Signal Processing 108 (2015) 459–475

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Two-level ℓ_1 minimization for compressed sensing

Xiaolin Huang [a,b,*], Yipeng Liu [a,b], Lei Shi [a,b,c], Sabine Van Huffel [a,b], Johan A.K. Suykens [a,b]

[a] KU Leuven, Department of Electrical Engineering (ESAT-STADIUS), B-3001 Leuven, Belgium
[b] KU Leuven, iMinds Future Health Department, B-3001, Lueven, Belgium
[c] School of Mathematical Sciences, Fudan University, Shanghai 200433, PR China

ARTICLE INFO

Article history:
Received 11 March 2014
Received in revised form
9 September 2014
Accepted 22 September 2014
Available online 28 September 2014

Keywords:
Compressed sensing
Non-convex penalty
ℓ_1 minimization
Thresholding algorithm

ABSTRACT

Compressed sensing using ℓ_1 minimization has been widely and successfully applied. To further enhance the sparsity, a non-convex and piecewise linear penalty is proposed. This penalty gives two different weights according to the order of the absolute value and hence is called the two-level ℓ_1-norm. The two-level ℓ_1-norm can be minimized by an iteratively reweighted ℓ_1 method. Compared with some existing non-convex methods, the two-level ℓ_1 minimization has similar sparsity and enjoys good convergence behavior. More importantly, the related soft thresholding algorithm has been established. The shrinkage operator for the two-level ℓ_1-norm is not non-expansive and its convergence is proved by showing the monotone of the objective value in the iterations. In numerical experiments, the proposed algorithms achieve good sparse signal estimation performance, which makes the two-level ℓ_1 minimization a promising technique for compressed sensing.

© 2014 Elsevier B.V. All rights reserved.

1. Introduction

Compressed sensing can acquire and reconstruct a signal from relatively fewer measurements than the classical Nyquist sampling. Random linear projections are used to sub-sample the analogue signal and get the digital measurements. The basics of compressed sensing can be found in [1–3]. The sampling can be formulated in a discrete form as $b=\Phi x$, where $\Phi\in\mathbb{R}^{m\times n}$ is the sampling matrix (also called the measurement matrix), $x\in\mathbb{R}^n$ is the signal, and $b\in\mathbb{R}^m$ represents the compressive measurements. Generally, we have $m<n$. x is then reconstructed from b using recovery methods which exploit the fact that many natural signals are sparse in certain representations, i.e., $x=\Psi\theta$ where Ψ is the representation matrix and θ is a sparse vector.

To recover a sparse signal, the following method has been well studied:

$$\min_{\theta} \quad \|\theta\|_{\ell_0} \quad \text{s.t. } \Phi\Psi\theta=b, \tag{1}$$

where $\|\theta\|_{\ell_0}$ counts the number of nonzero components of θ. We first consider the case $\Psi=\mathbf{I}_{n\times n}$, which stands for the identity matrix in $\mathbb{R}^{n\times n}$. This setting assumes that x is sparse and then (1) becomes

$$\min_{x} \quad \|x\|_{\ell_0}, \quad \text{s.t. } \Phi x=b. \tag{2}$$

In this paper, the theoretical part focuses on (2) and one application for $\Psi\neq\mathbf{I}_{n\times n}$ is considered in numerical experiments. The regularization related to the ℓ_0-norm is formulated as

$$\min_{x} \quad \|b-\Phi x\|_{\ell_2}^2+2\tau\|x\|_{\ell_0}, \tag{3}$$

* Corresponding author at: KU Leuven, Department of Electrical Engineering (ESAT-STADIUS), B-3001 Leuven, Belgium.
E-mail addresses: huangxl06@mails.tsinghua.edu.cn (X. Huang), yipeng.liu@esat.kuleven.be (Y. Liu), leishi@fudan.edu.cn (L. Shi), sabine.vanhuffel@esat.kuleuven.be (S. Van Huffel), johan.suykens@esat.kuleuven.be (J.A.K. Suykens).

http://dx.doi.org/10.1016/j.sigpro.2014.09.028
0165-1684/© 2014 Elsevier B.V. All rights reserved.

where $\tau > 0$ and $\|b-\Phi x\|_{\ell_2}^2$ is the sum of squared residuals. For this regularization problem, the iterative hard thresholding algorithm [4,5] is applicable, which has the following formulation:

$$x_{l+1} = \mathbb{S}_{\text{hard}}(x_l + \Phi^T(b-\Phi x_l)), \tag{4}$$

where l is the iteration number and the shrinkage operator $\mathbb{S}_{\text{hard}}(x)$ is below:

$$(\mathbb{S}_{\text{hard}}(x))(i) = \begin{cases} x(i) & \text{if } |x(i)| \geq \tau, \\ 0 & \text{if } |x(i)| < \tau. \end{cases}$$

In this paper, $x(i)$ stands for the ith component of x.

Generally, the ℓ_0-norm related problems are hard to solve. To effectively recover sparse signals, some fast algorithms have been established, including orthogonal matching pursuit [6], CoSaMP [7], subspace pursuit [8], and the iterative hard thresholding. Another possibility is to use a penalty function $L(x)$: $\mathbb{R}^n \to \mathbb{R}$ to approach $\|x\|_{\ell_0}$ and then to solve $\min_x\{L(x) \text{ s.t. } \Phi x = b\}$ instead of (2). The most popular choice of the penalty function is the ℓ_1-norm, i.e., $L(x) = \|x\|_{\ell_1} = \sum_{i=1}^n |x(i)|$. Then we obtain the following ℓ_1 minimization problem:

$$\min_x \quad \|x\|_{\ell_1} \quad \text{s.t. } \Phi x = b, \tag{5}$$

which is also called the basis pursuit [9]. Based on the ℓ_1-norm, some interesting techniques including Dantzig selector [10] and decoding by linear programming [11], and practical solving algorithms, such as iterative soft thresholding algorithm [12], have been established and widely applied in many fields. The iterative soft thresholding algorithm is to solve the ℓ_1 regularization problem:

$$\min_x \quad \|b-\Phi x\|_{\ell_2}^2 + 2\tau\|x\|_{\ell_1}. \tag{6}$$

Its iterative process is similar to the iterative hard thresholding (4) and the related shrinkage operator is below

$$(\mathbb{S}_{\text{soft}}(x))(i) = \begin{cases} x(i) - \tau\,\text{sgn}(x(i)) & \text{if } |x(i)| \geq \tau, \\ 0 & \text{if } |x(i)| < \tau. \end{cases}$$

Minimizing the ℓ_1-norm is an efficient relaxation for (2). If Φ satisfies some conditions, such as the restricted isometry property [11], the null space property [13], and the incoherence condition [14], then the sparse signal can be recovered by solving (5). But due to the relaxation, the accuracy is degenerated [15]. Moreover, for some applications, the result of the ℓ_1 minimization is not sparse enough and the original signals cannot be recovered. Fig. 1 displays an interesting and simple example, which is given by [16] to intuitively explain the failure of the ℓ_1 minimization for some sparse signals. In this example, the real-valued signal is $x^0 = [0,\ 1,\ 0]^T$ and $\Phi = \left[\begin{smallmatrix} 2 & 1 & 1 \\ 1 & 1 & 2 \end{smallmatrix}\right]$. Our task is to recover x^0 from $b = \Phi x^0 = [1,\ 1]^T$. Unfortunately, the ℓ_1 minimization does not lead to the real signal: instead of x^0, $\tilde{x} = [1/3,\ 0,\ 1/3]^T$ is optimal to (5). Fig. 1(a) (i.e., Fig. 1 (b) in [16]) shows the reason: the feasible set intersects the interior of the ℓ_1 ball $\{x: \|x\|_{\ell_1} \leq \|x^0\|_{\ell_1}\}$. Fig. 1(b) plots the contour map of the ℓ_1-norm in the plane $x(2) = 0$. The solid line represents the level set $\{x: \|x\|_{\ell_1} = \|x^0\|_{\ell_1}\}$ and $\tilde{x}$ is marked by the red cross. We observe that $\tilde{x}$ is in the interior of $\{x: \|x\|_{\ell_1} \leq \|x^0\|_{\ell_1}\}$, which makes (5) fail to recover x^0.

Motivated by this observation, we know that one possible way of recovering x^0 is to replace the ℓ_1-norm

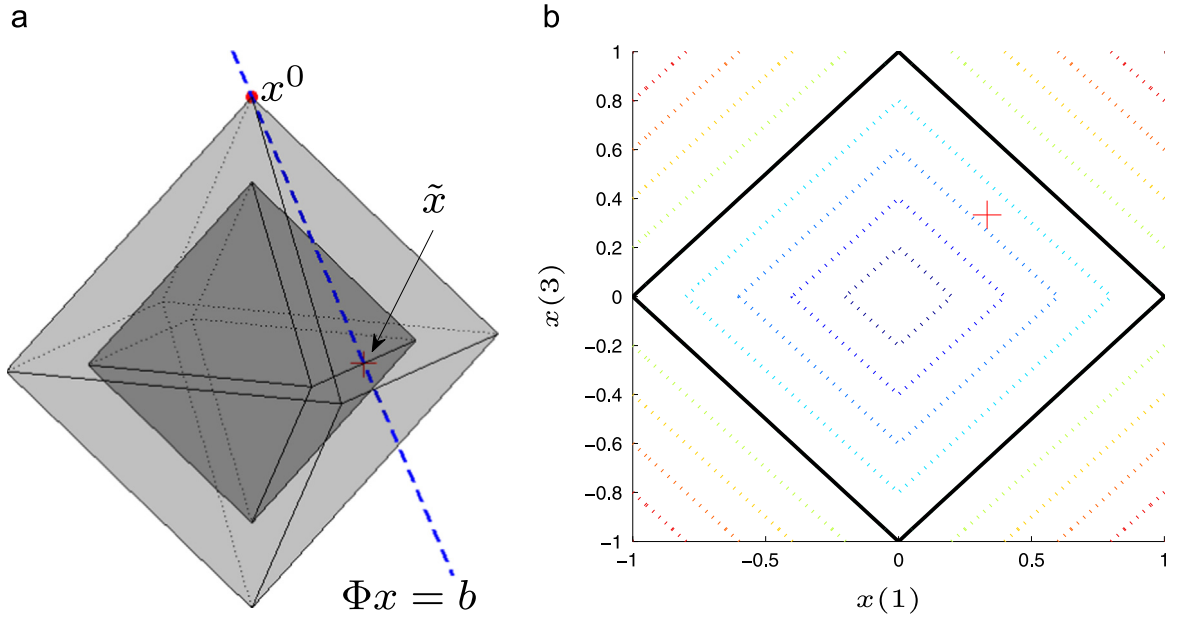


Fig. 1. An example that the ℓ_1 minimization cannot recover x^0: (a) (Fig. 1(b) in [16]) feasible set $\Phi x = b$ intersects the interior of the ℓ_1-norm ball $\{x: \|x\|_{\ell_1} \leq \|x^0\|_{\ell_1}\}$; (b) the contour map of the ℓ_1-norm in the plane $x(2) = 0$; the solid line is the level set $\{x: \|x\|_{\ell_1} = \|x^0\|_{\ell_1}\}$. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

by another penalty $L(x)$ to shrink the penalty ball $\{x: L(x) \le L(x^0)\}$. One successful attempt is the iteratively reweighted ℓ_1 minimization (IRWL1) developed by [16]. The scheme is to iteratively solve the following weighted problem:

$$x_{l+1} = \arg\min_x \left\{ \sum_{i=1}^{n} w_l(i)|x(i)| \text{ s.t. } \Phi x = b \right\} \tag{7}$$

where w_l is set as

$$w_l(i) = \frac{1}{|x_l(i)| + \varepsilon}. \tag{8}$$

IRWL1 is essentially minimizing a penalty of x, which is denoted by $\|\cdot\|_{\text{IRWL1}}$ and defined as

$$\|x\|_{\text{IRWL1}} = \sum_{i=1}^{n} \frac{1}{C(|x(i)| + \varepsilon)} |x(i)|$$

where $C = \sum_{i=1}^{n} 1/(|x(i)| + \varepsilon)$ is the normalization constant such that $\|w\|_{\ell_1} = 1$. We plot the contour map of $\|\cdot\|_{\text{IRWL1}}$ in Fig. 2(a), from which one can see its effect for enhancing sparsity. Assume that we have obtained x_l which is shown by the black square. Then IRWL1 (7) linearizes $\|\cdot\|_{\text{IRWL1}}$ and solves the linearized system, displayed by the dashed red lines, in order to get a better solution for $\min_x\{\|x\|_{\text{IRWL1}} \text{ s.t. } \Phi x = b\}$ from x_l.

