Smoothed Sparse Recovery via Locally Competitive Algorithm and Forward Euler Discretization Method

Qi Liu[a,*], Yuantao Gu[b], Hing Cheung So[a,1]

[a] *Department of Electronic Engineering, City University of Hong Kong, Hong Kong, China*
[b] *Department of Electronic Engineering and Tsinghua National Laboratory for Information Science and Technology (TNList), Tsinghua University, Beijing, China*

Abstract

This paper considers the problem of sparse recovery whose optimization cost function is a linear combination of a nonsmooth sparsity-inducing term and an ℓ_2-norm as the metric for the residual error. Since the resultant sparse approximation involves nondifferentiable functions, locally competitive algorithm and forward Euler discretization method are exploited to approximate the nonsmooth objective function, yielding a smooth optimization problem. Alternating direction method of multipliers is then applied as the solver, and Nesterov acceleration trick is integrated for speeding up the computation process. Numerical simulations demonstrate the superiority of the proposed method over several popular sparse recovery schemes in terms of computational complexity and support recovery.

Keywords: Locally competitive algorithm (LCA); Alternating direction method of multipliers (ADMM); Smoothed sparse recovery

1. Introduction

Sparse recovery aims to estimate an unknown sparse signal from a noisy underdetermined linear system, which is of great interest in signal processing and machine learning [1]. Mathematically, given a *sensing matrix* $\mathbf{A} \in \mathbb{R}^{M\times N}$ with $M < N$, the underlying task is to recover a *target signal* $\mathbf{x} \in \mathbb{R}^N$ from its undersampled set of noisy observations $\mathbf{y} \in \mathbb{R}^M$:

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \boldsymbol{\epsilon} \tag{1}$$

where $\boldsymbol{\epsilon} = [\epsilon_1\ \epsilon_2\ \ldots\ \epsilon_M]^T$ is the additive disturbance vector and $\{\epsilon_i\}$ are independent and identically distributed (i.i.d.) random variables with variance σ^2. One of the most popular approaches to sparse signal recovery is to minimize a linear combination of a sparsity-inducing term and an ℓ_2-norm as the metric for the residual error [2–4]:

$$\min_{\mathbf{x}} \lambda \|\mathbf{x}\|_p + \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2, \quad p = 0 \text{ or } 1 \tag{2}$$

where $\lambda > 0$ is a tradeoff parameter that balances the weight between the sparsity-inducing term and ℓ_2-norm error term, and can be chosen as suggested in [5, 6]. When $p = 1$, the cost function $\|\mathbf{x}\|_p$ is $\|\mathbf{x}\|_1 = \sum_i |x_i|$,

*Corresponding author
Email addresses: lq_skyven@126.com (Qi Liu), gyt@tsinghua.edu.cn (Yuantao Gu), hcso@ee.cityu.edu.hk (Hing Cheung So)
[1] EURASIP Member

resulting in an ℓ_1-norm penalty, which is used as a convex surrogate for the ideal counting norm (i.e., $\|\mathbf{x}\|_0$). The resulting ℓ_1-norm minimization is also known as the least absolute shrinkage and selection operator (LASSO) [7]. Numerical algorithms (some of them are extremely fast) have been proposed for the ℓ_1-norm minimization based sparse recovery [6, 8–10]. A specialized interior-point method has been designed for solving the ℓ_1-regularized least squares problem (ℓ_1-LS) [6], but it is time-consuming. In fact, the ℓ_1-norm is the closest convex norm to the ℓ_0-norm, nevertheless, it is known that the ℓ_1-norm underestimates the number of nonzero elements, when used as a sparsity-inducing regularizer.

Although the problem of ℓ_0-norm minimization is NP-hard and because the corresponding function is highly discontinuous and nondifferentiable, it gives the highest sparse recovery performance with very few measurements [11]. Hence, this motivates the use of approximate ℓ_0 function in solving (1). Several works have been proposed to relax the ℓ_0-norm minimization, such as focal underdetermined system solver (FOCUSS) [12], approximate projected generalized gradient (APGG) [3], iteratively reweighted least squares (IRLS) [13], iterative reweighted ℓ_1 minimization (IRℓ_1) [14], smoothed ℓ_0 (SL0) [5] and RSL0 [15]. Application of the iteratively reweighted algorithm for solving ℓ_p minimization with $p < 1$ is studied in [13], where sparse signals can be recovered even in the presence of noise from fewer measurements. SL0 [5] utilizes the steepest accent algorithm to maximize its cost function, which achieves fast implementation and improved performance in noisy environment. To achieve robustness against noise, RSL0 [15] is proposed to modify the projection step of the noise-free SL0 [5].

In this work, an efficient algorithm is devised for sparse recovery with nonsmooth sparsity-inducing term. We briefly summarize the contributions of this work as follows: i) Since the sparsity-inducing function is not differentiable, motivated by the concept of locally competitive algorithm (LCA) [16], we combine LCA and alternating direction method of multipliers (ADMM) [17] to devise a fast and accurate algorithm to solve the general nonsmooth sparse approximation problems. Also, Nesterov acceleration trick [18] is utilized to speed up the computing process. ii) A challenge is that LCA is a continuous-time algorithm, and forward Euler discretization method [19, 20] is applied to approximate the penalty function into the smoothed version; iii) For orthonormal and general sensing matrices, matrix inversion lemma and iterative method are introduced for their computationally efficient calculation, respectively.

The reminder of this paper is organized as follows. In Section 2, the background of LCA and its corresponding threshold function are described. Section 3 introduces the efficient algorithm for sparse recovery. In Section 4, numerical examples are conducted to evaluate the performance of the proposed algorithm. Section 5 concludes this paper.

Notation: $\mathrm{E}(\cdot)$, $\langle\cdot,\cdot\rangle$, $(\cdot)^T$ and $(\cdot)^{-1}$ stand for the expectation, inner product, transpose and inverse operators, respectively. $\partial J(\cdot)$ represents the subdifferential of the function J. $\mathbf{I}_M$ stands for the $M \times M$ identity matrix. $\mathbf{0}$ denotes a vector with all zero entries .

