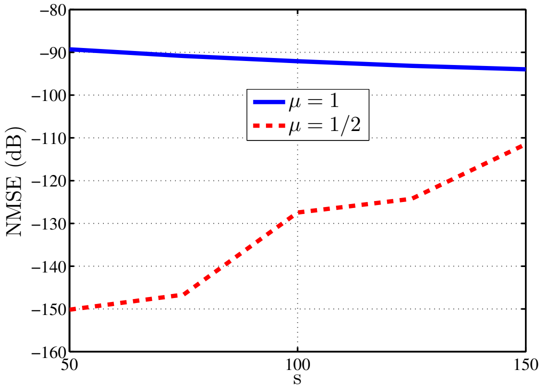
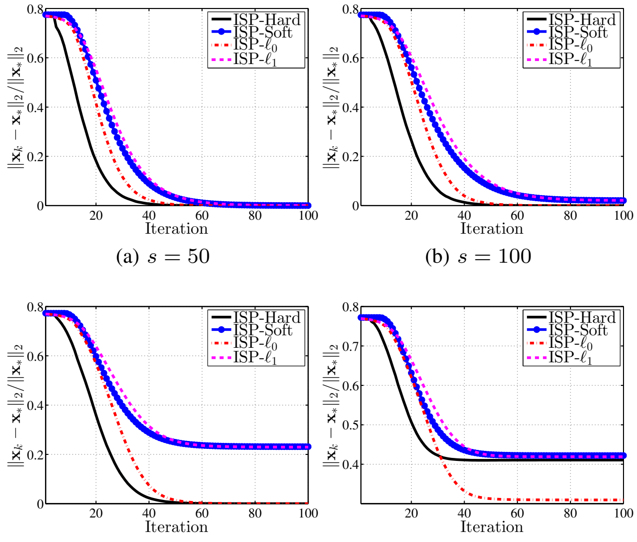


Fig. 5: NMSE (dB) of the ISP-ℓ_0 algorithm *vs.* the number of non-zero entries, s, in recovery of Bernoulli-Gaussian signals from Gaussian measurements for $\mu = 1$ and $\mu = 1/2$ ($m = 400$ and $n = 1000$).



(a) $s = 50$

(b) $s = 100$

(c) $s = 150$

(d) $s = 200$

Fig. 6: Normalized errors *vs.* iterations of the ISP algorithms in noiseless recovery of Bernoulli-Gaussian signals from Gaussian measurements with different number of non-zeros, s. For this experiment, $m = 400$ and $n = 1000$.

1) The effect of μ: Here, the performances of ISP-ℓ_0 for $\mu = 1$ suggested in [35], and $\mu = 1/2$, which is proposed in this paper, are compared. Figure 5 shows the NMSE values (in dB) versus the number of non-zero entries of $\mathbf{x}$. As can be clearly seen, the choice $\mu = 1/2$ corresponding to the upper-bound of being a majorization-minimization algorithm leads to much better results than $\mu = 1$. Therefor, we used $\mu = 1/2$ in ISP-ℓ_0 and ISP-ℓ_1 for the simulations.

2) Convergence behaviors: The normalized errors' evolutions of the ISP algorithms versus iterations for different sparsity levels and $\sigma_{noise} = 0$ are compared in Fig. 6. The final values of the normalized errors are also reported in Table I. As demonstrated, ISP-Hard has the best convergence rate, and except for large values of s, its final normalized errors are the lowest among the others. Notice also that, as expected, ISP-Soft and ISP-ℓ_1 have poor performances for large s.

The sparsity of the solutions (measured by GI) over iterations are shown in Fig. 7 for $s = 150$. As illustrated, the sparsity increases along the iterations for all the algorithms. Again, the ℓ_1 norm-based algorithms achieved less sparse solutions compared to the ℓ_0 norm-based ones, which was expected. From the computational complexity aspect, all the algorithms had nearly the same runtimes.

TABLE I: NMSE values for the ISP algorithms in noiseless recovery of Bernoulli-Gaussian signals of length $n = 1000$ from their $m = 400$ linear Gaussian measurements.

Algorithm	$s = 50$	$s = 100$	$s = 150$	$s = 200$
ISP-Hard	**3.4623e-08**	**1.9449e-08**	**4.6082e-07**	**0.4105**
ISP-ℓ_0	**9.1372e-08**	**6.3223e-07**	**1.5820e-06**	**0.3095**
ISP-Soft	**1.7035e-04**	**0.0206**	**0.2312**	**0.4219**
ISP-ℓ_1	**2.1137e-04**	**0.0214**	**0.2294**	**0.4191**