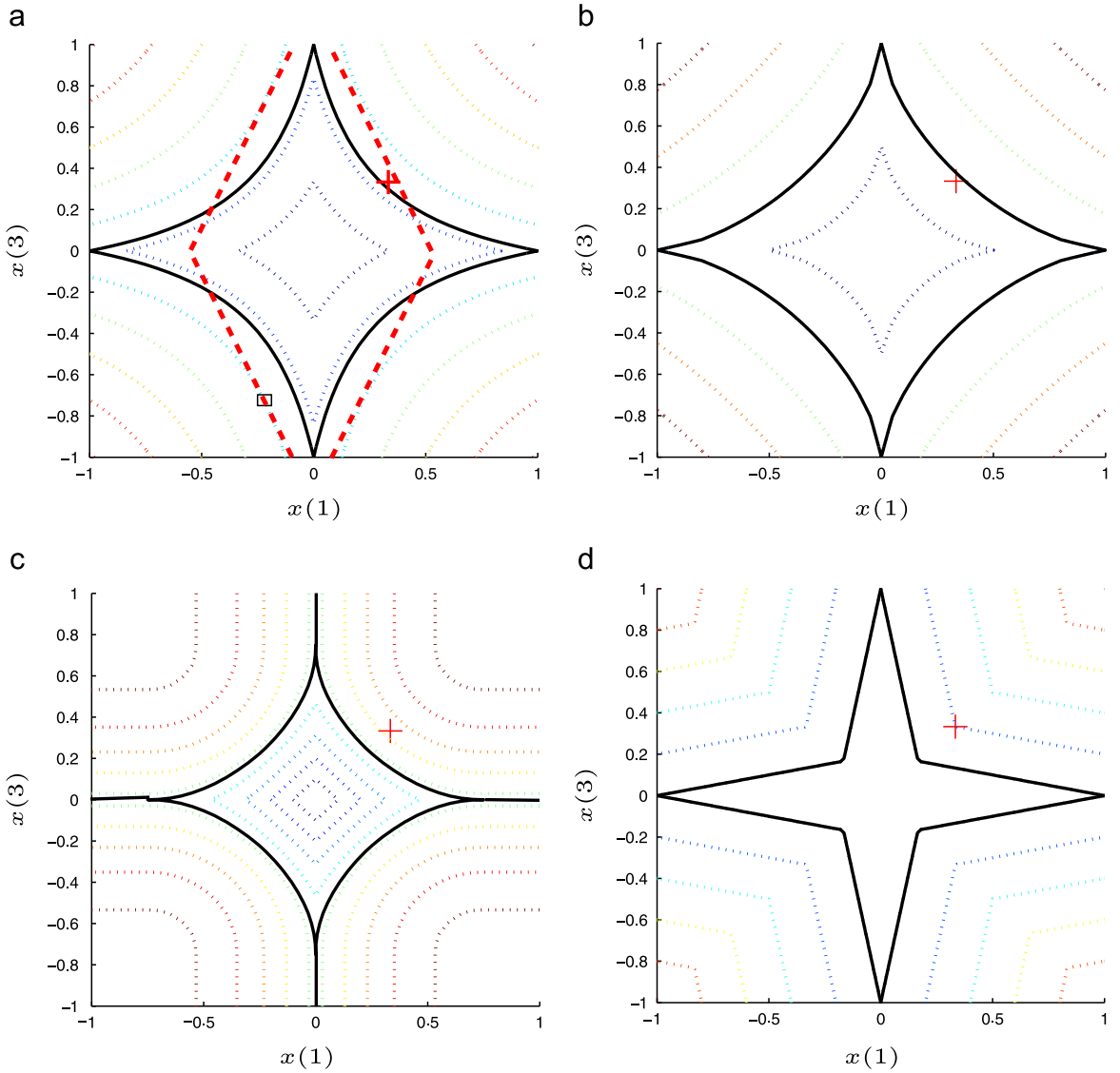


Fig. 2. The contour map of several non-convex penalties: (a) the iteratively reweighted ℓ_1-norm with $\varepsilon = 1$, the dashed line represents the weighted ℓ_1-norm used in IRWL1, which is essentially the local linearization of $\|\cdot\|_{\text{IRWL1}}$; (b) the ℓ_q-norm with $q = 2/3$; (c) the smoothly clipped absolute deviation penalty with $a_1 = 1/5, a_2 = 3/4$; (d) the expected penalty excludes $\tilde{x}$ and is piecewise linear. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

Besides $\|\cdot\|_{\mathrm{IRWL1}}$, the ℓ_q-norm $(0<q<1)$ has been well studied for compressed sensing, see, e.g., [17–21]. The definition of the ℓ_q-norm is given by

$$\|x\|_{\ell_q}=\left(\sum_{i=1}^{n}|x(i)|^q\right)^{1/q}.$$

The sparsity of the ℓ_q minimization $\min_x\{\|x\|_q^q \text{ s.t. } \Phi x=b\}$ is related to the value of q. For example, when $q=2/3$, $\tilde{x}$ cannot be excluded from the ℓ_q-norm ball, as shown in Fig. 2(b). While, when $q=1/2$, x^0 can be recovered. Generally, the performance of minimizing the ℓ_q-norm is improved but its computational complexity increases when decreasing q. For the ℓ_q minimization, one popular solving method is also based on linearization: we iteratively solve the reweighted ℓ_1 minimization, as shown in (7), but the weights are set as

$$w_l(i)=\frac{1}{(|x_l(i)|+\varepsilon)^{1-q}},$$

which is an approximation to the gradient of $|x(i)|^q$ at $x_l(i)$.

Another non-convex penalty which can be used to enhance the sparsity is the smoothly clipped absolute deviation penalty (SCAD) established in [22]. SCAD, of which the contour map is illustrated in Fig. 2(c), has two parameters a_1 and a_2. SCAD equals the ℓ_1-norm when $|x(i)|<a_1$, equals the $\ell_{1/2}-$norm when $a_1\le|x(i)|<a_2$, and equals the ℓ_0-norm when $|x(i)|\ge a_2$. One efficient local algorithm was given in [23] by applying the difference of convex functions (DC) programming, which was developed by [24,25]. The basic idea of DC programming is to do linearization using the gradient and minimize the linearized system. As analyzed previously, minimizing $\|\cdot\|_{\mathrm{IRWL1}}$ or the ℓ_q-norm is also based on local linearization, i.e., one uses the gradient information to construct a local linearized function then minimizes the approximate system. If the linearization is accurate over a wide range, the minimization technique based on the linearized system is effective.

The iterative thresholding algorithm, which is very efficient and particularly suitable for large-scale problems, is based on a linearized system as well. If the gradient changes significantly, it may be hard to construct an efficient thresholding algorithm. For example, to the best of our knowledge, the iterative thresholding algorithm for $\|\cdot\|_{\mathrm{IRWL1}}$ is not available, which limits the application of IRWL1 on large-scale problems. For the $\ell_{1/2}$ minimization, an iterative thresholding algorithm, called the half iterative thresholding algorithm, has been recently proposed by [20]. But its computational time is still significantly larger than the soft thresholding algorithm, since the half shrinkage operator involves cosine and arccosine.

Motivated by the discussion above, we in this paper will establish a new penalty, which can enhance the sparsity and is equal to a linear function in a large region. The contour map is shown in Fig. 2(d). In Section 2, we will give the corresponding analytic formulation. Generally, the proposed penalty considers two different penalty levels for different components according to the order of the component absolute values. This penalty is hence called a *two-level ℓ_1-norm*. Minimizing the proposed two-level ℓ_1-norm can be conducted in the framework of the reweighted ℓ_1 minimization (7) and the numerical experiments illustrate a better convergence behavior than IRWL1 and the ℓ_q minimization. Moreover, we can establish an efficient soft thresholding algorithm for the two-level ℓ_1-norm, due to its piecewise linearity. Generally, the two-level ℓ_1-norm has a similar sparse performance as the ℓ_q-norm or $\|\cdot\|_{\mathrm{IRWL1}}$, and enjoys computational effectiveness, especially its thresholding algorithm is available. Compared to existing reweighted methods, the key point is that we set weights according to the order not the value and then the penalty becomes piecewise linear. In this point of view, one can find a recent and parallel work in [26], which similarly gives weights by the order. They give heavy weights to components with large absolute values, while we give heavy weights to components with small absolute values, which is non-convex but enhances sparsity.

The remainder of the paper is organized as follows. The formulation of the two-level ℓ_1-norm is given in Section 2, where we also discuss some properties and establish an iterative algorithm. In Section 3, a soft thresholding algorithm for the two-level ℓ_1 minimization is established and its convergence is proved. The proposed algorithms are then evaluated by numerical experiments in Section 4. Section 5 ends the paper with concluding remarks.

2. Two-level ℓ_1 minimization

2.1. Two-level ℓ_1-norm

Define a penalty function

$$L^\rho(x)=\min\{\rho|x(1)|+|x(2)|+|x(3)|,|x(1)|+\rho|x(2)|+|x(3)|,|x(1)|+|x(2)|+\rho|x(3)|\}. \tag{9}$$

When $\rho=1$, $L^\rho(x)$ reduces to the ℓ_1-norm. With a decreasing value of ρ, the sparsity is enhanced. One can verify that $L^{1/5}(x)$ gives the contour map shown in Fig. 2(d). Fig. 3 plots $L^\rho(x)$ for some other ρ values.

Using (9), we got the expected penalty in Fig. 2(d). To establish the formulation in a higher dimensional space, for any vector x, we first introduce the index set of the largest K components of $|x(i)|$, denoted by $I(x)$, i.e.,

$$|I(x)|=K \text{ and } |x(i)|\ge|x(j)|,\quad \forall i\in I(x),\ \forall j\in I^c(x), \tag{10}$$

where I is a subset of $\{1,2,\ldots,n\}$, $|I|$ stands for the cardinality of I, and I^c is the complement of I with the respect to $\{1,2,\ldots,n\}$. In the rest of this paper, when we write an index set as a function with the respect to a vector, like $I(x)$, it implies that this index set satisfies condition (10). Using this notation, we define a new penalty

$$L_{2\mathrm{level}}^{\rho,K}(x)=\rho\sum_{i\in I(x)}|x(i)|+\sum_{i\in I^c(x)}|x(i)|. \tag{11}$$

One can verify that when $K=1$, $x\in\mathbb{R}^3$, the above expression reduces to (9). In (11), there are two parameters ρ and K: we give weight ρ for K components and weight 1 for the remainders. We hence call (11) the *two-level ℓ_1-norm*.

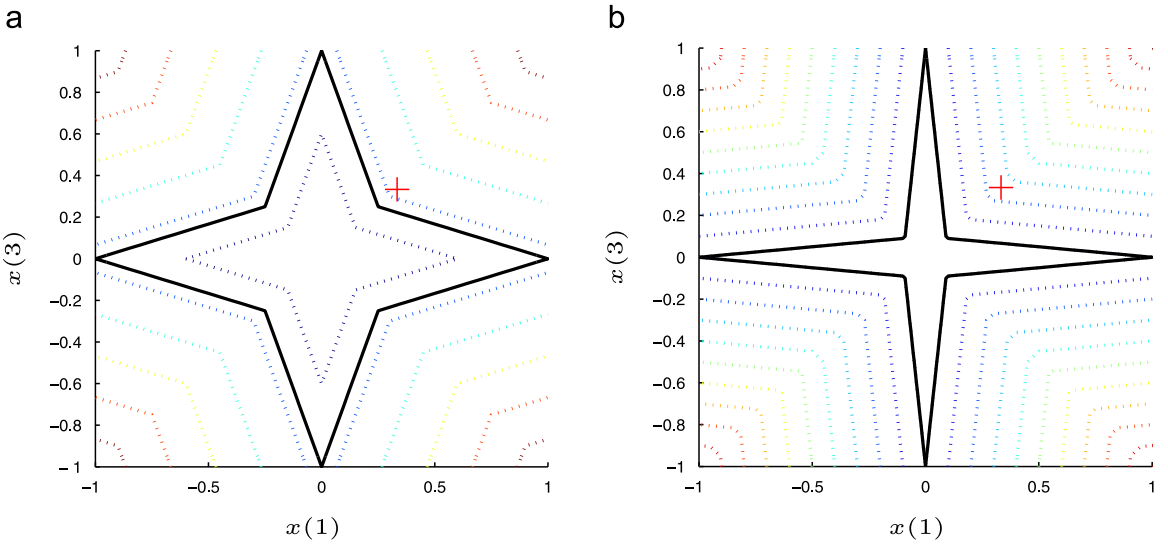


Fig. 3. The contour map of the two-level ℓ_1-norm: (a) $\rho=1/3$ and (b) $\rho=1/10$.

Minimizing the two-level ℓ_1-norm, i.e., $\min\{L^{\rho,K}_{2\text{level}}(x)$ s.t. $\Phi x=b\}$, can be formulated as

$$\min_{x}\ \min_{I:|I|=K}\left\{\rho\sum_{i\in I}|x(i)|+\sum_{i\in I^c}|x(i)|\right\}\quad \text{s.t. } \Phi x=b. \tag{12}$$

Obviously, (12) is equal to

$$\min_{x,I}\ \rho\sum_{i\in I}|x(i)|+\sum_{i\in I^c}|x(i)|\quad \text{s.t. } \Phi x=b,\ |I|=K. \tag{13}$$

When I is fixed, (13) becomes a linear programming problem. Thus, the two-level ℓ_1 minimization is one mixed integer linear programming problem (MILP). However, solving MILP is not practical when the problem size is too large. Though (12) can be written as an MILP, it is essentially a continuous problem and we will in the following establish a descent algorithm.