2. Preliminary

LCA is a continuous-time algorithm designed to solve the problem of sparse approximation. To begin with, we define $\mathcal{J}(\mathbf{x})$:

$$\mathcal{J}(\mathbf{x}) \coloneqq \lambda \|\mathbf{x}\|_1 + \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2. \tag{3}$$

The subdifferential of (3) with respect to $\mathbf{x}$ in the set valued framework is

$$\partial_{\mathbf{x}}\mathcal{J}(\mathbf{x}) = \lambda \mathrm{sgn}(\mathbf{x}) + \mathbf{A}^T(\mathbf{A}\mathbf{x}-\mathbf{y}) \tag{4}$$

where

$$\mathrm{sgn}(x) = \begin{cases} 1, & \text{if } x > 0 \\ \in [-1\ 1], & \text{if } x = 0 \\ -1, & \text{if } x < 0. \end{cases}$$

In this unconstrained convex optimization formulation (3), the LCA introduces an internal state vector $\mathbf{u} = [u_1\ u_2\ \ldots\ u_N]^T$ for $\mathbf{x}$:

$$\mathbf{x} \coloneqq \mathrm{Prox}_{(\eta,\delta,\lambda)}(\mathbf{u}). \tag{5}$$

where $\mathrm{Prox}_{(\eta,\delta,\lambda)}(\cdot)$ denotes the proximal mapping of ℓ_1-norm when setting $\eta \to \infty, \delta = 1$ and $\lambda = 1$. Then $\mathbf{x}$ and $\mathbf{u}$ should have the same sign. By simple calculation,

$$u - x = (|u| - \max(|u| - \lambda, 0) \cdot \mathrm{sgn}(u)) = \lambda \mathrm{sgn}(x). \tag{6}$$

Substituting (6) into (4), we obtain

$$\tau \frac{d\mathbf{u}}{dt} = -\partial_{\mathbf{x}}\mathcal{J}(\mathbf{x}) = -\mathbf{u} + \mathbf{x} - \mathbf{A}^T(\mathbf{A}\mathbf{x}-\mathbf{y}). \tag{7}$$

Here, τ represents the time constant of the physical solver in implementing the algorithm. In [16, 21, 22], the relationship among $\partial|x_i|$, x_i and u_i is established. Given $\mathbf{x}$, we have

$$\mathbf{u} - \mathbf{x} = \lambda \partial \|\mathbf{x}\|_1. \tag{8}$$

Since $\|\mathbf{x}\|_1$ is nondifferentiable at $\mathbf{x} = \mathbf{0}$, the subdifferential, denoted as $\partial\|\mathbf{x}\|_1$, is employed to describe the gradient of $\|\mathbf{x}\|_1$. The advantage of LCA is that we do not need to implement $\partial\|\mathbf{x}\|_1$ directly.

A general threshold function for mapping from $\mathbf{x}$ to $\mathbf{u}$, is given by [16]

$$x_i = \mathrm{Prox}_{(\eta,\delta,\lambda)}(u_i) = \mathrm{sign}(u_i)\frac{|u_i| - \delta\lambda}{1 + \mathrm{e}^{-\eta(|u_i|-\lambda)}} \tag{9}$$

where η is a parameter to control the speed of the threshold transition, $\delta \in [0, 1]$ indicates that the fraction of an additive adjustment is made for values above threshold, and $\mathrm{sign}(\cdot)$ denotes the sign of a quantity with $\mathrm{sign}(0) = 0$. Some examples of this general threshold function are provided in Fig. 1. The LCA embeds the tradeoff parameter into the threshold function $\mathrm{Prox}_{(\eta,\delta,\lambda)}(\cdot)$. Setting $\eta \to \infty, \delta = 0$ and $\lambda = 1$, we obtain an ideal hard threshold function, which approximates the ℓ_0-norm based sparse recovery. Setting $\eta \to \infty, \delta = 1$ and $\lambda = 1$, then the general threshold function is reduced to the soft threshold function [16], corresponding to the case of $p = 1$.

3. Algorithm Development

Now, we consider the nonsmooth ℓ_1/ℓ_0-norm approximation problem of (2). To solve (2), let $J(\mathbf{x}) \coloneqq \|\mathbf{x}\|_p$ ($p = 0$ or 1) for simplicity, we introduce an auxiliary variable $\mathbf{z} \in \mathbb{R}^M$ to rewrite it equivalently as

$$\min_{\mathbf{x},\mathbf{z}} \lambda J(\mathbf{x}) + \|\mathbf{z}\|_2, \quad \text{s.t. } \mathbf{A}\mathbf{x} - \mathbf{z} = \mathbf{y}. \tag{10}$$

ADMM is a powerful optimization framework that is suitable for large-scale problems arising in machine learning and signal processing. In the case of the ℓ_0-norm minimization, directly extending the ADMM algorithm is not guaranteed to converge since the regularization term is nonconvex. To derive a convergent algorithm for the nonconvex case, we develop a proximal ADMM algorithm via incorporating with the LCA.

Based on the decomposition-coordination procedure of the ADMM, we proceed to unveil each step for solving problem (10). The corresponding augmented Lagrangian of (10) is

$$\begin{aligned} L(\mathbf{z},\mathbf{x},\mathbf{w}) =& \lambda J(\mathbf{x}) + \|\mathbf{z}\|_2 - < \mathbf{w}, \mathbf{A}\mathbf{x} - \mathbf{z} - \mathbf{y} > \\ &+ \frac{\rho}{2}\|\mathbf{A}\mathbf{x} - \mathbf{z} - \mathbf{y}\|_2^2 \end{aligned} \tag{11}$$

where $\mathbf{w} \in \mathbb{R}^M$ is the dual variable vector, $\rho > 0$ is a penalty parameter that controls the convergence rate of the algorithm. Then ADMM applied to (11) consists of the following iterative steps.

In the $(t+1)$th iteration, the $\mathbf{z}$-minimization step has the closed-form solution, which degenerates to the proximal operator of ℓ_2-norm:

$$\begin{aligned} \mathbf{z}^{t+1} &= \arg\min_{\mathbf{z}} \left(\|\mathbf{z}\|_2 + \frac{\rho}{2}\left\|\mathbf{A}\mathbf{x}^t - \mathbf{z} - \mathbf{y} - \frac{\mathbf{w}^t}{\rho}\right\|_2^2 \right) \\ &= \operatorname{Prox}_{\frac{1}{\rho}\|\cdot\|_2}\left(\mathbf{A}\mathbf{x}^t - \mathbf{y} - \frac{\mathbf{w}^t}{\rho}\right) \end{aligned} \tag{12}$$

in which the proximal operator of ℓ_2-norm is known as [19]

$$\operatorname{Prox}_{\nu\|\cdot\|_2}(\mathbf{v}) = \mathbf{1}_{\|\mathbf{v}\|_2 \geq \nu}(\mathbf{v})\left(1 - \frac{\nu}{\|\mathbf{v}\|_2}\right)\mathbf{v} \tag{13}$$

where $\mathbf{1}(\cdot)$ is the indicator function and $\nu \coloneqq 1/\rho$.