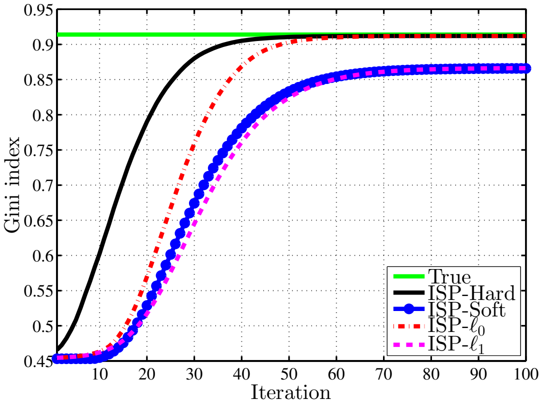
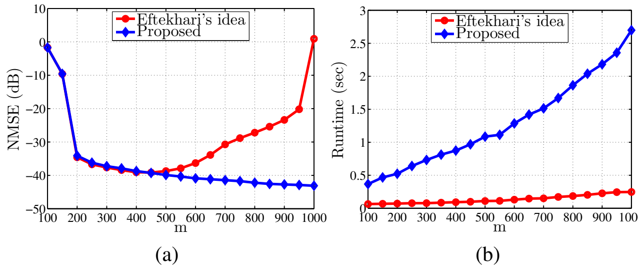


Fig. 7: Gini index *vs.* iterations of the ISP algorithms in noiseless recovery of Bernoulli-Gaussian signals of length $n = 1000$ that have $s = 150$ non-zero entries, from their $m = 400$ linear Gaussian measurements.

3) Robust recovery: To compare our proposed robust projection (Algorithm 3) with that of Eftekhari *et al.* [30], we conducted an experiment, in which, the sparse signal had 50 non-zero entries out of 1000, the noise standard deviation was $\sigma_{noise} = 0.005$, and the number of measurements, m, was variable. Figure 8 shows the results for ISP-ℓ_0. It is observed that the performance of Eftekhari's method deteriorates as the number of measurements increases. However, Eftekhari's method has a lower computational complexity. It is also noticeable that the two ideas have similar performances when the measurement matrix is highly overcomplete, with Eftekhari's method being faster. This suggests to use Eftekhari's projection for these situations. This behavior is mainly due to the properties of the measurement matrix, *e.g.,* its rank and condition number, and will be explored further in Subsection V-B-2.

B. Comparison with GOMP, EM-GM-AMP, and NESTA

In this subsection, the ISP-Hard algorithm, which showed a better performance than the other ISP algorithms, is compared with GOMP, EM-GM-AMP, and NESTA in different scenarios. First, we present the phase transitions (see [15]) of the algorithms. A phase transition describes the 2D success/failure regions of an algorithm in terms of the sparsity and the undersampling ratios, defined as $\rho \triangleq s/m$ and $\delta \triangleq m/n$, respectively. Next, the performances of the algorithms are compared for compressible signals, different measurement matrices, and sparse approximation of natural image patches.



(a)

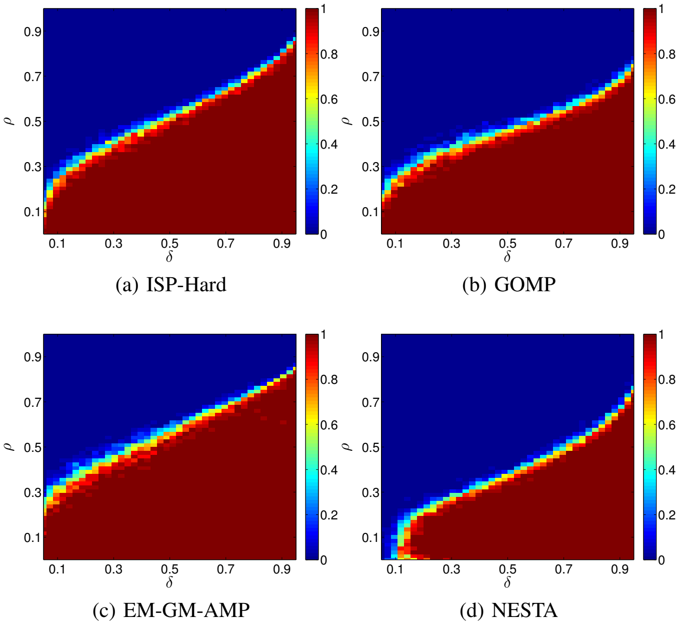
(b)

Fig. 8: (a) NMSE (dB) and (b) runtimes (sec) of ISP-ℓ_0 algorithm equipped with Eftekhari's projection and our proposed one (Algorithm 3) *vs.* the number of Gaussian measurements. The underlying signal is Bernoulli-Gaussian of length $n = 1000$ with $s = 50$ non-zeros.