Any I satisfying (10) defines a polyhedron $D_I=\{x:|x(i)|\geq|x(j)|,\ \forall i\in I,\ \forall j\in I^c\}$, in which (11) remains the exact linearization of $L^{\rho,K}_{2\text{level}}(x)$. That means, in this polyhedron, the linearized function is exactly equal to the original one. In other words, for any x_0, there is

$$L^{\rho,K}_{2\text{level}}(x)=\rho\sum_{i\in I(x_0)}|x(i)|+\sum_{i\in I^c(x_0)}|x(i)|,\ \forall x\in D_{I(x_0)}. \tag{14}$$

Hence, we can minimize a linear function in one polyhedron to get a descent for $L^{\rho,K}_{2\text{level}}(x)$ from x_0. Specifically, we consider the following linear programming problem related to I:

$$\begin{aligned}\min_{x,z,t}\ & \rho\sum_{i\in I}z(i)+\sum_{i\in I^c}z(i)\\ \text{s.t.}\ & \Phi x=b,\quad z(i)\geq x(i),\quad z(i)\geq -x(i),\quad \forall i\in\{1,2,\ldots,n\}\\ & z(i)\geq t,\quad \forall i\in I,\quad z(i)\leq t,\ \forall i\in I^c.\end{aligned} \tag{15}$$

It is not hard to see that when we choose $I=I(x_0)$, x_0 (with $z(i)=|x_0(i)|,\ t=\min_{i\in I(x_0)}z(i)$) is feasible to (15) and solving (15) can decrease the objective value of (12) from $L^{\rho,K}_{2\text{level}}(x_0)$. If x_0 is a local optimum, it should be optimal to (15) for all polyhedra intersecting at x_0. Otherwise, the objective value can be decreased as shown previously. Moreover, the number of possible I is finite and hence repeating the above process leads to a local optimum of the two-level ℓ_1 minimization. The local optimality condition for a continuous piecewise linear problem can be found in [28]. Since (12) is non-convex, a good initial point is needed. In this paper, we use the result of the ℓ_1 minimization as the initial solution and give Algorithm 1.

Algorithm 1. Descent method for two-level ℓ_1-norm.

- Set l, i.e., the iteration counter, to be zero;
- Solve (5), denote the result as x_0 and select $I:=I(x_0)$ according to (10);

repeat

- Set $l:=l+1$;
- Solve (15) and denote the result as x_l;
- Select $I:=I(x_l)$ (if there are multiple choices, select the one different from $I(x_{l-1})$);

until $L^{\rho,K}_{2\text{level}}(x_l)=L^{\rho,K}_{2\text{level}}(x_{l-1})$

- Algorithm ends and return x_l.

2.2. Iteratively reweighted method for two-level ℓ_1-norm

Minimizing the two-level ℓ_1-norm by the above algorithm can enhance the sparsity from the ℓ_1 minimization. To speed up the local search for (12), we in this subsection develop another algorithm based on the iteratively reweighted method. In Algorithm 1, the locally linearized function (11) is minimized with constraint $x\in D_{I(x_l)}$, where x_l is the current solution. As analyzed previously, in this polyhedron, the linearization of $L^{\rho,K}_{2\text{level}}(x)$ is exact and hence minimizing the locally linearized function (11) within $D_{I(x_l)}$ decreases the two-level ℓ_1-norm from $L^{\rho,K}_{2\text{level}}(x_l)$. If this constraint is ignored, (15) becomes a weighted ℓ_1 minimization problem:

$$\min_{x}\ \sum_{i=1}^{n}w(i)|x(i)|,\quad \text{s.t. } \Phi x=b, \tag{16}$$

where $w(i)=\rho, i\in I(x_l)$ and $w(i)=1, i\in I^c(x_l)$. We can solve (16) to update x. This scheme is similar to IRWL1 (7). When

a solution is obtained, we sort the components by their absolute values, then set the weights according to the order. This is different to the existing reweighted methods, such as [16] and [27], which give weights based on the component values. The iteratively reweighted method for the two-level ℓ_1-norm is summarized in the following algorithm.

Algorithm 2. Reweighted method for two-level ℓ_1-norm.

- Set l:=0 and $l_{\max}$ (maximal number of iterations);
- Solve (5) and denote the result as x_0;

repeat

- Set l:=$l+1$;
- Set $w(i)$:=$\rho, i\in I(x_{l-1}),\ w(i)$:=$1, i\in I^c(x_{l-1})$ (if there are multiple choices for $I(x_{l-1})$, select the one has not been considered; if all the possible $I(x_{l-1})$ have been tried, stop the loop);
- Solve (16) and denote the result as x_l;

until $x_l = x_{l-1}$ or $l = l_{\max}$

- Algorithm ends and return x_l.

In this algorithm, the suggested parameters are $\rho=\frac{1}{10}$ and $K=\lfloor\frac{1}{3}\|x_0\|_{\ell_0}\rfloor$, where $\lfloor a\rfloor$ stands for the largest integer not greater than a. Performance with other parameter values are evaluated in numerical experiments. In Algorithm 2, the stopping criterion can be checked by $\|x_l-x_{l-1}\|_{\ell_2}$. In the experiments of this paper, the loop ends when $\|x_l-x_{l-1}\|_{\ell_2}\le 10^{-6}$. The basic idea of IRWL1 is that people care less about the components which have large absolute values, since they are not easy to become zero. The reweighted method for the two-level ℓ_1 has a similar idea: in Algorithm 2, we give a small penalty on the component which has a large absolute value. The difference is that we do not care about the magnitude of $|x(i)|$ but focus on the order of $|x(i)|$. The main advantage is that this modification reduces the computational time, which will be supported by numerical study and can be explained as that the effective region of linearization of the two-level ℓ_1-norm is wider than that of $\|\cdot\|_{\mathrm{IRWL1}}$.

Recently, a convergence analysis on IRWL1 has been given by [29] for the reweighted ℓ_1 regularization problem. But the convergence of (7) is still an open problem, as pointed out in [16]. Similarly, we currently cannot guarantee the convergence of Algorithm 2 neither. By comparison, Algorithm 1 converges to a local optimum of (12). However, the convergence needs many steps and solving (15) takes much more time than (16). Though the precision of Algorithm 2 is less than Algorithm 1, numerical experiments will show that Algorithm 2 already gives good sparse estimation performance and hence we prefer to use Algorithm 2 in practice.

2.3. Analysis based on null space property

As analyzed previously, for a decreasing ρ, minimizing the two-level ℓ_1-norm takes more computational time and leads to a more sparse result. If we take $\rho=0$, minimizing the two-level ℓ_1-norm has some similarity to the iterative support detection (ISD, [27]), which sets the weights for components larger than one threshold to be zero. Though ISD focuses on the value not the order of the absolute values of components, the analysis in [27] is applicable for the two-level ℓ_1-norm as well. The analysis for ISD is mainly based on the *null space property* (NSP). We use x_I to denote a vector satisfying

$$x_I(i)=\begin{cases} x(i), & i\in I,\\ 0, & i\notin I.\end{cases}$$

Then NSP can be expressed as follows.

Definition 1 (*null space property*). Consider a matrix $\Phi\in\mathbb{R}^{m\times n}$, an order s, and $\gamma>0$. If the following holds:

$$\|\eta_S\|_{\ell_1}\le\gamma\|\eta_{S^c}\|_{\ell_1},\quad \forall\eta\in\mathcal{N}(\Phi),\ \forall S:|S|\le s,$$

then we say that Φ has the null space property of order s for γ, where $\mathcal{N}(\Phi)$ stands for the null space of Φ.

NSP was defined in [13], where the relationship between NSP and the *restricted isometry property* (RIP [11]) was also given: if a matrix Φ has RIP of order k with constant δ, then Φ satisfies NSP of order s_1 for $\gamma=\sqrt{s_2/s_1}(1+\delta)/(1-\delta)$, where $s_1+s_2=k$. In [30,31], NSP has been applied to investigate the properties for the iteratively reweighted least squares minimization. For ISD analysis, NSP can be extended to the truncated-NSP, which was defined in [27] and is reviewed below.

Definition 2 (*truncated null space property*). Consider a matrix $\Phi\in\mathbb{R}^{m\times n}$, an order s, an integer h, and $\gamma>0$. If the following holds:

$$\|\eta_S\|_{\ell_1}\le\gamma\|\eta_{H\cap S^c}\|_{\ell_1},\ \forall\eta\in\mathcal{N}(\Phi),\quad \forall S:|S|\le s,\ \forall H:|H|=h,$$

then we say that Φ has truncated-NSP of order s for h,γ.

Using the truncated-NSP, we give a sufficient condition for successful recovery via the two-level ℓ_1 minimization.

Theorem 1. *Suppose $y=\Phi x^0$ and Φ has the truncated-NSP of order $\|x^0\|_{\ell_0}$ for K and γ. If $\gamma<1$ and $K\le n/2$, x^0 is the minimizer of (12).*

Proof. x^0 being a minimizer of (12) means that

$$L_{2\mathrm{level}}^{\rho,K}(x^0+v)\ge L_{2\mathrm{level}}^{\rho,K}(x_0),\quad \forall v\in\mathcal{N}(\Phi).$$

For any $v\in\mathcal{N}(\Phi)$, denote $I_0=I(x^0)$, $I_1=I(x^0+v)$, and $S^0=\mathrm{supp}(x^0)$, where $\mathrm{supp}(x^0)$ is the support set of x^0. For any index set $I\subseteq\{1,2,\ldots,n\}$, there is

$$\|x_I^0\|_{\ell_1}=\|x^0_{I\cap S^0}\|_{\ell_1}.$$

Therefore,

$$\begin{aligned}L_{2\mathrm{level}}^{\rho,K}(x^0+v)&=\rho\|x^0_{I_1}+v_{I_1}\|_{\ell_1}+\|x^0_{I_1^c}+v_{I_1^c}\|_{\ell_1}=\rho(\|x^0_{I_1\cap S^0}+v_{I_1\cap S^0}\|_{\ell_1}\\&\quad+\|v_{I_1\cap S^{0c}}\|_{\ell_1})+\|x^0_{I_1^c\cap S^0}+v_{I_1\cap S^0}\|_{\ell_1}+\|v_{I_1^c\cap S^{0c}}\|_{\ell_1}.\end{aligned}$$

Considering the first item, we have

$$\begin{aligned}&\|x^0_{I_1\cap S^0}+v_{I_1\cap S^0}\|_{\ell_1}+\|v_{I_1\cap S^{0c}}\|_{\ell_1}=\|x^0_{I_1\cap S^0}+v_{I_1\cap S^0}\|_{\ell_1}\\&\quad-\|x^0_{I_1\cap S^0}\|_{\ell_1}+\|v_{I_1\cap S^0}\|_{\ell_1}+\|x^0_{I_1\cap S^0}\|_{\ell_1}+\|v_{I_1\cap S^{0c}}\|_{\ell_1}-\|v_{I_1\cap S^0}\|_{\ell_1}\\&\quad\ge\|x^0_{I_1}\|_{\ell_1}+(\|v_{I_1\cap S^{0c}}\|_{\ell_1}-\|v_{S^0}\|_{\ell_1})\ge\|x^0_{I_1}\|_{\ell_1}.\end{aligned}$$

The last inequality comes from the condition that Φ has the truncated-NSP of order $\|x^0\|_{\ell_0}$ for K and γ.

Similarly, if $K\le n/2$, the truncated-NSP guarantees that

$$\|x^0_{I_1^c\cap S^0}+v_{I_1^c\cap S^0}\|_{\ell_1}+\|v_{I_1^c\cap S^{0c}}\|_{\ell_1}\ge\|x^0_{I_1^c}\|_{\ell_1},$$

and hence $L^{\rho,K}_{2\text{level}}(x^0+v) \geq \rho\|x^0_{I_1}\|_{\ell_1} + \|x^0_{I_1^c}\|_{\ell_1}$. According to the definition of $I(x^0)$, we know that I_0 includes k components with the largest absolute values $|x^0(i)|$, thus $\rho\|x^0_{I_1}\|_{\ell_1} + \|x^0_{I_1^c}\|_{\ell_1} \geq \rho\|x^0_{I_0}\|_{\ell_1} + \|x^0_{I_0^c}\|_{\ell_1} = L^{\rho,K}_{2\text{level}}(x^0)$, which follows that

$$L^{\rho,K}_{2\text{level}}(x^0+v) \geq L^{\rho,K}_{2\text{level}}(x^0), \quad \forall v \in \mathcal{N}(\Phi),$$

and this theorem is proved. □

Theorem 1 gives the condition on the measurement matrix Φ for successful recovery. However, due to the non-convexity of the two-level ℓ_1-norm, Algorithms 1 and 2 cannot guarantee the global optimality, which prevents the use of Theorem 1 on the recovery analysis for the results of the proposed algorithms. Instead, we can illustrate the performance by the phase transition diagram. We randomly generate a 100-dimensional k-sparse signal x^0, i.e., there are k non-zero components, and a matrix Φ with m measurements. The non-zero components of the signal and the elements of Φ follow the standard Gaussian distribution. From $b = \Phi x^0$, we recover the signal by Algorithm 2 with the suggested parameter values. When the maximal absolute error between the recovered signal and x^0 is smaller than 10^{-3}, we say that the signal is successfully recovered. For each m and k, we test 300 trials and display the percentage of successful recovery in Fig. 4. In this figure, the 50% successful recovery lines by the ℓ_1 minimization (5) and the $\ell_{2/3}$ minimization [33] are given as well for reference. Minimizing the $\ell_{2/3}$−norm can enhance the sparsity. The proposed two-level ℓ_1-norm leads to a similar (slightly better) performance. The attractive property is that the two-level ℓ_1-norm is piecewise linear and hence its convergence behavior is better. Based on the piecewise linearity, in the next section, we will develop a fast algorithm and then compare the computation time in Section 5.