The $\mathbf{x}$-minimization step computes

$$\mathbf{x}^{t+1} = \arg\min_{\mathbf{x}} \left(J(\mathbf{x}) + \frac{\rho}{2\lambda}\left\|\mathbf{A}\mathbf{x} - \mathbf{z}^{t+1} - \mathbf{y} - \frac{\mathbf{w}^t}{\rho}\right\|_2^2 \right). \tag{14}$$

To gain overall efficiency of the algorithm, we use the standard trick for ADMM to solve (14) approximately. Based on the *descent lemma*, we linearize the term $J(\mathbf{x})$ at a given point $\mathbf{x}^t$ as the form of

$$J(\mathbf{x}) \approx J(\mathbf{x}^t) + < \mathbf{x} - \mathbf{x}^t, \partial_{\mathbf{x}} J(\mathbf{x}^t) > + \frac{1}{2}\|\mathbf{x} - \mathbf{x}^t\|_2^2. \tag{15}$$

With this linearization, $\mathbf{x}$-minimization results in the following closed-form solution:

$$\mathbf{x}^{t+1} = (\mathbf{I}_N + \rho\mathbf{A}^T\mathbf{A})^{-1}\left[\mathbf{x}^t - \partial_{\mathbf{x}} J(\mathbf{x}^t) + \frac{\rho}{\lambda}\mathbf{A}^T\left(\mathbf{y} + \mathbf{z}^{t+1} + \frac{\mathbf{w}^t}{\rho}\right)\right]. \tag{16}$$

However, since the penalty function in (14) is not smooth, we cannot obtain $\partial_{\mathbf{x}} J(\mathbf{x}^t)$. LCA can be applied to solve the nonsmooth sparse recovery problems, but it is a continuous-time algorithm. Assigning

a step size for the discretization equal to the LCA time-constant τ, and then applying the forward Euler discretization method to the LCA, we proximate the penalty function in (14) into the smoothed version:

$$\begin{aligned} \tau\frac{\mathbf{u}^{t+1}-\mathbf{u}^t}{\tau} &= \tau\frac{d\mathbf{u}}{dt} = -\partial_{\mathbf{u}}J(\mathbf{u}^t) \\ \Rightarrow \mathbf{u}^{t+1} &= \mathbf{u}^t - \partial_{\mathbf{u}}J(\mathbf{u}^t) = \mathbf{x}^t - \partial_{\mathbf{x}}J(\mathbf{x}^t) \\ \mathbf{x}^{t+1} &= \mathrm{Prox}_{(\eta,\delta,\lambda)}(\mathbf{u}^{t+1}). \end{aligned} \tag{17}$$

It is easy to see that the first-order discretization of the LCA dynamics is expressed as

$$\mathbf{x}^{t+1} = \mathrm{Prox}_{(\eta,\delta,\lambda)}(\mathbf{x}^t - \rho\partial_{\mathbf{x}}J(\mathbf{x}^t)). \tag{18}$$

According to the definition of the proximity operator [17, 19], we have:

$$\mathrm{Prox}_{(\eta,\delta,\lambda)}(\mathbf{x}^t) \approx \mathbf{x}^t - \lambda\partial_{\mathbf{x}}J(\mathbf{x}^t). \tag{19}$$

Therefore, $\mathbf{x}$-minimization in (16) corresponds to

$$\mathbf{x}^{t+1} = (\mathbf{I}_N + \rho\mathbf{A}^T\mathbf{A})^{-1}\left[\mathrm{Prox}_{(\eta,\delta,\lambda)}(\mathbf{x}^t) + \frac{\rho}{\lambda}\mathbf{A}^T\left(\mathbf{y} + \mathbf{z}^{t+1} + \frac{\mathbf{w}^t}{\rho}\right)\right]. \tag{20}$$

Here, the term $(\mathbf{I}_N + \rho\mathbf{A}^T\mathbf{A})^{-1}$ requires the inversion of a $N \times N$ matrix, which is computationally expensive. However, in the special case where the sensing matrix $\mathbf{A}$ is orthonormal, i.e., $\mathbf{A}\mathbf{A}^T = \kappa\mathbf{I}_M$, the complexity of matrix inverse operator can be reduced via the *matrix inversion lemma*:

$$\begin{aligned} (\mathbf{I}_N + \rho\mathbf{A}^T\mathbf{A})^{-1} &= \mathbf{I}_N - \rho\mathbf{A}^T(\mathbf{I}_M + \rho\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A} \\ &= \mathbf{I}_N - \frac{\rho}{1+\kappa\rho}\mathbf{A}^T\mathbf{A}. \end{aligned} \tag{21}$$

Then, based on (21), (20) is rewritten as

$$\mathbf{x}^{t+1} = \left(\mathbf{I}_N - \frac{\rho}{1+\kappa\rho}\mathbf{A}^T\mathbf{A}\right)\cdot\left[\mathrm{Prox}_{(\eta,\delta,\lambda)}(\mathbf{x}^t) + \frac{\rho}{\lambda}\mathbf{A}^T\left(\mathbf{y} + \mathbf{z}^{t+1} + \frac{\mathbf{w}^t}{\rho}\right)\right]. \tag{22}$$

When the rows of $\mathbf{A}$ are not orthonormal, the term $(\mathbf{I}_N + \rho\mathbf{A}^T\mathbf{A})^{-1}$ does not result in a computationally efficient calculation, which can be tackled by a designed matrix $\mathbf{B}$. A well-known iterative method to compute $\mathbf{B}$ is [23]:

$$\mathbf{B}_0 = \xi(\mathbf{I}_N + \rho\mathbf{A}^T\mathbf{A})^T, \quad \mathbf{B}_k = \mathbf{B}_{k-1}(2\mathbf{I}_N - (\mathbf{I}_N + \rho\mathbf{A}^T\mathbf{A})\mathbf{B}_{k-1}), \text{ and } \mathbf{B} = \mathbf{B}_k \text{ for some } k,$$

where $0 < \xi < 2/\left\|(\mathbf{I}_N + \rho\mathbf{A}^T\mathbf{A})(\mathbf{I}_N + \rho\mathbf{A}^T\mathbf{A})^T\right\|_1$. Let $\bar{\mathbf{A}} := (\mathbf{I}_N + \rho\mathbf{A}^T\mathbf{A})$. Since

$$\begin{aligned} &\left\|\mathbf{I}_N - \bar{\mathbf{A}}\mathbf{B}_0\right\|_2 = \left\|\mathbf{I}_N - \xi\bar{\mathbf{A}}\bar{\mathbf{A}}^T\right\|_2 < 1 \\ &\left\|\mathbf{I}_N - \bar{\mathbf{A}}\mathbf{B}_k\right\|_2 \le \left\|\mathbf{I}_N - \bar{\mathbf{A}}\mathbf{B}_{k-1}\right\|_2^2 \le \left\|\mathbf{I}_N - \bar{\mathbf{A}}\mathbf{B}_0\right\|_2^{2^k}, \end{aligned} \tag{23}$$

the iterative method for computing $\mathbf{B}$ converges quadratically. By using a proper $\mathbf{B}$, a nice approximate analytic solution of (20) is easily obtained without evaluating the matrix inverse. Hence, (20) is computed as

$$\mathbf{x}^{t+1} = \mathbf{B}\cdot\left[\mathrm{Prox}_{(\eta,\delta,\lambda)}(\mathbf{x}^t) + \frac{\rho}{\lambda}\mathbf{A}^T\left(\mathbf{y} + \mathbf{z}^{t+1} + \frac{\mathbf{w}^t}{\rho}\right)\right]. \tag{24}$$

The update of $\mathbf{w}$ is

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \rho(\mathbf{A}\mathbf{x}^{t+1} - \mathbf{z}^{t+1} - \mathbf{y}). \tag{25}$$