1) Phase transition: To construct the phase transition diagrams, the sparsity–undersampling parameter space was divided into a 40×60 grid within the region specified by $\delta \in [0.05, 0.95]$ and $\rho \in [0.01, 0.99]$. The matrix $\mathbf{D}$ and the sparse signal $\mathbf{x}$ were generated in the same way as described in Subsection V-A. A recovery was declared successful if NMSE ≤ 0.001. At each grid point, the success rates were computed over 100 realizations. Phase transition diagrams for different algorithms are shown in Fig. 9, while the phase transition curves (PTCs) that separate the regions with success rates above and below 0.5 are illustrated in Fig. 10. This figure also includes the theoretical LASSO PTC [15]. As depicted, EM-GM-AMP has the highest PTC, meaning that, it successfully recovers Bernoulli-Gaussian signals in a wider region of the $\delta - \rho$ space than the other algorithms. Moreover, ISP-Hard and GOMP have similar performances for small δ and ρ, while for other values, ISP-Hard outperforms GOMP. The empirical performance of NESTA is also comparable to the theoretical LASSO, except for small δ and ρ. Figure 11 compares the averaged runtimes of the algorithms versus both the sparsity ratio ρ and the undersampling ratio δ. As demonstrated, while ISP-Hard has an almost constant runtime over various sparsity levels, it takes more times for the other algorithms to recover less sparse signals. In terms of the number of measurements m, however, almost all the algorithms have increasing runtimes with m. In addition, ISP-Hard has the lowest runtime in average.

2) Different measurement matrices: Similar to [43], the recovery performances of the algorithms are compared under the following four types of the measurement matrix $\mathbf{D}$:

- **Sparse**: A Gaussian matrix generated from $\mathcal{N}(0, 1)$ that a portion of its entries were uniformly set to zero.
- **Non-zero mean**: A matrix generated from a non-zero



(a) ISP-Hard (b) GOMP

(c) EM-GM-AMP (d) NESTA

Fig. 9: Phase transitions for noiseless recovery of Bernoulli-Gaussian signals from Gaussian measurements, which indicate successful recovery rates, in terms of the sparsity ratio $\rho \triangleq s/m$ and the undersampling ratio $\delta \triangleq m/n$, for $n = 1000$.

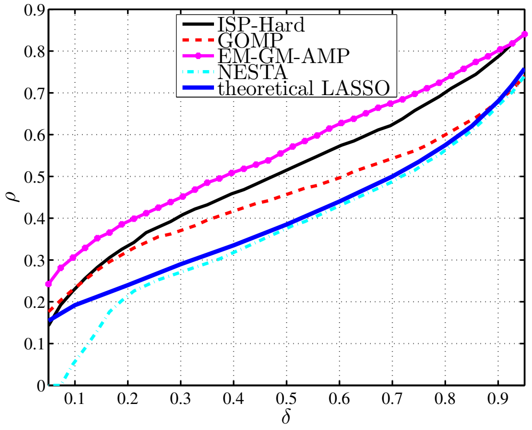
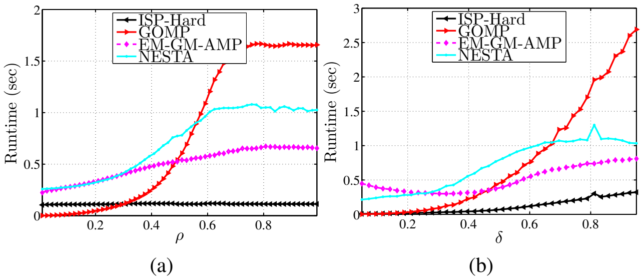


Fig. 10: Phase transition curves (PTCs) that divide the phase transition diagrams depicted in Fig. 9 into two regions corresponding to success rates above and below 0.5. The theoretical LASSO PTC [15] is also included for comparison.

mean, unit-variance Gaussian distribution.

- **Ill-conditioned**: A matrix $\mathbf{D}$ with singular value decomposition (SVD) as $\mathbf{D} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ that the diagonal entries of $\mathbf{\Sigma}$ are $\Sigma_{ii} = \kappa^{(m-i)/m}$, where $\kappa > 1$ is the condition number of $\mathbf{D}$.
- **Low-rank**: A matrix $\mathbf{D} = \frac{1}{m}\mathbf{U}_{m\times r}\mathbf{V}^T_{r\times n}$, where r is the rank of $\mathbf{D}$, and the entries of $\mathbf{U}$ and $\mathbf{V}$ are drawn from $\mathcal{N}(0,1)$.

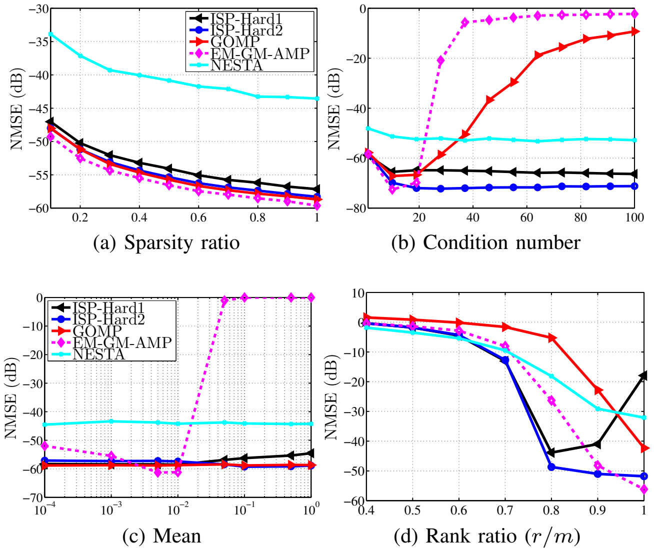
We considered recovery of Bernoulli-Gaussian sparse signals of length 1000 with 150 non-zero entries, from 500 noisy measurements ($\sigma_{noise} = 0.0005$) taken with measurement matrices of the above four types. Furthermore, except for the sparse type, we set `optEM.robust_gamp=true` in EM-GM-AMP, which makes the algorithm more robust to problematic measurement matrices including low-rank, ill-conditioned, and non-zero mean.