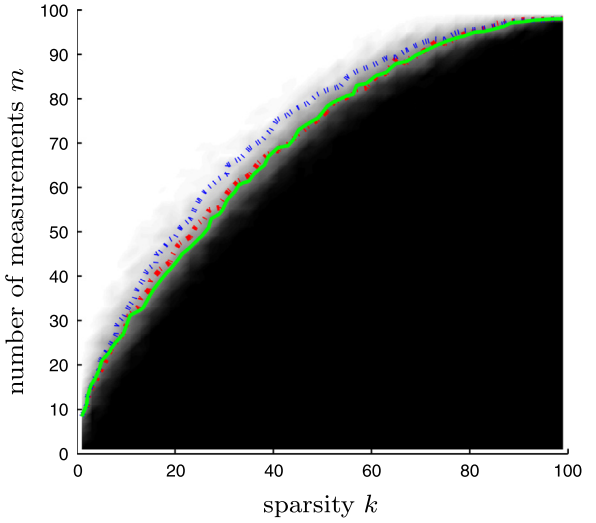


Fig. 4. Phase transition diagram for Algorithm 2. The grey level stands for the successful recovery percentage, white means 100% recovery and black means 0% recovery, the 50% recovery is also displayed by the green solid line. The blue and red dotted lines show the 50% successful recovery by the ℓ_1 minimization (5) and the $\ell_{2/3}$ minimization [33], respectively. Compared with $\ell_{2/3}$ minimization, the two-level ℓ_1 minimization has slightly better sparsity and can be effectively solved. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

3. Thresholding algorithm for two-level ℓ_1-norm

To suppress additive noise in measurements, one penalizes the violation of $\Phi x = b$ and modifies the ℓ_1 minimization (5) into the regularization problem (6). For (6), an iterative soft thresholding algorithm has been established in [12] and applied widely. The mentioned non-convex penalties are applicable to construct a regularization problem $\min_x\{\|\Phi x - b\|^2_{\ell_2} + 2\tau L(x)\}$, where $L(x)$ could be the ℓ_q-norm, $\|\cdot\|_{\text{IRWL1}}$, SCAD penalty, and the proposed two-level ℓ_1-norm.

To effectively solve the regularization problem, an efficient iterative thresholding algorithm is needed, especially for large-scale problems. In this section, we will establish such a thresholding algorithm for the following problem:

$$\min_x \quad \|\Phi x - b\|^2_{\ell_2} + 2\tau L^{\rho,K}_{2\text{level}}(x). \tag{17}$$

Define a two-level soft shrinkage operator $\mathbb{S}^{\tau,K}$ below

$$(\mathbb{S}^{\tau,K,\rho}(x))(i) = \begin{cases} x(i) - \rho\tau\ \text{sgn}(x(i)), & \text{if } i \in I(x),\ |x(i)| \geq \rho\tau, \\ 0 & \text{if } i \in I(x),\ |x(i)| < \rho\tau, \\ x(i) - \tau\ \text{sgn}(x(i)) & \text{if } i \in I^c(x),\ |x(i)| \geq \tau, \\ 0 & \text{if } i \in I^c(x),\ |x(i)| < \tau. \end{cases}$$

Notice that there could be several index sets $I(x)$ satisfying (10), i.e., there are multiple components having the Kth largest absolute value.

Considering the first-order derivative condition, we know that a stationary point of (17) should satisfy

$$x = \mathbb{S}^{\tau,K,\rho}(x + \Phi^T(b - \Phi x)). \tag{18}$$

For regular non-convex problems, the first-order derivative condition is not sufficient for local optimality. Since the two-level ℓ_1-norm is continuous piecewise linear, we can verify that x^* satisfying (18) is a local optimum of (17), which is concluded in the following theorem.

Theorem 2. *If for all possible $I(x^* + \Phi^T(b - \Phi x^*))$ defining $\mathbb{S}^{\tau,K,\rho}(x^* + \Phi^T(b - \Phi x^*))$, there are*

$$x^* = \mathbb{S}^{\tau,K,\rho}(x^* + \Phi^T(b - \Phi x^*)), \tag{19}$$

then x^* is a local optimum for (17).

Proof. We first investigate possible $I(x^* + \Phi^T(b - \Phi x^*))$. When x^* is given, $x^* + \Phi^T(b - \Phi x^*)$ can be calculated. We sort its components by the absolute value and denote the Kth largest value by t_K. Then one can find three index sets: $I^+ = \{i: \ |(x^* + \Phi^T(b - \Phi x^*))(i)| > t_K\}$, $I^0 = \{i: |(x^* + \Phi^T(b - \Phi x^*))\ (i)\ | = \ t_K\}$, and $I^- = \{i: |(x^* + \Phi^T(b - \Phi x^*))(i)| < t_K\}$. Arbitrarily selecting $K - |I^+|$ elements from I^0, denoted by I^0_+, and combining them with I^+, we can construct an $I(x^* + \Phi^T(b - \Phi x^*))$ satisfying (10) for $x^* + \Phi^T(b - \Phi x^*)$. In other words,

$$I(x^* + \Phi^T(b - \Phi x^*)) = I^0_+ \cup I^+ \quad \text{where } I^0_+ \subseteq I^0,\ |I^0_+| = K - |I^+|.$$

From the continuity and the definition of I^- and I^+, one can always find a small neighborhood of x^*, denoted by $\mathcal{O}(x^*)$, such that for any $x\in\mathcal{O}(x^*)$, there is

$$|(x+\Phi^T(b-\Phi x))(i)|<|(x+\Phi^T(b-\Phi x))(j)|<|(x+\Phi^T(b-\Phi x))(k)|,$$
$$\forall i\in I^-,\ j\in I^0,\ k\in I^+.$$

Once $I(x^*+\Phi^T(b-\Phi x^*))$, i.e., $I^0_+\cup I^+$, has been selected, one can verify that

$$|x^*(i)|\geq|x^*(j)|,\quad \forall i\in I^0_+\cup I^+,\ j\notin I^0_+\cup I^+. \tag{20}$$

To verify (20), we first consider the case $t_K>\rho\tau$. Then (19) for $i\in I^0_+\cup I^+$ tells us

$$(\Phi^T(b-\Phi x^*))(i)=\rho\tau\ \mathrm{sgn}(\Phi^T(b-\Phi x^*)(i)),$$

which follows that $|x^*(i)+\Phi^T(b-\Phi x^*)(i)|=|x^*(i)+\rho\tau\ \mathrm{sgn}(\Phi^T(b-\Phi x^*)(i))|\geq t_K$, i.e., $|x^*(i)|\geq t_K-\rho\tau$. Meanwhile, for $j\notin I^0_+\cup I^+$, there is

$$x^*(j)=x^*(j)+\Phi^T(b-\Phi x^*)(j)-\tau\ \mathrm{sgn}(\Phi^T(b-\Phi x^*)(j))\quad\text{or}\quad x^*(j)=0.$$

Thus, $|x^*(j)|\leq t_K-\tau$ or $x^*(j)=0$, which follows that $|x^*(i)|\geq|x^*(j)|$.

Another and simpler case is $t_K\leq\rho\tau$. In this case, $|(x^*+\Phi^T(b-\Phi x^*))(j)|\leq\rho\tau$, $\forall j\notin I^0_+\cup I^+$, which together with (19) imply that $x^*(j)=0$.

The union of I^+ and a given I^0_+ defines a corresponding subregion

$$D_{I^0_+\cup I^+}=\{x:|x_i|\geq|x_j|, i\in I^0_+\cup I^+, j\notin I^0_+\cup I^+\}.$$

Because of (20), there are

$$x^*\in\bigcap_{I^0_+\subseteq I^0,|I^0_+|=K-|I^+|} D_{I^0_+\cup I^+}, \tag{21}$$

and

$$\mathcal{O}(x^*)\subset\bigcup_{I^0_+\subseteq I^0,|I^0_+|=K-|I^+|} D_{I^0_+\cup I^+}. \tag{22}$$

For each I^0_+, we know the following two facts:

1. If (19) is true for $I^0_+\cup I^+$, x^* is the minimum of $\|\Phi x-b\|^2_{\ell_2}+2\tau\rho\sum_{i\in I^0_+\cup I^+}|x(i)|+2\tau\sum_{i\notin I^0_+\cup I^+}|x(i)|$;
2. $L^{\rho,K}_{2\text{level}}(x)=\rho\sum_{i\in I^0_+\cup I^+}|x(i)|+\sum_{i\notin I^0_+\cup I^+}|x(i)|$, $\forall x\in D_{I^0_+\cup I^+}$.

In other words, (19) for $I^0_+\cup I^+$ guarantees that x^* is the minimum of $\|\Phi x-b\|^2_{\ell_2}+2\tau L^{\rho,K}_{2\text{level}}(x)$ in $D_{I^0_+\cup I^+}$. Together with (22), we know that

$$\|\Phi x^*-b\|^2_{\ell_2}+2\tau L^{\rho,K}_{2\text{level}}(x^*)\leq\|\Phi x-b\|^2_{\ell_2}+2\tau L^{\rho,K}_{2\text{level}}(x),\quad\forall x\in\mathcal{O}(x^*),$$

which equivalently means that x^* is locally optimal to (17). Moreover, (21) indicates that the local optimality requires (19) for all possible I^0_+. □

In Theorem 2, when $|I^0|>1$, we need to check (18) for multiple $I(x+\Phi^T(b-\Phi x))$. Suppose $i\in I^0$ and there are more than one elements in I^0. Then we can set $i\in I^0_+$ or $i\notin I^0_+$. If $t_K>\rho\tau$, (18) with $i\in I^0_+$ requires that

$$(\Phi^T(b-\Phi x))(i)=\rho\tau\ \mathrm{sgn}(\Phi^T(b-\Phi x)(i)). \tag{23}$$

Meanwhile, (18) with $i\notin I^0_+$ requires that

$$(\Phi^T(b-\Phi x))(i)=\tau\ \mathrm{sgn}(\Phi^T(b-\Phi x)(i)),$$

which conflicts with (23). Therefore, when t_K, the Kth largest absolute component value of $x+\Phi^T(b-\Phi x)$, is larger than $\rho\tau$, $|I^0|=1$ is a necessary condition for the validity of (18). If $|I^0|>1$, one can easily find an $I(x+\Phi^T(b-\Phi x))$ against (18). On the other hand, if $t_K\leq\rho\tau$, putting $i\in I^0$ in I^0_+ or not both require that $x(i)=0$. In that case, (18) being valid for one $I(x+\Phi^T(b-\Phi x))$ is sufficient for all $I(x+\Phi^T(b-\Phi x))$. In general, though there could be multiple choices $I(x+\Phi^T(b-\Phi x))$ in the optimality condition (18), one can still use (18) to check the optimality easily.

Motivated by the optimality condition (18), we establish the following iterative soft thresholding algorithm for (17).

Algorithm 3. Two-level soft thresholding algorithm.

- Set l:=0 and $x_0:=\Phi^Tb$;

repeat

- $x_{l+1}:=\mathbb{S}^{\tau,K,\rho}(x_l+\Phi^T(b-\Phi x_l))$;
- Set l:=l+1;

until $x_l=x_{l-1}$

- Algorithm ends and return x_l.

Notice that there could be multiple choices when defining $\mathbb{S}^{\tau,K,\rho}$. In that case, we choose the one for which (18) is not valid for the current solution.