To accelerate the convergence of a proximal gradient method, the Nesterov acceleration trick has been used in FISTA [8], in which the convergence rate is accelerated from $O(1/t)$ to $O(1/t^2)$. Such technique is also applicable to our method with the following computing process:

$$\begin{aligned} t^{t+1} &= \frac{1+\sqrt{1+4(t^t)^2}}{2} \\ \mathbf{x}^{t+1} &= \mathbf{x}^t + \frac{t^t-1}{t^{t+1}}(\mathbf{x}^t - \mathbf{x}^{t-1}) \\ \mathbf{w}^{t+1} &= \mathbf{w}^t + \frac{t^t-1}{t^{t+1}}(\mathbf{w}^t - \mathbf{w}^{t-1}). \end{aligned} \tag{26}$$

This completes one update cycle and the algorithm terminates once the difference between two consecutive iterations is smaller than a given threshold or if the maximum iteration number is reached. While the convergence analysis under such acceleration is not in the scope of this work, we have the algorithm summarized in Algorithm 1, where the initial vector of $\mathbf{x}^0$ is taken as the least squares (LS) solution, i.e., $\mathbf{x}^0 = \mathbf{A}^\dagger\mathbf{y}$.

Theorem : For any fixed $\rho > 0$, the sequence $\{(\mathbf{z}^t, \mathbf{x}^t, \mathbf{w}^t)\}$ generated by (12), (24) and (25) from any starting point $(\mathbf{x}^0, \mathbf{w}^0)$ converges to $(\mathbf{z}^*, \mathbf{x}^*, \mathbf{w}^*)$, where $(\mathbf{z}^*, \mathbf{x}^*, \mathbf{w}^*)$ is a solution of (10).

Proof: Since the ℓ_1-norm minimization in (10) is convex, the Karush-Kuhn-Tucker (KKT) conditions are sufficient and necessary, then there exists $\mathbf{w}^*$ such that $\frac{1}{\lambda}\nabla\|\mathbf{z}^*\|_2 = -\mathbf{w}^*$, $\mathbf{A}^T\mathbf{w}^* \in \partial J(\mathbf{x}^*)$ and $\mathbf{A}\mathbf{x}^* - \mathbf{y} = \mathbf{w}^*$, and further results in $(\mathbf{z}^* - \mathbf{z}^{t+1})^T(\mathbf{w}^{t+1} - \mathbf{w}^* - \rho\mathbf{A}(\mathbf{x}^{t+1} - \mathbf{x}^*)) = \frac{1}{\lambda}(\mathbf{z}^{t+1} - \mathbf{z}^*)^T(\nabla\|\mathbf{z}^{t+1}\|_2 - \nabla\|\mathbf{z}^*\|_2) \geq 0$, which is just the equation (A.2) in Theorem 2.1 of [24], and the rest of the proof follows similarly the proof of Theorem 2.1 in [24], which is omitted here for succinctness. For the ℓ_p-norm minimization at $p = 0$ in (10), we set $J(\mathbf{x}) = \|\mathbf{x}\|_p$, $h(\mathbf{x}) = \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2$, $g(\mathbf{x}) = J(\mathbf{x}) + h(\mathbf{x})$ and let $\hat{h}_\delta(\mathbf{x}, \mathbf{x}^t) = h(\mathbf{x}^t) + < \mathbf{x} - \mathbf{x}^t, \nabla h(\mathbf{x}^t) > + \frac{1}{2\delta}\|\mathbf{x} - \mathbf{x}^t\|_2^2$ be a convex upper bound to $h(\mathbf{x})$ that is tight at $\mathbf{x}^t$. It is easy to check that ℓ_p-norm with $0 \leq p < 1$ is non-decreasing in $\mathbf{x} \in [0, \infty)$ [25]. According to (14), we have $J(\mathbf{x}^{t+1}) + \frac{\rho}{2\lambda}\hat{h}_\delta(\mathbf{x}^{t+1}, \mathbf{x}^t) \leq J(\mathbf{x}^t) + \frac{\rho}{2\lambda}\hat{h}_\delta(\mathbf{x}^t, \mathbf{x}^t)$, which is equivalent to $g(\mathbf{x}^{t+1}) - g(\mathbf{x}^t) \leq \frac{\xi}{2}\|\mathbf{A}(\mathbf{x}^{t+1} - \mathbf{x}^t)\|_2^2 - \frac{\xi}{2\delta}\|\mathbf{x}^{t+1} - \mathbf{x}^t\|_2^2$ with $\xi := \frac{\rho}{2\lambda}$. Since $0 < \delta < \frac{1}{L}$ with L is the Lipschitz constant of the gradient of $h(\mathbf{x})$, we can further derive $g(\mathbf{x}^{t+1}) - g(\mathbf{x}^t) \leq \frac{\xi}{2\delta}(\delta L - 1)\|\mathbf{x}^{t+1} - \mathbf{x}^t\|_2^2 \leq 0$. As a result, the sequence $\{g(\mathbf{x}^t)\}$ is non-increasing, lower bounded, and is also convergent. Moreover, since $\{g(\mathbf{x}^t)\}$ is convergent and $\|\mathbf{x}^{t+1} - \mathbf{x}^t\|_2^2 \leq \frac{2\delta}{\xi(1-\delta L)}(g(\mathbf{x}^t) - g(\mathbf{x}^{t+1}))$, $\|\mathbf{x}^{t+1} - \mathbf{x}^t\|_2^2 \to 0$ as $t \to \infty$. Thus, an important consequence in [26] is that the bounded sequences $\{\mathbf{x}^t\}_{t\epsilon N}$ generated by $P_C(\mathbf{x}^t - \rho\partial_{\mathbf{x}}J(\mathbf{x}^t))$ are convergent sequences as long as C is a closed semi-algebraic subset of $\mathbb{R}^N$ and $h(\mathbf{x})$ is C semi-algebraic with L-Lipschitz gradient. Therefore, since $g(\mathbf{x})$ is bounded from below, and together with the fact that the sequences $\{\mathbf{x}^t\}_{t\epsilon N}$ generated by (12), (24) and (25) are bounded. Moreover, $h(\mathbf{x})$ is a polynomial function and $J(\mathbf{x})$ has a piecewise linear graph, hence the sum $g(\mathbf{x}) = J(\mathbf{x}) + h(\mathbf{x})$ is semi-algebraic (or a Kurdyka-Lojasiewicz function) and there exists a constant $C > 0$ such that $\text{dist}(0, \partial g(\mathbf{x}^{t+1})) \leq C\|\mathbf{x}^{t+1} - \mathbf{x}^t\|$. Similarly, following the proof of Theorem 3 in [27] with minor changes, it shows that the sequence $\{(\mathbf{z}^t, \mathbf{x}^t, \mathbf{w}^t)\}$ has finite length. ∎