(a) (b)

Fig. 11: Runtimes of different algorithms, for noiseless recovery of Bernoulli-Gaussian signals, in terms of: (a) the sparsity ratio $\rho \triangleq s/m$, and (b) the undersampling ratio $\delta \triangleq m/n$, with $n = 1000$.

The averaged NMSEs (dB) over 100 realizations are illustrated in Fig. 12. In this figure, ISP-Hard1 and ISP-Hard2 denote the ISP-Hard algorithm equipped with Eftekhari's robust projection and Algorithm 3, respectively. As shown in this figure, EM-GM-AMP is very sensitive to ill-conditioned and non-zero mean measurement matrices. Moreover, GOMP does not work well for low-rank and ill-conditioned matrices, while it has close performances to ISP-Hard2 for other types of matrices. Another noticeable point is that, ISP-Hard1 has inferior performances to ISP-Hard2, especially for low-rank product matrices, as confirmed by Fig. 12 (d).



(a) Sparsity ratio (b) Condition number

(c) Mean (d) Rank ratio (r/m)

Fig. 12: NMSEs (dB) in recovery of Bernoulli-Gaussian signals with measurement matrix $\mathbf{D}$ of types: (a) sparse, (b) ill-conditioned, (c) non-zero mean, and (d) low-rank. The signal length is $n = 1000$, with $s = 150$ non-zeros, and the number of measurements is $m = 500$.

3) Compressible signals: A signal $\mathbf{x} \in \mathbb{R}^n$ is called compressible if its sorted coefficients $\{\bar{x}_i\}$ exhibit a power-law decay as $|\bar{x}_i| \lesssim R \cdot i^{-d}$, where $R > 0$, and $d > 0$ is the decay rate [44]. A set of probability distributions were described in [44] whose *i.i.d.* realizations result in compressible signals. Generalized Pareto distribution (GPD) is such a compressible signal prior, whose probability density function (pdf) is given by

$$P(x; q, \lambda) = \frac{q}{2\lambda}(1 + \frac{|x|}{\lambda})^{-(q+1)}. \tag{26}$$

It has been shown in [44] that $R = \lambda n^{1/q}$ and $d = 1/q$, and that, wavelet coefficients of natural images can be well approximated by this distribution.

We chose GPD, and used the MATLAB code available in ‘http://dsp.rice.edu/randcs’ to produce compressible signals of length $n = 1000$. The scaling parameter λ was set to 1, and the compressibility parameter q was changed from 0.1 to 1.5. The performances of different algorithms were then evaluated in recovery of compressible signals from a number of $m = 500$ Gaussian measurements.

Figure 13 compares the averaged NMSEs (dB) (over 100 realizations) versus q, where GI values corresponding to each q are also plotted. As can be seen, GOMP has the best recovery performance among the others. Also, ISP-Hard outperforms EM-GM-AMP. However, all the algorithms fail to successfully recover less compressible signals.

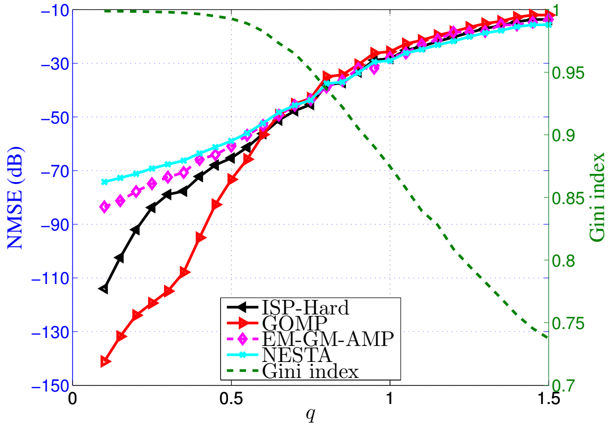


Fig. 13: NMSEs (dB) and GIs in recovery of compressible signals *vs.* the compressibility parameter q. The signal length is $n = 1000$, and a total of $m = 500$ measurements were taken.

4) Sparse approximation of image patches: Here, we consider a sparse decomposition problem, where the ability of the algorithms in sparsely approximating natural image patches within a certain error bound are compared. To this end, 1000 blocks of size 8 × 8 from some well-known images, including Lena, Barbara, Cameraman, and House were randomly extracted. The blocks were then converted to 64-dimensional vectors. The dictionaries over which the blocks were decomposed are 64 × 64 and 64 × 256 DCT. The approximation error's upper bound was set to $\epsilon = 0.005$. For EM-GM-AMP, in addition to the suggested parameters in [17], we set `optEM.learn_noisevar=false` and `optEM.noise_var=`ϵ^2.