The local optimality condition (18) means that the stationary point of the two-level soft thresholding algorithm is a local optimum of (17). In the following, we will prove that Algorithm 3 converges to a local optimum. The proof is different from the soft thresholding algorithm proposed by [12] for the ℓ_1-norm, since $\mathbb{S}^{\tau,K,\rho}$ itself is not non-expansive, i.e., there exist $x,y\in\mathbb{R}^n$ such that

$$\|\mathbb{S}^{\tau,K,\rho}(x)-\mathbb{S}^{\tau,K,\rho}(y)\|>\|x-y\|. \tag{24}$$

A simple example is that $K=1$, $\rho=0.5$, and $\tau=1$. In this case, $\mathbb{S}^{\tau,K,\rho}(x)$ shrink the largest component by 0.5 and the others by 1. Then consider a two-dimensional vector $x=[1,\ 2]^T$, for which $\mathbb{S}^{\tau,K,\rho}(x)=[0,\ 1.5]^T$. While, for $y=[2,\ 1]^T$, there is $\mathbb{S}^{\tau,K,\rho}(y)=[1.5,\ 0]^T$. Via this simple example, one can verify the inequality (24), i.e., $\mathbb{S}^{\tau,K,\rho}$ is not non-expansive. Compared with the soft shrinkage operator $\mathbb{S}_{\text{soft}}$, the key difference is that $\mathbb{S}^{\tau,K,\rho}$ may give a smaller shrinkage to a larger component. This property is the reason that $\mathbb{S}^{\tau,K,\rho}$ can enhance sparsity but it results in non-expansivity. Instead of the analysis based on a non-expansive operator, the convergence of Algorithm 3 is guaranteed by Theorem 3, where $\|\Phi\|_2$ denotes the 2-norm of matrix Φ. The general convergence (to a stationary solution) analysis of this kind of non-smooth non-convex shrinkage methods can be found in [32] as well. In this paper, we focus on Algorithm 3 and discuss its convergence to a local optimum with the help of a surrogate function. From the discussion, one can also find the link between the two-level ℓ_1-norm and the ℓ_1-norm.

Theorem 3. *If $\|\Phi\|_2<1$, Algorithm 3 converges to a local optimum of (17).*

Proof. Algorithm 3 defines the following update formulation:

$$y_l = x_l + \Phi^T(b-\Phi x_l) \quad \text{and} \quad x_{l+1} = \mathbb{S}^{\tau,K,\rho}(y_l).$$

We will show that the process above can iteratively decrease the value of $\|\Phi x-b\|_{\ell_2}^2 + 2\tau L_{2\text{level}}^{\rho,K}(x)$, i.e.,

$$\|\Phi x_{l+1}-b\|_{\ell_2}^2 + 2\tau L_{2\text{level}}^{\rho,K}(x_{l+1}) \le \|\Phi x_l-b\|_{\ell_2}^2 + 2\tau L_{2\text{level}}^{\rho,K}(x_l).$$

At the first stage, we need to verify that for any y_l, $x_{l+1} = \mathbb{S}^{\tau,K,\rho}(y_l)$ is the minimizer of

$$g_{y_l}(x) = \|x-y_l\|_{\ell_2}^2 + 2\tau L_{2\text{level}}^{\rho,K}(x).$$

For any given x, $g_{y_l}(x)$ can be written as a separable function $g_{y_l}(x) = \sum_{i=1}^n g_{y_l,I(x)}^i(x(i))$, where

$$g_{y_l,I(x)}^i(x(i)) = \begin{cases} (y_l(i)-x(i))^2 + 2\rho\tau|x(i)|, & i\in I(x), \\ (y_l(i)-x(i))^2 + 2\tau|x(i)|, & i\notin I(x). \end{cases}$$

Suppose x^* is the minimum of $g_{y_l}(x)$. There could be multiple choices for $I(x^*)$ and the following statement is valid for all of them. Simple calculation for the minimum of $g_{y_l,I(x^*)}^i$ shows that for any $i\in I(x^*)$,

$$x^*(i) = \begin{cases} y_l(i) - \rho\tau\,\text{sgn}(y_l(i)), & |y_l(i)| \ge \rho\tau, \\ 0, & |y_l(i)| < \rho\tau. \end{cases}$$

Similarly, for any $i\in I^c(x^*)$,

$$x^*(i) = \begin{cases} y_l(i) - \tau\,\text{sgn}(y_l(i)), & |y_l(i)| \ge \tau, \\ 0, & |y_l(i)| < \tau. \end{cases}$$

In the following, we show that $g_{y_l}(x_{l+1}) \le g_{y_l}(x^*)$. Recalling the formulation of $x_{l+1} = \mathbb{S}^{\tau,K,\rho}(y_l)$, one can verify that

$$x^*(i) = x_{l+1}(i), \forall i \in \left(I(y_l)\cap I(x^*)\right) \cup \left(I^c(y_l)\cap I^c(x^*)\right). \tag{25}$$

We need to focus on $I(y_l)\bigcap I^c(x^*)$ only. If there exists an i such that $i\in I(y_l)\bigcap I^c(x^*)$, one can find another index such that $j\in I^c(y_l)\bigcap I(x^*)$, since $|I(y_l)| = |I(x^*)|$. Then $|y_l(i)| > |y_l(j)|$ and $|x^*(i)| < |x^*(j)|$. Notice that here we only discuss strict inequalities: if $|y_l(i)| = |y_l(j)|$, we can put j into $I(y_l)$ to replace i. The new set satisfies (10) and the discussion goes back to (25). Similarly, we do not need to consider $|x^*(i)| = |x^*(j)|$ in the following.

There are four possible cases:

1. $|y_l(i)| \le \tau, |y_l(j)| \le \rho\tau$, then $x^*(i) = x^*(j) = 0$, which conflicts with the assumption $|x^*(i)| < |x^*(j)|$;
2. $|y_l(i)| \le \tau, |y_l(j)| > \rho\tau$, then

$$\begin{aligned} & g_{y_l,I(x^*)}^i(x^*(i)) + g_{y_l,I(x^*)}^j(x^*(j)) \\ = \;& y_l^2(i) + \rho^2\tau^2 + 2\rho\tau|y_l(j) - \rho\tau\,\text{sgn}(y_l(j))| \\ \ge \;& \rho^2\tau^2 + 2\rho\tau|y_l(i) - \rho\tau\,\text{sgn}(y_l(i))| + y_l^2(j) \\ = \;& g_{y_l,I(y_l)}^i(y_l(i) - \rho\tau\,\text{sgn}(y_l(i))) + g_{y_l,I(y_l)}^j(0) \\ = \;& g_{y_l,I(x_{l+1})}^i(x_{l+1}(i)) + g_{y_l,I(x_{l+1})}^j(x_{l+1}(j)); \end{aligned}$$

3. $|y_l(i)| > \tau, |y_l(j)| \le \rho\tau$, then $x^*(i) = y_l(i) - \tau\,\text{sgn}\,(y_l(i))$ and $x^*(j) = 0$, which conflicts with the assumption $|x^*(i)| < |x^*(j)|$;
4. $|y_l(i)| > \tau, |y_l(j)| > \rho\tau$, then

$$\begin{aligned} & g_{y_l,I(x^*)}^i(x^*(i)) + g_{y_l,I(x^*)}^j(x^*(j)) \\ = \;& \tau^2 + 2\tau|y_l(i) - \tau\,\text{sgn}(y_l(i))| + \rho^2\tau^2 + 2\rho\tau|y_l(j) - \rho\tau\,\text{sgn}(y_l(j))| \\ \ge \;& \tau^2 + 2\rho\tau|y_l(i) - \rho\tau\,\text{sgn}(y_l(i))| + \rho^2\tau^2 + 2\tau|y_l(j) - \tau\,\text{sgn}(y_l(j))| \\ = \;& g_{y_l,I(x_{l+1})}^i(x_{l+1}(i)) + g_{y_l,I(x_{l+1})}^j(x_{l+1}(j)). \end{aligned}$$

These discussions mean that $g_{y_l}(x_{l+1}) \le g_{y_l}(x^*)$, i.e.,

$$x_{l+1} = \min_x \left\{ \|x-y_l\|_{\ell_2}^2 + 2\tau L_{2\text{level}}^{\rho,K}(x) \right\}.$$

As long as $\|\Phi\|_2 < 1$, one can verify that

$$\begin{aligned} & \|\Phi x_{l+1}-b\|_{\ell_2}^2 + 2\tau L_{2\text{level}}^{\rho,K}(x_{l+1}) \\ & \le \|\Phi x_l-b\|_{\ell_2}^2 + 2\tau L_{2\text{level}}^{\rho,K}(x_{l+1}) \\ & \quad + (x_{l+1}-x_l)^T\Phi^T(\Phi x_l-b) + \frac{1}{2}\|x_{l+1}-x_l\|_{\ell_2}^2 \\ & = \|\Phi x_l-b\|_{\ell_2}^2 - \frac{1}{2}\left\|\Phi^T(\Phi x_l-b)\right\|_{\ell_2}^2 \\ & \quad + 2\tau L_{2\text{level}}^{\rho,K}(x_{l+1}) + \frac{1}{2}\left\|x_{l+1}-x_l+\Phi^T(\Phi x_l-b)\right\|_{\ell_2}^2 \end{aligned}$$

As proved, $x_{l+1} = \mathbb{S}^{\tau,K,\rho}(y_l)$ is the minimizer of $\|x-y_l\|_{\ell_2}^2 + 2\tau L_{2\text{level}}^{\rho,K}(x)$, where

$$y_l = x_l + \Phi^T(b-\Phi x_l).$$

Therefore,

$$\begin{aligned} & \|\Phi x_{l+1}-b\|_{\ell_2}^2 + 2\tau L_{2\text{level}}^{\rho,K}(x_{l+1}) \\ & \le \|\Phi x_l-b\|_{\ell_2}^2 - \frac{1}{2}\left\|\Phi^T(\Phi x_l-b)\right\|_{\ell_2}^2 + 2\tau L_{2\text{level}}^{\rho,K}(x_l) \\ & \quad + \frac{1}{2}\left\|\Phi^T(\Phi x_l-b)\right\|_{\ell_2}^2 \\ & = \|\Phi x_l-b\|_{\ell_2}^2 + 2\tau L_{2\text{level}}^{\rho,K}(x_l). \end{aligned}$$

According to the above inequality, the iterations defined by Algorithm 3 provide a non-increasing sequence of the objective function until

$$x_{l+1} = \mathbb{S}^{\tau,K,\rho}(x_l + \Phi^T(b-\Phi x_l)),$$

which means that x_{l+1} meets (18) for the currently selected $I(x_l+\Phi^T(b-\Phi x_l))$. Notice that (18) needs to be satisfied for all possible choice to guarantee the local optimality of (17). As explained before, from the current $I(x_l+\Phi^T(b-\Phi x_l))$, one can easily verify that (18) is valid for all the selections, or quickly find a new set to continue the iteration. Therefore, when Algorithm 3 finally converges, i.e., we cannot find a new solution by two-level soft thresholding, we arrive at a local optimum of (17). □

4. Numerical experiments

We have introduced the two-level ℓ_1-norm (11), where there are two parameters: the number of components having a small penalty and the weight for such components. Parameter ρ gives a trade-off between sparsity and computational complexity, which can be observed from the contour map of $L_{2\text{level}}^{\rho,K}(x)$. For a smaller ρ, the two-level ℓ_1-norm is more similar to the ℓ_0-norm, i.e., the global minimizer of $L_{2\text{level}}^{\rho,K}(x)$ is more sparse. But at the same time, there are more local optima, which makes it harder to get the global optimum. In the following experiments, we will test the performance of the two-level ℓ_1 minimization with $\rho = 0, 1/10$, and $1/3$.

For any $\rho<1$, $L^{\rho,K}_{2\text{level}}(x)$ is non-convex. The established algorithms do not have global search capability. We use the result of the ℓ_1 minimization, denoted by x_0, as the starting point for Algorithm 2. There are $\|x_0\|_{\ell_0}$ non-zero components in x_0 and we minimize the two-level ℓ_1-norm to enhance the sparsity from x_0. $L^{\rho,K}_{2\text{level}}(x)$ gives less penalty for K components and tries to make the other $n-K$ components to be zero. Hence, when we have no prior knowledge about the sparsity of real signals, we set K according to x_0. In the experiments, we test several K values and suggest $K=\lfloor\frac{1}{3}\|x_0\|_{\ell_0}\rfloor$. The proposed two-level soft thresholding algorithm (Algorithm 3) does not calculate x_0, but we can expect that $\|x_0\|_{\ell_0}$ is less than m, the number of measurements. Hence, for Algorithm 3, K can be heuristically chosen to be $K=\lfloor m/5\rfloor$. All the experiments are done in Matlab R2011a in Core i5-1.80 GHz, 4.00 G RAM.

4.1. Simulated sparse signal recovery

We first evaluate the performance of the two-level ℓ_1 minimization on a sparse signal recovery problem. We randomly generate a k-sparse signal $x^0\in\mathbb{R}^n$, i.e., $\|x^0\|_{\ell_0}=k$, and a sampling matrix $\Phi\in\mathbb{R}^{m\times n}$. Then we solve (12) to recover the signal from the observed $b=\Phi x^0$. In this paper, the experiment used in [16] is repeated: $n=256$, $m=100$, the components of x^0 and the elements of Φ come from the standard Gaussian distribution. If the maximum absolute error between the obtained result and x^0 is smaller than 10^{-3}, we say that the signal is recovered. A different sparsity k is considered. For each k value, we repeat the above process 300 times, record the recovery percentage and the average computational time.