Algorithm 1 : Efficient algorithm for nonsmooth sparse recovery

Input : $\mathbf{A}, \mathbf{y}$ and $\rho > 0$

Initialize : $\mathbf{z}^0, \mathbf{w}^0$ and $\mathbf{x}^0 = \mathbf{A}^\dagger \mathbf{y}, \mathbf{B}$

for $t = 0, 1, 2 \cdots$ **do**

$\mathbf{z}^{t+1} = \mathrm{Prox}_{\frac{1}{\rho}\|\cdot\|_2}(\mathbf{A}\mathbf{x}^t - \mathbf{y} - \frac{\mathbf{w}^t}{\rho})$

$\mathbf{x}^{t+1} = \mathbf{B} \cdot \left[\mathrm{Prox}_{(\eta,\delta,\lambda)}(\mathbf{x}^t) + \frac{\rho}{\lambda}\mathbf{A}^T\left(\mathbf{y} + \mathbf{z}^{t+1} + \frac{\mathbf{w}^t}{\rho}\right)\right]$

$\mathbf{w}^{t+1} = \mathbf{w}^t - \rho(\mathbf{A}\mathbf{x}^{t+1} - \mathbf{z}^{t+1} - \mathbf{y})$

Speed up computing process:

$t^{t+1} = \frac{1+\sqrt{1+4(t^t)^2}}{2}$

$\mathbf{x}^{t+1} = \mathbf{x}^t + \frac{t^t-1}{t^{t+1}}(\mathbf{x}^t - \mathbf{x}^{t-1})$

$\mathbf{w}^{t+1} = \mathbf{w}^t + \frac{t^t-1}{t^{t+1}}(\mathbf{w}^t - \mathbf{w}^{t-1})$

Stop if termination condition satisfied.

end for

Output : $\mathbf{x}$

4. Numerical Examples

In this section, numerical simulation results are provided to evaluate the recovery performance of the proposed method with ℓ_p-norm, and to compare with ℓ_1-LS [6], IRLS [13] at $p = 0.5$ and RSL0 [15]. To guaranteed the fairness in the performance comparison, we have employed the same experimental and parameter settings of IRLS in [13], where the sensing matrix $\mathbf{A} \in \mathbb{R}^{M\times N}$ is an i.i.d Gaussian random matrix with $N = 256$ and $M = 100$. The nonzero entries of the sparse vector are uniformly located among all possible choices, and their values follow the standard normal distribution. In all simulation examples, the additive noise vector is zero-mean white Gaussian distributed with covariance $\sigma^2\mathbf{I}_M$. The parameters of other algorithms are selected as recommended by respective authors to guarantee the fairness in the comparison. We repeat our experiments 500 times with different random matrices and sparse signals.

In practice, we cannot set $\eta \to \infty$, instead a sufficiently large value is employed. In the proximate ℓ_0-norm minimization, we set $\eta = 10000, \delta = 0$ and $\lambda = 1$, and hence the threshold of the proximate ℓ_0-norm is

$$x_i = T_{(10000,0,1)}(u_i) = \mathrm{sign}(u_i)\frac{|u_i|}{1 + \mathrm{e}^{-10000(|u_i|-\lambda)}}. \tag{27}$$

In the first experiment, we study the normalized relative error, which is defined as $\mathrm{E}(\|\hat{\mathbf{x}} - \mathbf{x}^*\|_2/\|\mathbf{x}^*\|_2)$. We test the normalized relative error performance of different algorithms versus noise variance. The sparsity K is fixed at 20 and the noise standard deviation σ is now varied from 0 to 20, and the result is shown in Fig. 2. It is observed that the proposed method outperforms the other algorithms and achieves the best robustness to noise.

In the second experiment, we examine the normalized relative error versus the number of iterations at $\sigma = 0.5$. Fig. 3 shows that although the proposed method is inferior to the other algorithms for smaller number of iterations, it yields a better recovery performance when the number of iterations is sufficiently large.

In the third experiment, we test the normalized relative error versus the sparsity K, and the result is shown in Fig. 4. It seen that our proposed method outperforms other algorithms when $K \leq 30$, i.e., the strictly sparse signal satisfies $M \geq CK\ln(N/K)$, where C is roughly set at $C < 2$, based on the restricted isometric property.

In the fourth experiment, normalized relative error versus the number of measurements with noise standard deviation $\sigma = 0.5$ is studied, where the sparsity K is fixed at 20, and the result is plotted in Fig. 5. When the number of measurements is larger than 80, the normalized recovery error of the proposed method is much less than those of the other algorithms. When we increase the number of measurements up to 400, the recovery error of the proposed method decreases dramatically. This phenomenon agrees with the well-known property of the sparse approximation. Thus, we test the successful recovery probability versus number of measurements in Fig. 6. If the relative error is less than 10^{-2}, the recovery is regarded as a success. We observe that the proposed method can recover sparse signal with more measurements than RSL0 and IRLS.

In the fifth experiment, comparisons based on index support recovery and runtime are studied. The sensing matrix $\mathbf{A}$ of size 60×100 is randomly generated with entries drawn from the standard Gaussian distribution. The support of $\mathbf{x}$, denoted by $\mathcal{I}$, is randomly selected from the range $[1, 100]$. The number of nonzero elements in $\mathbf{x}$ is 10. The noise standard deviation is set at $\sigma = 0.5$. The results are shown in Figure 7. In the presence of noise, all algorithms correctly recover the support of original signal. In particular, since there are rather few incorrect indices associated with small amplitudes, the proposed method can provide more reliable support estimation than others. On the other hand, to compare the computational complexity of the different recovery algorithms, the average CPU runtime is used as the performance metric, although the runtime gives only a rough estimation of complexity. Our simulations are performed using MATLAB R2015b on a personal computer with 3.40GHz Intel core i7 CPU and 4 GB RAM, under a 64-bit Microsoft Windows 7 operating system. In such implementations, the maximum number of iterations and the stopping criteria of different algorithms are set at the same values, i.e., 200 and 10^{-7}, respectively. It is observed in Table 1 that the proposed method enjoys comparable runtime performance for sparse recovery, except that the RSL0 is the fastest.

5. Conclusion

In this paper, a fast and accurate algorithm has been devised for ℓ_1/ℓ_0-norm minimization to solve general nonsmooth sparse approximation problems. To tackle the nonsmooth formulation, LCA is utilized for its approximation as a smoothed version with the help of forward Euler discretization method and then ADMM is applied as the solver. Moreover, Nesterov acceleration trick is also implemented to speed up the computing process. Numerical results show that the proposed method has the advantages of better recovery accuracy against noise and comparable runtime with several conventional approaches.

Acknowledgments

This work was supported by a grant from the NSFC/RGC Joint Research Scheme sponsored by the National Natural Science Foundation of China and the Research Grants Council of Hong Kong (61531166005,

N_CityU 104/15).

References

[1] E. J. Candès, J. Romberg, T. Tao, Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information, IEEE Trans. Inf. Theory 52 (2) (2006) 489–509.

[2] G. Gasso, A. Rakotomamonjy, S. Canu, Recovering sparse signals with a certain family of nonconvex penalties and DC programming, IEEE Trans. Signal Process. 57 (12) (2009) 4686–4698.