TABLE II: Decomposition errors and GIs in approximating natural 8×8 image blocks over DCT dictionaries of sizes 64×64 and 64×256, with an error constraint of 0.005. For each algorithm, top cells denote decomposition errors and bottom cells correspond to GIs.

Algorithm	64 × 64	64 × 256
ISP-Hard1	**0.0042** ± 7.9350E − 04	**0.0049**±8.3817E − 05
	0.6983±0.0986	**0.9500**±0.0154
ISP-Hard2	**0.0050**±3.2035E − 05	**0.0050**±1.5489E − 05
	0.7139±0.1003	**0.9513**±0.0151
GOMP	**0.0044**±0.00195	**0.0047**±2.0694E − 04
	0.7123±0.1027	**0.9507**±0.0149
EM-GM-AMP	**0.0062**±0.0049	**22.8665**±30.4031
	0.6991±0.0901	**0.8693**±0.0805
NESTA	**0.0050**±2.3100E − 09	**0.0050**±8.4619E − 10
	0.7009±0.0975	**0.9133**±0.0283

The averaged approximation errors and GIs of the representation vectors are reported in Table II. Examining the results reveals that ISP-Hard2 has better GIs than the other algorithms, while it satisfies the error constraint with a good precision. Moreover, EM-GM-AMP failed in maintaining the decomposition errors lower than $\epsilon = 0.005$, especially for the 64 × 256 dictionary. In addition, ISP-Hard1 has a poor performance in the complete DCT dictionary. It is also worth mentioning that ISP-Hard1's error converged to the true one in the overcomplete DCT dictionary.

VI. Conclusion

In this paper, we addressed the sparse recovery problem. We first investigated the already-proposed SL0 algorithm from the proximal algorithmic framework. Using this, we shed some lights on SL0, including the determination of its step-size parameter and providing a convergence guarantee for it. In the sequel, inspired by the mechanism of SL0 and the proximal algorithms, a general family of algorithms, called iterative-sparsification-projection (ISP), were introduced, which possesses SL0 as a special case. Moreover, to solve the projection step of the ISP algorithms in noisy cases, a new algorithm was proposed. Through a set of extensive experiments on sparse signal recovery from compressed measurements in various scenarios, and sparse approximation of natural image patches, it was demonstrated that in most cases, our new algorithms outperform a number of well-known algorithms, including the state-of-the-art EM-GM-AMP algorithm [17].

Appendix A
Proof of Lemma 1

To derive the Lipschitz constant of the gradient of $f_\sigma(\mathbf{x}) = \|\mathbf{x}\|_\sigma$, we first prove the following lemma:

Lemma 2. *Let $F(\mathbf{x}) \triangleq \sum_{i=1}^{n} f(x_i)$, in which dom$F = \mathbb{R}^n$ and the derivative of $f : \mathbb{R} \longrightarrow \mathbb{R}$ is Lipschitz continuous with constant L. Then, F is gradient Lipschitz with constant L.*

Proof: The gradient of F is given by

$$\nabla F(\mathbf{x}) = [f^{'}(x_1), \cdots, f^{'}(x_n)]^T.$$

Then, for all $\mathbf{x}, \mathbf{z} \in \mathbb{R}^n$ we have

$$\begin{aligned}\|\nabla F(\mathbf{x}) - \nabla F(\mathbf{z})\|_2 =& \sqrt{\sum_{i=1}^{n}(f'(x_i) - f'(z_i))^2} \\ \leq& \sqrt{\sum_{i=1}^{n} L^2 \cdot (x_i - z_i)^2} \\ =& L \cdot \|\mathbf{x} - \mathbf{z}\|_2,\end{aligned}$$

which concludes the proof. ■

Now, $f_\sigma(\mathbf{x}) = \sum_{i=1}^{n}(1 - \exp(-\frac{x_i^2}{\sigma^2}))$. Thus, using the above lemma, we only need to derive the Lipschitz constant of the derivative of $f(x) = 1 - \exp(-\frac{x^2}{\sigma^2})$. For the second derivative of f, which is given by

$$f^{''}(x) = \frac{2}{\sigma^2}(1 - \frac{2x^2}{\sigma^2})\exp(-\frac{x^2}{\sigma^2}),$$

we have $\forall x: \; |f^{''}(x)| \leq (2/\sigma^2)$, which by using the mean value theorem shows that $f^{'}$ is Lipschitz with constant $L = 2/\sigma^2$. Consequently, the Lipschitz constant of ∇f_σ is $2/\sigma^2$.