In this experiment, we recover the signal by Algorithm 2 in comparison with the results of the ℓ_1 minimization, the iteratively reweighted ℓ_1 minimization, the iteratively reweighted least squares (IRLS, [34]), and the ℓ_q minimization with $q=\frac{1}{2}$, $\frac{2}{3}$. The ℓ_q minimization is solved by the reweighted method as well. Except IRLS, those methods involve linear programming and we use the linear programming solver embedded in Matlab. As introduced previously, there is one parameter in the update formulation (8) of the iteratively reweighted ℓ_1 minimization. Besides, one should give the maximum number of iterations $l_{\max}$. According to the result given in [16], we set $\varepsilon=0.1$ and $l_{\max}=4$. For the ℓ_q minimization, the parameter setting follows the suggestion in [33]. When applying Algorithm 2 to minimize the two-level ℓ_1-norm for signal recovery, we also set $l_{\max}=4$ and consider $\rho=0$, $\frac{1}{10}$ and $\frac{1}{3}$. The recovery percentage and the time are reported in Fig. 5, where the black dashed-dotted lines correspond to the ℓ_1 minimization, the blue dotted lines correspond to IRWL1, the green dashed lines correspond to the ℓ_q minimization (circle: $q=\frac{1}{2}$; square: $q=\frac{2}{3}$), the black solid lines marked by cross correspond to IRLS, and the red solid lines correspond to the two-level ℓ_1 minimization (down triangle: $\rho=0$; up triangle: $\rho=\frac{1}{10}$; diamond: $\rho=\frac{1}{3}$).

From Fig. 5(a), one can see that the two-level ℓ_1 minimization indeed can enhance the sparsity from the result of the ℓ_1 minimization. We are also concerned about the average computational time, which is plotted in Fig. 5(b). The computational time of Algorithm 2 is not more than those of IRWL1 and the ℓ_q minimization, which coincides with our expectation. Generally, a small ρ corresponds to more computational time and a more sparse result. But the difference between different ρ is not significant and we suggest $\rho=1/10$ in practice. In this experiment, the iteratively reweighted least squares also give good recovery performance. Particularly, its computational time is much less than other methods. In fact, the iteration number for IRLS is much more than the reweighted ℓ_1 methods. In this experiment, we simply solve weighted ℓ_1 minimization by optimization toolbox in Matlab. There is much potential for speeding up ℓ_1 minimization. Hence, we cannot

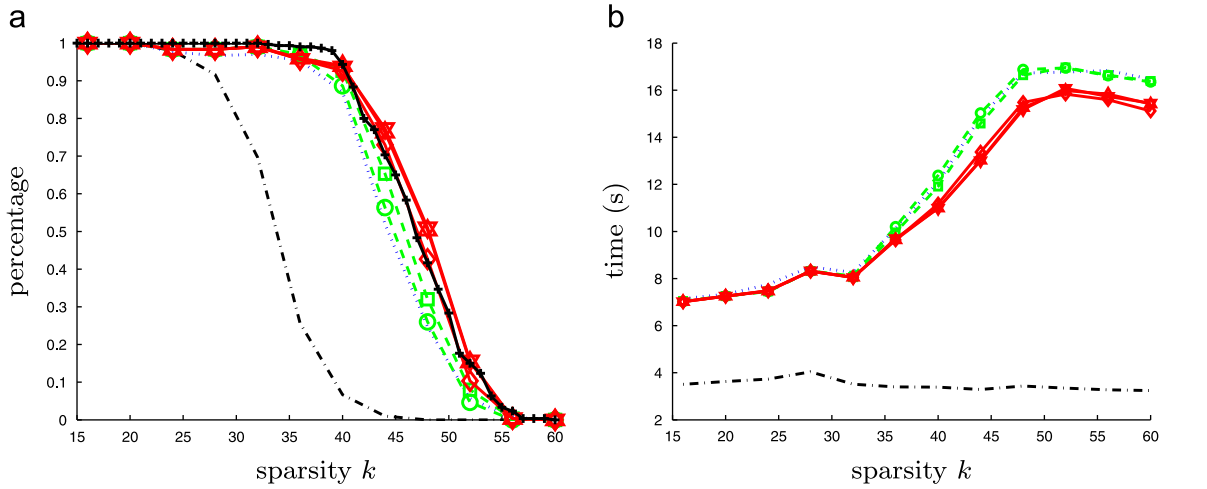


Fig. 5. Performance on signal recovery of the ℓ_1 minimization (black dotted-dashed line [1]), the iteratively reweighted ℓ_1 minimization (blue dotted line [16]), the iteratively reweighted least squares (black lines marked by cross [34]), the ℓ_q minimization (green dashed lines, circle: $q=\frac{1}{2}$; square: $q=\frac{2}{3}$ [33]) and the two-level ℓ_1 minimization (red solid lines, down triangle: $\rho=0$; up triangle: $\rho=\frac{1}{10}$; diamond: $\rho=\frac{1}{3}$): (a) recovery percentage; (b) average computational time (for the iteratively reweighted least squares, the computational time is always less than one second and omitted). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

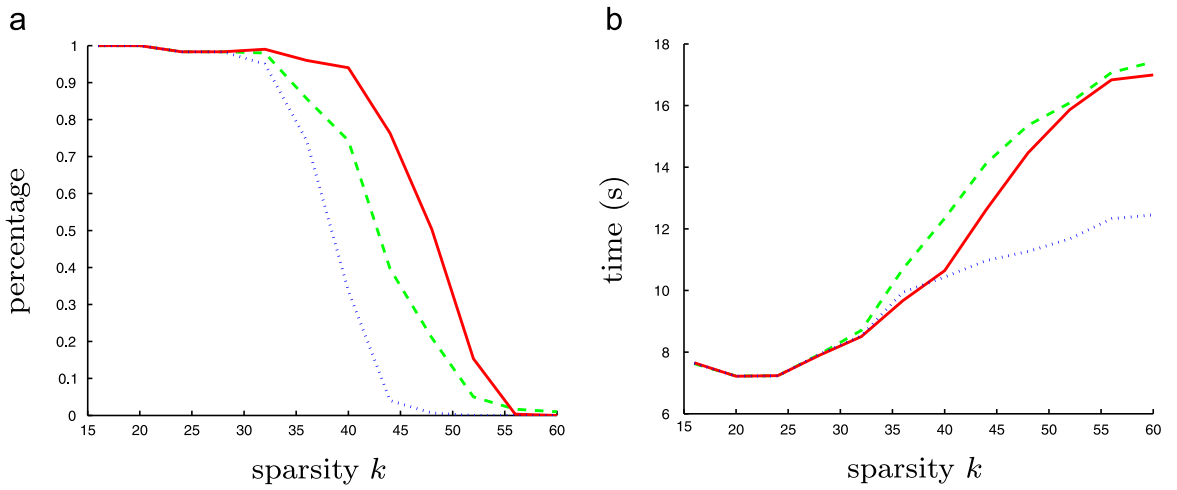


Fig. 6. Performance on signal recovery for K: $K=\lfloor\|x_0\|_{\ell_0}/2\rfloor$ (green dashed line), $K=\lfloor\|x_0\|_{\ell_0}/3\rfloor$ (red solid line), and $K=\lfloor\|x_0\|_{\ell_0}/10\rfloor$ (blue dotted line): (a) recovery percentage and (b) average computational time. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

arbitrarily conclude the comparison on computational time. The rigorous comparison can be found in [35]. This experiment also implies that we need fast algorithms for solving weighted ℓ_1 minimization in applications. For example, the proposed two-level soft thresholding method is a fast algorithm and its effectiveness can be observed in Section 4.3.

The results plotted in Fig. 5 are obtained using $K=\lfloor\|x_0\|_{\ell_0}/3\rfloor$, where x_0 is the result of the ℓ_1 minimization (5). We also consider other K values. When we want to recover a n-dimensional k-sparse signal, the ℓ_1 minimization is to deal with a problem of sparsity k/m. While the two-level ℓ_1 minimization is roughly for sparsity $(k-K)/(m-K)$. The components with small penalty weights will be kept non-zero with a high probability. Hence we need to find a good guess for the non-zero components, which is heuristically set according to x_0. If K is small, e.g., $K=\lfloor\|x_0\|_{\ell_0}/10\rfloor$, the improvement from ℓ_1 minimization is not significant. If K is large, some components selected from x_0 is not necessarily the real nonzero ones, which will affect the performance of the two-level ℓ_1-norm. Therefore, the performance is not monotone with the respective to K. Fig. 6 shows the recovery percentage and average computational time on 300 trials for several K values, from which $K=\lfloor\|x_0\|_{\ell_0}/3\rfloor$ is suggested.

4.2. Simulated error correction

One interesting variation of compressed sensing was designed in [11] for decoding, when a fraction of codewords are corrupted by noise. Again, we repeat the experiment in [16] for this application. In this experiment, a signal $x^0\in\mathbb{R}^n$ and a coding matrix $\Phi\in\mathbb{R}^{m\times n}$ with $m=512$, $n=128$ are randomly generated by choosing entries from the standard Gaussian distribution. We wish to decode x^0 from the received codeword $b=\Phi x^0+e$, where e is an unknown sparse corruption pattern. e being k-sparse means that k components of the codeword are corrupted by completely arbitrary values. In this experiment, we randomly select k components of b and let them be corrupted by sign flips. To decode x^0 from b, Ref. [11] solves the following ℓ_1 minimization problem:

$$\min_x \quad \|\Phi x-b\|_{\ell_1}. \tag{31}$$

If the ℓ_∞ distance between x^0 and the result of decoding is smaller than 10^{-3}, we claim that the error is corrected and the signal is recovered. The percentage of recovery using (31) is shown in Fig. 7(a) by the black dotted-dashed line. To improve the recovery performance, one can follow the idea of (7) and consider the following reweighted problem:

$$\min_x \quad \|W(\Phi x-b)\|_{\ell_1}, \tag{32}$$

where W is the diagonal matrix with diagonal entries $w_1, w_2, \ldots, w_m$. In [16], W is iteratively updated according to the components of the residual $\Phi x_{l-1}-b$ and the formulation is similar to (8). With some simple variable substitutions, we can use Algorithm 2 for error correction. For each k, we repeat the process above 300 times and calculate the recovery percentage for the ℓ_1 minimization, IRWL1, the ℓ_q minimization, the reweighted least squares, and the proposed two-level ℓ_1 minimization. In Fig. 7(a) and (b), we report the recovery percentage and the average computational time, respectively.

4.3. ECG signal recovery

In the above synthetic experiments, we have shown the performance of the two-level ℓ_1 minimization. Generally, though the two-level ℓ_1 minimization is non-convex and the global optimality cannot be guaranteed, the algorithms for the two-level ℓ_1 minimization give more sparse results in practice. Compared with IRWL1 and the ℓ_q minimization, the proposed method can give a better solution within comparable or less time. Moreover, we establish an iterative thresholding algorithm for the two-level

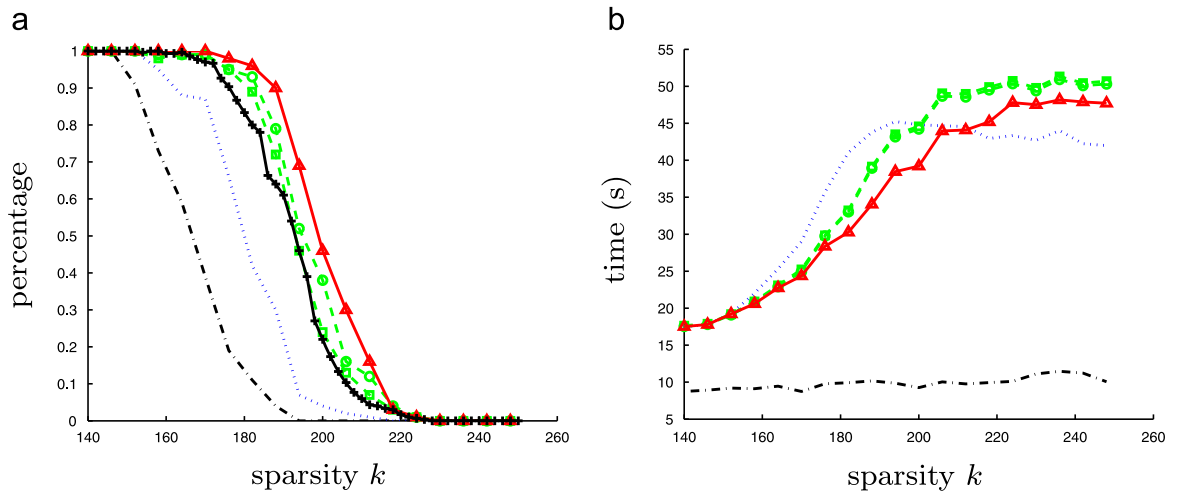


Fig. 7. Performance on error correction of the ℓ_1 minimization (black dotted-dashed line [1]), the iteratively reweighted ℓ_1 minimization (blue dotted line [16]), the iteratively reweighted least squares (black lines marked by cross [34]), the ℓ_q minimization (green dashed lines, circle: $q=\frac{1}{2}$; square: $q=\frac{2}{3}$ [33]), and the two-level ℓ_1 minimization with $\rho=\frac{1}{10}$ (red solid lines marked by up triangle): (a) recovery percentage and (b) average computational time (for the iteratively reweighted least squares, the computational time is always less than one second and omitted). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