[3] L. Chen, Y. Gu, The convergence guarantees of a non-convex approach for sparse recovery, IEEE Trans. Signal Process. 62 (15) (2014) 3754–3767.

[4] X. Shen, L. Chen, Y. Gu, H. C. So, Square-root lasso with nonconvex regularization: An ADMM approach, IEEE Signal Process. Lett. 23 (7) (2016) 934–938.

[5] H. Mohimani, M. Babaie-Zadeh, C. Jutten, A fast approach for overcomplete sparse decomposition based on smoothed ℓ^0 norm, IEEE Trans. Signal Process. 57 (1) (2009) 289–301.

[6] S. J. Kim, K. Koh, M. Lustig, S. Boyd, D. Gorinevsky, An interior-point method for large-scale ℓ_1-regularized least squares, IEEE J. Sel. Topics Signal Process. 1 (4) (2007) 606–617.

[7] R. Tibshirani, Regression shrinkage and selection via the lasso, J. Roy. Statist. Soc. Ser. B. 58 (1994) 267–288.

[8] A. Beck, M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, SIAM J. Imag. Sci. 2 (1) (2009) 183–202.

[9] W. Yin, S. Osher, D. Goldfarb, J. Darbon, Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing, SIAM J. Imag. Sci. 1 (1) (2008) 143–168.

[10] E. Candès, ℓ_1-MAGIC : Recovery of sparse signals via convex programming, 2005 [Online]. Available: www.acm.caltech.edu/l1magic/downloads/l1magic.pdf.

[11] D. L. Donoho, Compressed sensing, IEEE Trans. Inf. Theory. 52 (4) (2006) 1289–1306.

[12] I. F. Gorodnitsky, B. D. Rao, Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm, IEEE Trans. Signal Process. 45 (3) (1997) 600–616.

[13] R. Chartrand, W. Yin, Iteratively reweighted algorithms for compressive sensing, in: 2008 IEEE Int. Conf. on Acoust., Speech Signal Process., Las Vegas, NV, USA, 2008, pp. 3869–3872.

[14] E. J. Candès, M. B. Wakin, S. P. Boyd, Enhancing sparsity by reweighted $\ell 1$ minimization, J. Fourier Anal. Appl. 14 (5) (2008) 877–905.

[15] A. Eftekhari, M. Babaie-Zadeh, C. Jutten, H. Abrishami-Moghaddam, Robust-SL0 for stable sparse representation in noisy settings, in: in Proc. ICASSP, Taipei, Taiwan, 2009, pp. 3433–3436.

[16] C. J. Rozell, D. H. Johnson, R. G. Baraniuk, B. A. Olshausen, Sparse coding via thresholding and local competition in neural circuits, Neural Comput. 20 (10) (2008) 2526–2563.

[17] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers, Found. Trends Mach. Learn. 3 (1) (2011) 1–122.

[18] Y. Nesterov, A method of solving a convex programming problem with convergence rate $O(1/k^2)$, Soviet Mathematics Doklady 27 (1983) 372–376.

[19] N. Parikh, S. Boyd, Proximal algorithms, Found. Trends Optim. 1 (3) (2014) 127–239.

[20] A. Balavoine, C. J. Rozell, J. Romberg, Discrete and continuous-time soft-thresholding for dynamic signal recovery, IEEE Trans. Signal Process. 63 (12) (2015) 3165–3176.

[21] R. Feng, C. S. Leung, A. G. Constantinides, W.-J. Zeng, Lagrange programming neural network for nondifferentiable optimization problems in sparse approximation, to appear in IEEE Trans. Neural Netw. Learn. Syst.

[22] A. Balavoine, J. Romberg, C. J. Rozell, Convergence and rate analysis of neural networks for sparse approximation, IEEE Trans. Neural Netw. Learn. Syst. 23 (9) (2012) 1377–1389.

[23] A. Ben-Israel, D. Cohen, On iterative computation of generalized inverses and associated projections, SIAM J. Numer. Anal. 3 (3) (1966) 410–419.

[24] J. Yang, Y. Zhang, Alternating direction algorithms for ℓ_1-problems in compressive sensing, SIAM J. Sci. Comput. 33 (1) (2011) 250–278.

[25] L. Chen, Y. Gu, Local and global optimality of lp minimization for sparse recovery, in: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brisbane, QLD, Australia, 2015, pp. 3596–3600.

[26] H. Attouch, J. Bolte, Svaiter, B. Fux, Convergence of descent methods for semi-algebraic and tame problems: Proximal algorithms, forward–backward splitting, and regularized Gauss–Seidel methods, Math. Program 137 (1) (2013) 91–129.

[27] G. Li, T. K. Pong, Global convergence of splitting methods for nonconvex composite optimization, SIAM Journal on Optimization 25 (2015) 2434–2460.

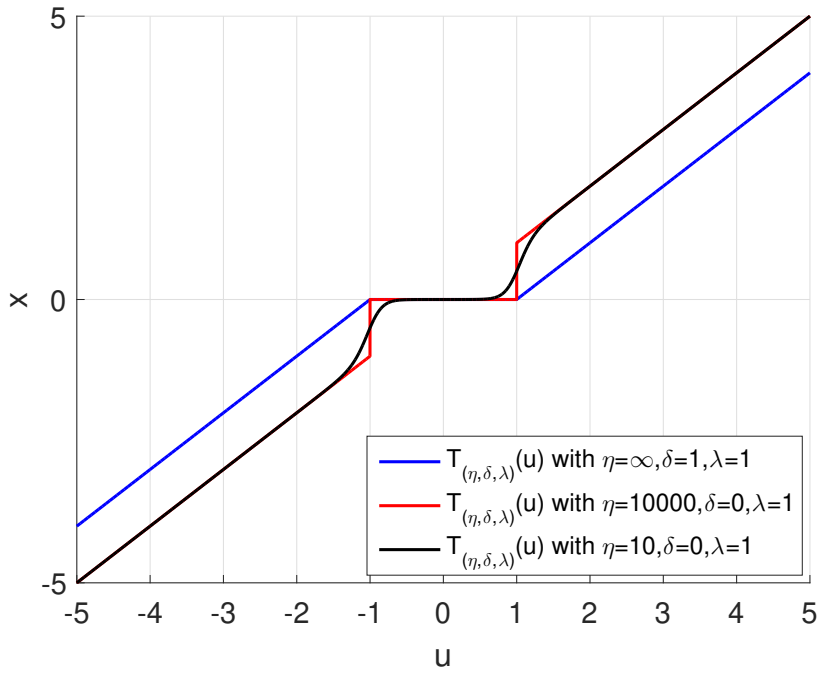

Figure 1: Examples of general threshold function.

Table 1: The average CPU runtime comparison.