Appendix B
Proof of Theorem 1

To prove Theorem 1, we borrow ideas from recent works on convergence analysis of proximal methods for non-convex problems [33], [45]. To begin with, recall our target problem

$$\min_{\mathbf{x}} \; \Big\{ F_\sigma(\mathbf{x}) \triangleq f_\sigma(\mathbf{x}) + g(\mathbf{x}) \Big\}, \tag{27}$$

where $f_\sigma(\mathbf{x}) = \|\mathbf{x}\|_\sigma$ and $g(\mathbf{x}) = \mathcal{I}_\epsilon(\mathbf{x})$, and its associated algorithm

$$\mathbf{x}_{k+1} = \operatorname*{argmin}_{\mathbf{x}} \left\{ \nabla f_\sigma(\mathbf{x}_k)^T(\mathbf{x} - \mathbf{x}_k) + \frac{1}{2\mu_\sigma}\|\mathbf{x} - \mathbf{x}_k\|_2^2 + g(\mathbf{x}) \right\} \tag{28}$$

Before presenting the proof, first notice the following necessary definitions and useful lemmas.

Definition 3 ([46], [47]). *The Fréchet subdifferential of a function g at $\mathbf{x} \in \mathbb{R}^n$, denoted by $\hat{\partial} g(\mathbf{x})$, is defined as*

$$\begin{aligned}\hat{\partial} g(\mathbf{x}) \triangleq \Bigg\{ \zeta \in \mathbb{R}^n | \; \liminf_{\mathbf{v}\to\mathbf{x}_{\mathbf{v}\neq\mathbf{x}}} \; \frac{1}{\|\mathbf{x} - \mathbf{v}\|_2^2} \cdot \\ \Big(g(\mathbf{v}) - g(\mathbf{x}) - \langle \mathbf{v} - \mathbf{x}, \zeta \rangle \Big) \geqslant 0 \Bigg\}\end{aligned} \tag{29}$$

Definition 4 ([47]). *The limiting-subdifferential of a proper, lower semi-continuous function g at $\mathbf{x} \in \mathbb{R}^n$, denoted by $\partial g(\mathbf{x})$, is defined as*

$$\begin{aligned}\partial g(\mathbf{x}) \triangleq \Bigg\{ \zeta \in \mathbb{R}^n | \; \exists \; \mathbf{x}_k \to \mathbf{x}, \\ g(\mathbf{x}_k) \to g(\mathbf{x}), \zeta_k \to \zeta, \zeta_k \in \hat{\partial} g(\mathbf{x}_k) \Bigg\}\end{aligned} \tag{30}$$

Proposition 1 ([37]). *Let $\{(\mathbf{x}_k, \mathbf{u}_k)\}_{k=0}^{\infty}$ be a sequence in Graph$(\partial g) \triangleq \{(\mathbf{z}, \mathbf{v}) \mid \; \mathbf{v} \in \partial g(\mathbf{z})\}$ that converges to $(\mathbf{x}, \mathbf{u})$ as $k \to \infty$. By the definition of ∂g, if $g(\mathbf{x}_k)$ converges to $g(\mathbf{x})$ as $k \to \infty$, then $(\mathbf{x}, \mathbf{u}) \in$ Graph(∂g).*

Now, we are ready to prove the theorem. Since $\mathbf{x}_{k+1}$ is the minimizer of (28), optimality conditions imply that

$$\begin{aligned}\nabla f_\sigma(\mathbf{x}_k)^T(\mathbf{x}_{k+1} - \mathbf{x}_k) + \frac{1}{2\mu_\sigma}\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2^2 + g(\mathbf{x}_{k+1}) \\ \leq g(\mathbf{x}_k)\end{aligned} \tag{31}$$

and

$$0 \in \partial g(\mathbf{x}_{k+1}) + \nabla f_\sigma(\mathbf{x}_k) + \frac{1}{\mu_\sigma}(\mathbf{x}_{k+1} - \mathbf{x}_k), \tag{32}$$

where in (32) the following lemma has been used [45]:

Lemma 3. *Let $h = f + g$, where f is continuously differentiable and g is convex. Then, $\forall \mathbf{x} \in domh$*

$$\partial h(\mathbf{x}) = \nabla f(\mathbf{x}) + \partial g(\mathbf{x}) \cdot$$

On the other hand, by the Lipschitz continuity of ∇f_σ we have

$$\begin{aligned}f_\sigma(\mathbf{x}_{k+1}) \leq f_\sigma(\mathbf{x}_k) + \nabla f_\sigma(\mathbf{x}_k)^T(\mathbf{x}_{k+1} - \mathbf{x}_k) \\ + \frac{L}{2}\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2^2\end{aligned} \tag{33}$$