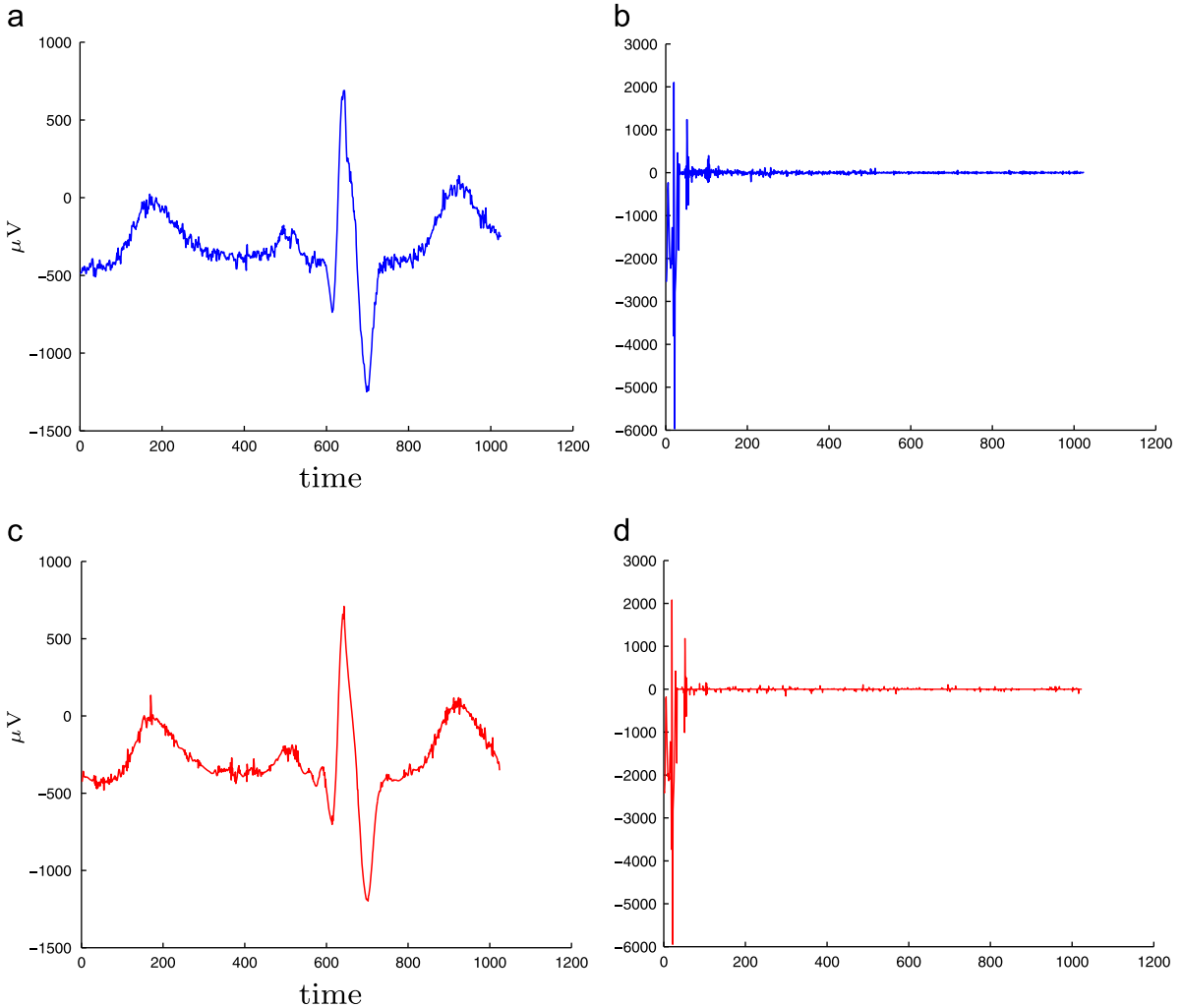


Fig. 8. The ECG signal in signal channel no. 1: (a) x^0; (b) θ^0; (c) $\tilde{x}$ recovered by the two-level soft thresholding ($m=300$); (d) $\tilde{\theta}$ recovered by the two-level soft thresholding ($m=300$).

Table 1
Recovery performance on ECG data sets.

No.	m	Soft		SCAD		Half		IRLS		Two-level	
		MSE	Time	MSE	Time	MSE	Time	MSE	Time	MSE	Time
1	64	8.174	0.474	9.281	0.881	7.927	5.113	10.31	0.871	7.993	1.014
	128	5.629	0.555	8.080	1.093	6.666	6.217	8.110	2.211	4.570	1.128
	256	2.107	0.452	4.720	1.278	3.244	6.887	4.339	3.192	1.393	0.612
	512	0.975	0.273	0.853	0.560	1.083	4.368	1.218	6.367	0.848	0.386
2	64	11.09	0.499	11.58	0.768	10.42	4.482	11.61	0.535	11.54	1.082
	128	8.174	0.598	10.47	1.013	8.904	5.158	8.873	1.155	6.622	1.184
	256	2.156	0.705	6.478	1.646	5.076	6.762	3.540	4.580	1.142	0.756
	512	0.747	0.405	0.606	1.164	0.829	5.897	1.164	15.29	0.608	0.510
3	64	11.51	0.601	12.66	0.899	11.66	4.515	11.80	0.952	11.46	1.222
	128	8.166	0.642	11.57	1.095	10.17	4.877	8.545	1.246	6.602	1.277
	256	2.645	0.720	4.243	1.444	5.337	3.488	4.560	3.576	1.414	0.924
	512	0.873	0.431	0.702	1.299	4.241	1.921	1.634	16.99	0.699	0.558
4	64	7.837	0.551	8.325	0.960	8.126	5.762	8.297	0.738	8.129	1.083
	128	5.929	0.538	7.358	1.029	7.668	8.853	6.573	1.335	5.579	1.035
	256	1.688	0.612	2.935	1.455	3.692	11.12	3.000	4.103	1.088	0.671
	512	0.761	0.339	0.644	0.598	2.563	12.17	1.084	16.24	0.649	0.431
5	64	8.230	0.555	9.370	0.930	8.110	5.262	8.725	0.691	8.204	1.113
	128	5.486	0.614	8.260	1.068	6.836	7.125	6.295	1.256	4.328	1.192
	256	2.006	0.516	4.263	1.519	4.085	4.776	3.494	4.473	1.276	0.680
	512	0.863	0.274	0.732	0.545	3.727	3.580	1.410	11.27	0.736	0.379
6	64	10.28	0.486	11.24	0.816	10.12	6.214	10.87	0.529	10.35	1.070
	128	7.252	0.422	8.266	0.852	7.681	7.345	7.964	1.255	5.407	0.983
	256	1.784	0.554	3.489	1.322	4.335	5.334	3.767	3.577	1.013	0.611
	512	0.647	0.298	0.520	0.759	2.879	4.912	1.026	13.53	0.522	0.382
7	64	10.67	0.585	13.74	0.663	11.89	7.154	11.47	0.571	10.47	1.102
	128	7.403	0.544	8.881	0.937	8.430	4.193	8.624	1.361	6.687	1.124
	256	2.327	0.613	4.757	1.218	5.540	2.900	4.294	3.496	1.052	0.865
	512	0.524	0.324	0.380	1.317	3.238	1.352	1.161	16.38	0.388	0.433
8	64	12.68	0.533	13.97	0.655	11.97	3.075	13.38	0.421	12.73	1.096
	128	9.291	0.575	9.051	0.946	10.45	3.057	10.04	1.225	8.095	1.118
	256	2.401	0.673	5.748	1.195	9.512	1.888	4.835	3.884	1.014	0.789
	512	0.529	0.312	0.379	1.207	4.212	1.345	1.038	15.27	0.389	0.395
9	64	16.34	0.549	18.32	0.518	17.44	4.751	16.29	0.469	15.68	1.108
	128	11.51	0.621	12.05	0.739	13.86	4.366	11.76	1.126	8.588	1.293
	256	3.581	0.751	7.463	1.119	8.587	3.655	6.227	3.394	1.505	0.993
	512	0.590	0.408	3.403	1.670	6.568	3.133	1.588	16.10	0.413	0.524

Table 1 (*continued*)

No.	m	Soft		SCAD		Half		IRLS		Two-level	
		MSE	Time	MSE	Time	MSE	Time	MSE	Time	MSE	Time
10	64	9.792	0.605	11.82	0.821	8.597	11.21	9.998	0.517	9.001	1.149
	128	6.261	0.457	5.016	0.835	7.352	10.77	7.302	1.248	5.025	0.944
	256	1.959	0.516	3.393	1.124	3.715	9.891	4.216	4.390	0.993	0.718
	512	0.522	0.272	0.394	0.834	2.889	6.928	1.023	14.92	0.339	0.367
11	64	6.210	0.469	7.137	0.891	7.074	10.44	6.694	0.685	6.313	0.943
	128	4.257	0.457	6.191	0.907	6.661	7.761	5.186	1.621	3.527	0.961
	256	1.209	0.408	0.974	1.330	2.902	11.57	2.552	4.037	0.733	0.478
	512	0.468	0.208	0.373	0.306	2.679	4.536	0.739	15.04	0.377	0.269
12	64	5.328	0.433	5.730	0.943	5.095	11.25	5.548	0.751	5.628	0.852
	128	3.584	0.435	4.415	0.901	4.020	7.566	3.988	1.616	2.778	0.848
	256	0.900	0.353	0.515	0.538	2.222	9.980	1.880	4.192	0.580	0.368
	512	0.416	0.163	0.339	0.206	1.874	10.07	0.538	12.51	0.335	0.214
13	64	4.511	0.489	5.637	0.801	5.857	10.85	5.068	0.575	4.422	0.935
	128	2.812	0.404	4.074	0.957	5.403	8.999	3.632	1.423	2.506	0.738
	256	0.690	0.328	0.337	0.662	2.675	14.12	1.867	4.524	0.421	0.409
	512	0.260	0.161	0.194	0.214	1.392	14.81	0.358	12.53	0.188	0.217
14	64	7.164	0.463	7.698	0.984	7.158	7.125	7.529	0.571	7.307	0.988
	128	4.858	0.368	6.353	0.786	6.228	8.188	5.552	1.234	3.513	0.859
	256	1.302	0.414	0.782	1.098	2.237	6.369	2.570	4.734	0.745	0.458
	512	0.486	0.216	0.380	0.316	1.987	6.347	0.696	15.07	0.378	0.271
15	64	5.287	0.470	6.188	0.845	5.842	5.811	5.662	0.615	5.379	0.859
	128	3.434	0.459	5.269	0.866	4.785	8.786	4.377	1.215	2.691	0.895
	256	0.892	0.379	0.457	0.764	4.033	13.23	2.287	4.425	0.523	0.445
	512	0.286	0.176	0.209	0.235	3.735	16.26	0.449	11.75	0.207	0.225

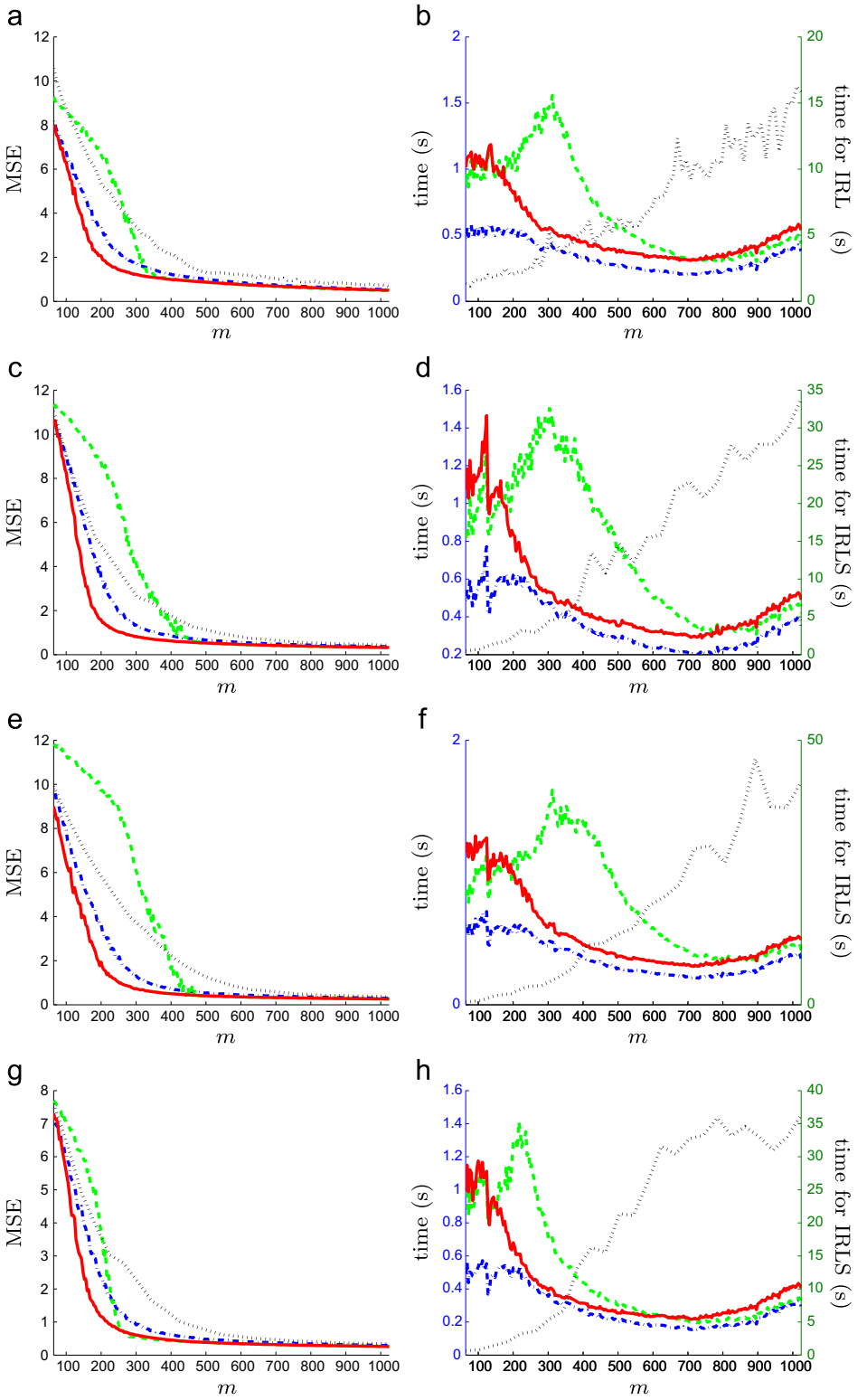


Fig. 9. MSE (figures in the left column) and computational time (figures in the left column) of the soft thresholding (blue dot-dashed lines [12]), SCAD thresholding (green dashed lines [22]), the two-level soft thresholding (red solid lines), and the iteratively reweighted least squares (black dotted lines). Notice that for clear illustration, the computational time for the iteratively reweighted least squares is for the right y-axis. (a) and (b) Channel no. 1; (c) and (d) channel no. 6; (e) and (f) channel no. 10; (g) and (h) channel no. 14. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