Algorithm	Proposed	L1-LS	RSL0	IRLS
Time	0.5058s	1.4988s	0.0062s	0.7448s

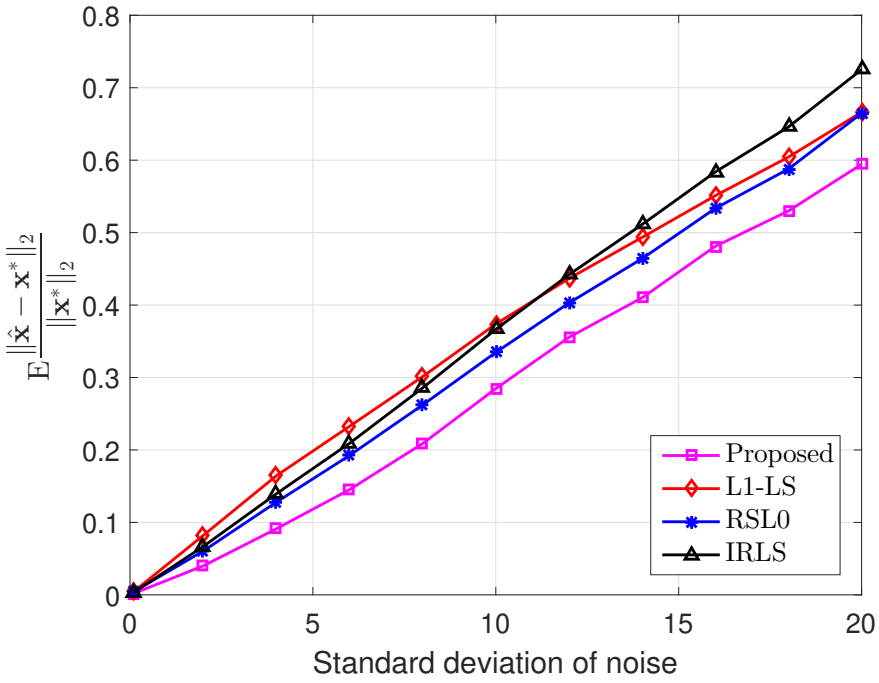

Figure 2: Normalized relative error versus noise standard deviation.

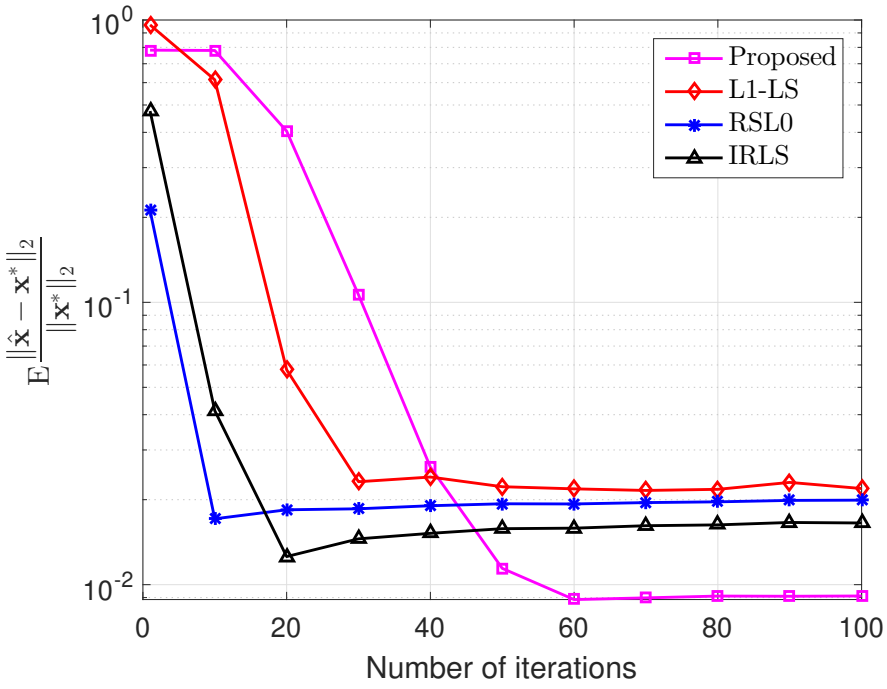

Figure 3: Normalized relative error versus number of iterations.

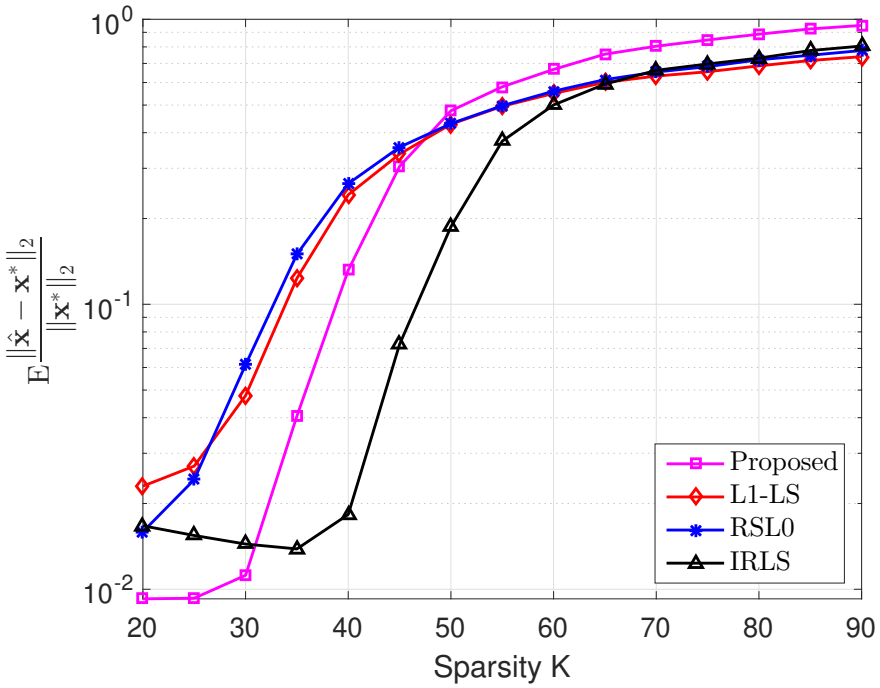

Figure 4: Normalized relative error versus sparsity K.