Adding (31) and (33) results in

$$\begin{aligned}f_\sigma(\mathbf{x}_{k+1}) + g(\mathbf{x}_{k+1}) \leq \\ f_\sigma(\mathbf{x}_k) + g(\mathbf{x}_k) - (\frac{1}{2\mu_\sigma} - \frac{L}{2})\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2^2\end{aligned} \tag{34}$$

which implies that the sequence $\{F_\sigma(\mathbf{x}_k)\}_{k=0}^{\infty}$ is decreasing if $\mu_\sigma \in (0, \frac{1}{L}]$. Since F_σ is bounded from below, we conclude that $\{F_\sigma(\mathbf{x}_k)\}_{k=0}^{\infty}$ converges. Summing (34) for $k = 0, \cdots, \infty$ results in

$$\sum_{k=0}^{\infty} \left\{ (\frac{1}{2\mu_\sigma} - \frac{L}{2})\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2^2 \right\} \leq F_\sigma(\mathbf{x}_0) - F_\sigma(\mathbf{x}_\infty) \tag{35}$$

which together with the fact that the right-hand side is non-negative and finite, results in $\mathbf{x}_{k+1} \to \mathbf{x}_k$. Moreover, the whole sequence $\{\mathbf{x}_k\}_{k=0}^{\infty}$ is contained in the level set $\{\mathbf{x} | \; F_\sigma(\mathbf{x}_\infty) \leq F_\sigma(\mathbf{x}) \leq F_\sigma(\mathbf{x}_0)\}$, which is bounded. Using this, the boundedness of the sequence $\{\mathbf{x}_k\}_{k=0}^{\infty}$ is readily concluded. So, according to the Bolzano–Weierstrass theorem [31], there exists a convergent subsequence $\{\mathbf{x}_{k_j}\}_{j=0}^{\infty}$ that converges to an accumulation point, say $\mathbf{x}^*$. We next prove that $\mathbf{x}^*$ is a stationary point of F_σ. Let us define

$$\mathbf{u}_j \triangleq \nabla f_\sigma(\mathbf{x}_{k_j}) - \nabla f_\sigma(\mathbf{x}_{k_j - 1}) - \frac{1}{\mu_\sigma}(\mathbf{x}_{k_j} - \mathbf{x}_{k_j - 1}) \cdot \tag{36}$$

Then, from (32) it follows that $\mathbf{u}_j \in \partial F_\sigma(\mathbf{x}_{k_j})$. Now, using the Lipschitz continuity of ∇f_σ we have

$$\|\mathbf{u}_j\|_2 \leq L\|\mathbf{x}_{k_j} - \mathbf{x}_{k_j - 1}\|_2 + \frac{1}{\mu_\sigma}\|\mathbf{x}_{k_j} - \mathbf{x}_{k_j - 1}\|_2 \to 0 \tag{37}$$

where we have used the fact that $\mathbf{x}_{k+1} - \mathbf{x}_k \to \mathbf{0}$. So, $\mathbf{u}_j \to \mathbf{0}$. On the other hand, due to the continuity of f_σ and the lower semicontinuity of g, we have $F_\sigma(\mathbf{x}_{k_j}) \to F_\sigma(\mathbf{x}^*)$. Finally, using Proposition 1

$$\mathbf{0} \in \partial F_\sigma(\mathbf{x}^*),$$

which concludes the proof.

References

[1] M. Elad, *Sparse and Redundant Representations*, Springer, 2010.

[2] H. Hastie and J. Tibshirani, R. Friedman, *The Elements of Statistical Learning*, Springer Series in Statistics. Springer, 2009.

[3] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Proc. Magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[4] P. Comon and C. Jutten, Eds., *Handbook of Blind Source Separation*, Elsevier, 2010.

[5] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly, "Compressed sensing MRI," *IEEE Signal Proc. Magazine*, vol. 25, no. 2, pp. 72–82, 2008.

[6] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang, and S. Yan, "Sparse representation for computer vision and pattern recognition," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1031–1044, 2010.

[7] G. Davis, S. Mallat, and M. Avellaneda, "Adaptive greedy approximations," *Constructive Approximation*, vol. 13, no. 1, pp. 57–98, 1997.

[8] D. L. Donoho, M. Elad, and V. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Trans. Info. Theory*, vol. 52, no. 1, pp. 6–18, 2006.

[9] M. Babaie-Zadeh and C. Jutten, "On the stable recovery of the sparsest overcomplete representations in presence of noise," *IEEE Transactions on Signal Processing*, vol. 58, no. 10, pp. 5396–5400, 2010.

[10] J. A. Tropp and S. J. Wright, "Computational methods for sparse solution of linear inverse problems," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 948–958, 2010.

[11] I. Daubechies, M. Defrise, and C. De-Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Comm. Pure Appl. Math.*, vol. 57, no. 11, pp. 1413–1457, 2004.

[12] M. Elad, "Why simple shrinkage is still relevant for redundant representations?," *IEEE Trans. on Information Theory,*, vol. 52, no. 12, pp. 5559–5569, 2006.

[13] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci.*, vol. 2, no. 1, pp. 183–202, 2009.

[14] J. M. Bioucas-Dias and M. A. T. Figueiredo, "A new twist: Two-step iterative shrinkage/thresholding algorithms for image restoration," *IEEE Transactions on Image Processing*, vol. 16, no. 12, pp. 2992–3004, 2007.

[15] D. L. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proc. Nat. Acad. Sci.*, vol. 106, no. 45, pp. 18 914–18 919, 2009.

[16] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," in *Proc. IEEE Int. Symp. Inform. Thy.*, 2011, pp. 2168–2172.

[17] J. P. Vila and P. Schniter, "Expectation-maximization gaussian-mixture approximate message passing," *IEEE Trans. on Signal Proc.*, vol. 61, no. 19, pp. 4658–4672, 2013.

[18] M. A.T. Figueiredo, R. D. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 586–597, 2007.

[19] M. Friedlander and E. Van den Berg, "Probing the pareto frontier for basis pursuit solutions," *SIAM J. Sci. Comput.*, vol. 31, no. 2, pp. 890–912, 2008.

[20] J. Becker, S. Bobin and E. J. Candès, "NESTA: A fast and accurate first-order method for sparse recovery," *SIAM J. Imaging Sci.*, vol. 4, no. 1, pp. 1–39, 2011.

[21] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition," in *In Proc. Asilomar Conf. Signal Syst. Comput.*, 1993.

[22] J. Wang, S. Kwon, and B. Shim, "Generalized orthogonal matching pursuit," *IEEE Trans. on Signal Proc.*, vol. 60, no. 12, pp. 6202–6216, 2012.

[23] D. Needell and J. A. Tropp, "Cosamp: Iterative signal recovery from in-complete and inaccurate samples," *Appl. Comput. Harmon. Anal.*, vol. 26, no. 3, pp. 301–321, 2009.

[24] L. Zheng, A. Maleki, X. Wang, and T. Long, "Does ℓ_p-minimization outperform ℓ_1-minimization?," 2015, http://arxiv.org/abs/1501.03704.

[25] R. Chartrand and W. Yin, "Iteratively reweighted algorithms for compressive sensing," in *IEEE ICASSP*, 2008.

[26] T. Blumensath and M. E. Davies, "Iterative hard thresholding for compressed sensing," *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265–274, 2009.

[27] H. Mohimani, M. Babaie-Zadeh, and Ch. Jutten, "A fast approach for overcomplete sparse decomposition based on smoothed ℓ^0 norm," *IEEE Trans. on Signal Processing*, vol. 57, pp. 289–301, 2009.

[28] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 123–231, 2014.

[29] A. Blake and A. Zisserman, *Visual Reconstruction*, MIT Press, Cambridge, 1987.

[30] A. Eftekhari, M. Babaie-Zadeh, C. Jutten, and H. Abrishami-Moghaddam, "Robust-SL0 for stable sparse representation in noisy settings," in *Proceedings of ICASSP2009*, 2009, pp. 3433–3436.

[31] H. H. Sohrab, *Basic Real Analysis*, Birkhäuser Basel, 2014.

[32] P. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pp. 185–212, 2011.

[33] H. Attouch, J. Bolte, and B. F. Svaiter, "Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized gaussseidel methods," *Mathematical Programming*, vol. 137, no. 1, pp. 91–129, 2013.

[34] E. T. Hale, W. Yin, and Y. Zhang, "Fixed-point continuation for l1-minimization: Methodology and convergence," *SIAM Journal on Optimization*, vol. 19, no. 3, pp. 1107–1130, 2008.

[35] "SL0 website," http://ee.sharif.edu/∼SLzero/.

[36] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[37] J. Bolte, S. Sabach, and M. Teboulle, "Proximal alternating linearized minimization for nonconvex and nonsmooth problems," *Mathematical Programming*, vol. 146, no. 1-2, pp. 459–494, 2014.

[38] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2210–2239, 1998.

[39] W. Chen, M. R. D. Rodrigues, and I. J. Wassell, "Projection design for statistical compressive sensing: {A tight frame based approach," *IEEE Trans. on Signal Proc.*, vol. 61, no. 8, pp. 2016–2029, 2013.

[40] W. W. Hager, "Updating the inverse of a matrix," *SIAM Review*, vol. 31, no. 2, pp. 221–239, 1989.

[41] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[42] N. Hurley and S. Rickard, "Comparing measures of sparsity," *IEEE Transactions on Information Theory*, vol. 55, no. 10, pp. 4723–4741, 2009.

[43] J. Vila, P. Schniter, S. Rangan, F. Krzakala, and L. Zdeborova, "Adaptive damping and mean removal for the generalized approximate message passing algorithm," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 2021–2025.

[44] V. Cevher, "Learning with compressible priors," in *Advances in Neural Information Processing Systems 22 (NIPS 2009)*, 2009.

[45] P. Ochs, Y. Chen, T. Brox, and T. Pock, "iPiano: Inertial proximal algorithm for nonconvex optimization," *SIAM J. Imag. Sci.*, vol. 7, no. 2, pp. 388–1419, 2014.

[46] R. Tyrrell Rockafellar and R. J-B Wets, *Variational Analysis*, Springer, 1998.

[47] B. Mordukhovich, *Variational Analysis and Generalized Differentiation I. Basic Theory*, vol. 330 of *Grundlehren der mathematischen Wissenschaften*, Springer-Verlag Berlin Heidelberg, 1st edition, 2006.