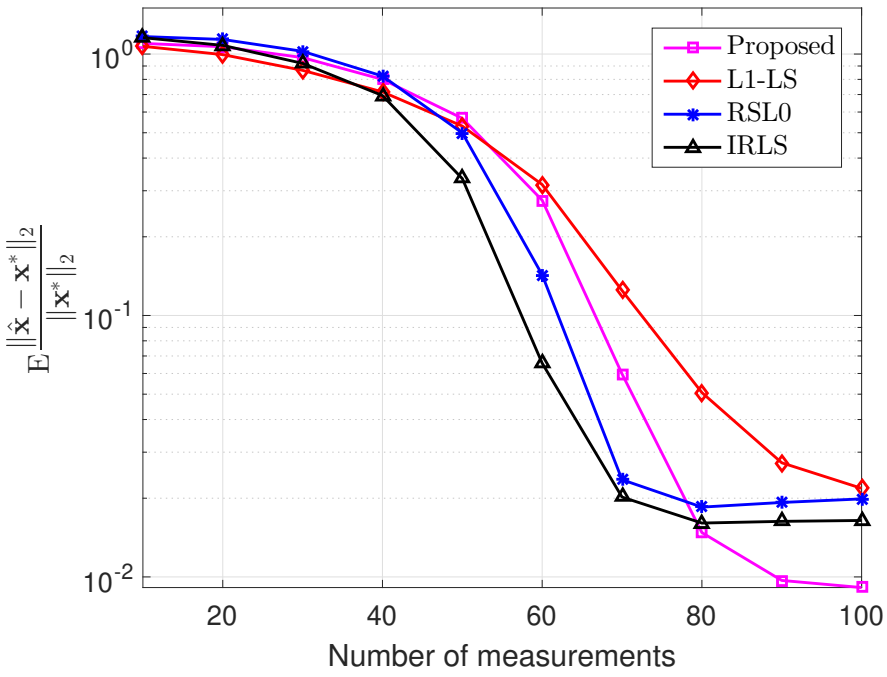

Figure 5: Normalized relative error versus number of measurements.

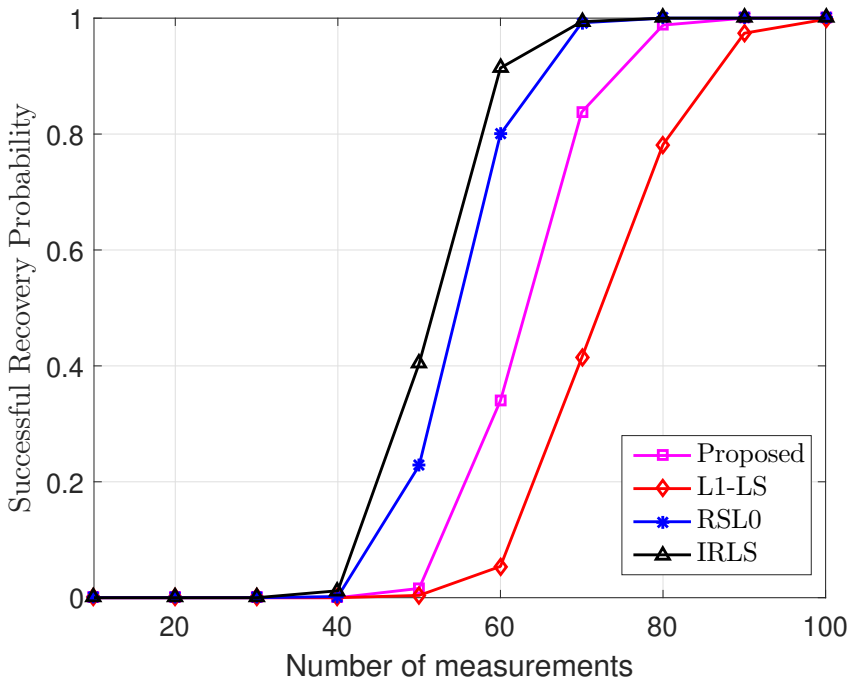

Figure 6: Successful recovery probability versus number of measurements.

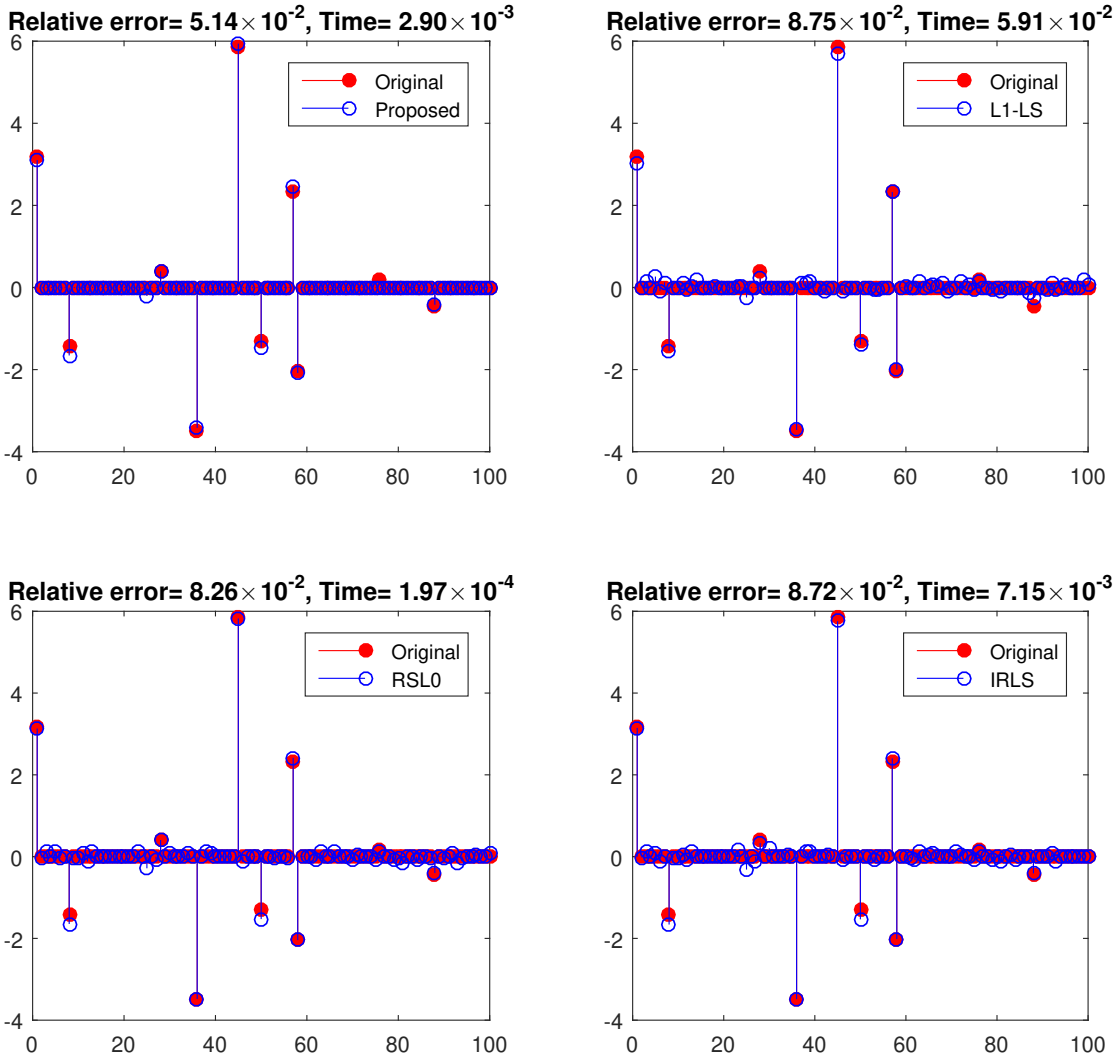

Figure 7: Support recovery and runtime comparison. Blue and red denote the recovered and original signals, respectively.