ℓ_1-norm, which is suitable for large-scale problems. In the following, we evaluate Algorithm 3 on real-life electrocardiography (ECG) data, and compare its performance with other algorithms, including the soft thresholding [31], the half thresholding [20], SCAD thresholding algorithm [22], and the iteratively reweighted least squares. In this experiment, we set $\tau=1$ for all the regularization problems, according to the noise level. The parameter setting for the half thresholding follows the scheme 1 in [20]. The parameter for SCAD thresholding is set by cross-validation.

The used ECG data come from the National Metrology Institute of Germany, which is online available in the PhysioNet [36,37]. The record includes 15 simultaneously measured signals sampled from one subject simultaneously. Each signal is digitized at 1000 samples per second, with 16 bit resolution over a range of ± 16.384 mV. On special request to the contributors of the database, recordings may be available at sampling rates up to 10 kHz. For each signal channel, there are 38,400 data points. We start from the first 1024 data and randomly generate one Gaussian matrix $\Phi \in \mathbb{R}^{m\times n}$, where $n=1024$ and m varies from 64 to 1024. Then we move to the next 1024 data and so on.

The ECG data itself is not sparse in the time domain as shown in Fig. 8(a), which plots x^0 for time $1\sim 1024$ of signal channel no.1. Following the suggestion in [38], we apply the orthogonal Daubechies wavelets (db 10), which is reported to be the most popular wavelet family for ECG compression, to design Ψ and get θ^0 such that $x^0=\Psi\theta^0$. Then, θ has sparsity as shown in Fig. 8(b). Here, we illustrate one example with $m=300$, i.e., we generate a matrix Φ with $m=300$ and compute $b=\Phi\Psi\theta^0$. Then Algorithm 3 is applied to recover θ. The resulted $\tilde{\theta}$ and the corresponding reconstructed signal $\tilde{x}=\Psi\tilde{\theta}$ are illustrated by Fig. 8(c) and (d), respectively.

As displayed by Fig. 8, the two-level soft thresholding can recover the sparse patten θ and also has de-noising effect. To measure the accuracy of recovery, we calculate the mean of squared error (MSE) between the recovered signal and the original signal. The MSEs for $m=64$, 128, 256, and 512 are reported in Table 1, where the computational times are given as well. From the result, one can see the effectiveness of the two-level soft thresholding algorithm: it uses a bit more time than the soft thresholding and achieves a better accuracy. The performance of SCAD is good when there are enough measurements, but it is not very suitable for low sparsity case. The half thresholding is established for the $\ell_{1/2}$ regularization. It gives a good result when m is small. But the computational time is significantly larger than the other methods, since we have to compute cosine and arccosine at each iteration. For IRLS, we need to calculate matrix inverse iteratively. Thus, when m grows, this method takes more computational time than the two-level soft thresholding algorithm.

We report the performance of the soft thresholding, SCAD thresholding, IRLS, and the two-level soft thresholding for several signal channels in Fig. 9, where the average MSE and average computing time for each m are plotted. By comparison, we find the good performance of Algorithm 3, especially when m is between 100 and 500.

5. Conclusion and further study

To enhance the sparsity of the ℓ_1-norm, we proposed the two-level ℓ_1-norm in this paper. $L^{\rho,K}_{2\text{level}}(x)$ can be minimized via an iteratively reweighted method. The major difference from the existing reweighted methods is that we focus on the order of the component values. Setting weights according to the order instead of the value corresponds to the piecewise linearity. From this property, we established a two-level soft thresholding algorithm and proved its convergence. We also developed a reweighted method. These schemes have appeared for other penalties, but the weight setting and the shrinkage operator are novel. Numerical experiments illustrated the good performance of the two-level ℓ_1 minimization.

For future work, the recovery analysis for the two-level ℓ_1 minimization algorithms is an important topic. The global search methods are also meaningful, which can enhance the sparsity further. Compared to ℓ_q-norm and $\|\cdot\|_{\text{IRWL1}}$, one advantage of the two-level ℓ_1-norm is that it can be written as a difference of two convex functions, which makes DC programming applicable. In [39–41], some acceleration techniques have been designed for the ℓ_1 minimization. These methods are also applicable for the two-level ℓ_1-norm, since it can be minimized via a series of weighted ℓ_1 problems. Extending the two-level ℓ_1-norm to multiple levels is also interesting. Once the weights for different levels can be effectively determined, introducing the flexibility can further enhance the sparsity.

Acknowledgments

The authors are grateful to the anonymous reviewers for helpful comments. The authors also thanks Dr. Ming Yan in Department of Mathematics, University of California, Los Angeles, for his important suggestion on Theorem 3.

This work was supported by (1) EU: the research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC AdG A-DATADRIVE-B (290923), BIOTENSORS (339804). This paper reflects only the authors' views, the Union is not liable for any use that may be made of the contained information. (2) Research Council KUL: GOA/10/09 MaNet, CoE PFV/10/002 (OPTEC), BIL12/11T; PhD/Postdoc Grants by (3) Flemish Government: (i) FWO: Projects: G.0377.12 (Structured systems), G.088114N (Tensor based data similarity), G.0108.11 (Compressed Sensing); PhD/Postdoc grants by (ii) IWT: Projects: SBO POM (100031); PhD/Postdoc Grants by (iii) iMinds Medical Information Technologies SBO 2014. (4) Belgian Federal Science Policy Office: IUAP P7/19 (DYSCO, Dynamical systems, control and optimization, 2012–2017). L. Shi is also supported by the National Natural Science Foundation of China (11201079) and the Fundamental Research Funds for the Central Universities of China (20520133238 and 20520131169). Johan Suykens is a professor at KU Leuven, Belgium.

References

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

[2] E.J. Candès, J.K. Romberg, T. Tao, Stable signal recovery from incomplete and inaccurate measurements, Commun. Pure Appl. Math. 59 (2006) 1207–1223.

[3] Y.C. Eldar, G. Kutyniok, Compressed Sensing: Theory and Applications, Cambridge University Press, Cambridge, UK, 2012.

[4] T. Blumensath, M. Davies, Iterative thresholding for sparse approximations, J. Fourier Anal. Appl. 14 (2008) 629–654.

[5] T. Blumensath, M. Davies, Iterative hard thresholding for compressed sensing, Appl. Comput. Harmon. Anal. 27 (2009) 265–274.

[6] J.A. Tropp, A.C. Gilbert, Signal recovery from random measurements via orthogonal matching pursuit, IEEE Trans. Inf. Theory 53 (12) (2007) 4655–4666.

[7] D. Needell, J.A. Tropp, Cosamp: iterative signal recovery from incomplete and inaccurate samples, Appl. Comput. Harmon. Anal. 26 (2009) 301–321.

[8] W. Dai, O. Milenkovic, Subspace pursuit for compressive sensing signal reconstruction, IEEE Trans. Inf. Theory 55 (2008) 2230–2249.

[9] S.S. Chen, D.L. Donoho, M.A. Saunders, Atomic decomposition by basis pursuit, SIAM J. Sci. Comput. 20 (1998) 33–61.

[10] E.J. Candès, T. Tao, The Dantzig selector: statistical estimation when p is much larger than n, Ann. Stat. 35 (2007) 2313–2351.

[11] E.J. Candès, T. Tao, Decoding by linear programming, IEEE Trans. Inf. Theory 51 (2005) 4203–4215.

[12] I. Daubechies, M. Defrise, C. De Mol, An iterative thresholding algorithm for linear inverse problems with a sparsity constraint, Commun. Pure Appl. Math. 57 (2004) 1413–1457.

[13] A. Cohen, W. Dahmen, R. DeVore, Compressed sensing and best k-term approximation, J. Am. Math. Soc. 22 (2009) 211–231.

[14] J.A. Tropp, Greed is good: algorithmic results for sparse approximation, IEEE Trans. Inf. Theory 50 (2004) 2231–2242.

[15] D.L. Donoho, I. Johnstone, A. Maleki, A. Montanari, Compressed Sensing over ℓ_p-balls: minimax mean square error, in: IEEE International Symposium on Information Theory Proceedings, 2011, pp. 129–133.

[16] E.J. Candès, M.B. Wakin, S.P. Boyd, Enhancing sparsity by reweighted ℓ_1 minimization, J. Fourier Anal. Appl. 14 (2008) 877–905.

[17] R. Chartrand, Exact reconstruction of sparse signals via nonconvex minimization, IEEE Signal Process. Lett. 14 (2007) 707–710.

[18] X. Chen, F. Xu, Y. Ye, Lower bound theory of nonzero entries in solutions of ℓ_2–ℓ_p minimization, SIAM J. Sci. Comput. 32 (2010) 2832–2852.

[19] M.J. Lai, J. Wang, An unconstrained ℓ_q minimization with $0 < q \leq 1$ for sparse solution of underdetermined linear systems, SIAM J. Optim. 21 (2011) 82–101.

[20] Z. Xu, X. Chang, F. Xu, H. Zhang, $\ell_{1/2}$ regularization: a thresholding representation theory and a fast solver, IEEE Trans. Neural Netw. Learn. Syst. 23 (2012) 1013–1027.

[21] L. Shi, X. Huang, J.A.K. Suykens, Sparse kernel regression with coefficient-based ℓ_q-regularization, Internal Report 13-50, ESAT-SISTA, KU Leuven, submitted for publication.

[22] J. Fan, R. Li, Variable selection via nonconcave penalized likelihood and its oracle properties, J. Am. Stat. Assoc. 96 (2001) 1348–1360.

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

[24] R. Horst, N. Thoai, DC programming: overview, J. Optim. Theory Appl. 103 (1999) 1–43.

[25] L. An, P. Tao, The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems, Ann. Oper. Res. 133 (2005) 23–46.

[26] M. Bogdan, E. van den Berg, W. Su, E. J. Candès, Statistical Estimation and Testing via the Ordered l_1 norm, http://arxivpreprintarXiv:1310.1969 arxiv preprint arXiv:1310.1969, 2013.

[27] Y. Wang, W. Yin, Sparse signal reconstruction via iterative support detection, SIAM J. Imaging Sci. 3 (2010) 462–491.

[28] X. Huang, J. Xu, S. Wang, Exact penalty and optimality condition for nonseparable continuous piecewise linear programming, J. Optim. Theory Appl. 155 (2012) 145–164.

[29] X. Chen, W. Zhou, Convergence of the reweighted ℓ_1 minimization algorithm for ℓ_2–ℓ_p minimization, Comput. Optim. Appl. 59 (2014) 47–61.

[30] I. Daubechies, R. DeVore, M. Fornasier, C.S. Güntürk, Iteratively reweighted least squares minimization for sparse recovery, Commun. Pure Appl. Math. 63 (2010) 1–38.

[31] I. Daubechies, R. DeVore, M. Fornasier, C. S. Güntürk, Iteratively reweighted least squares minimization: proof of faster than linear rate for sparse recovery, in: The 42nd IEEE Annual Conference on Information Sciences and Systems, 2008, pp. 26–29.

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

[33] S. Foucart, M. Lai, Sparsest solutions of underdetermined linear system by ℓ_q minimization for $0 < q \leq 1$, Appl. Comput. Harmon. Anal. 26 (2009) 395–407.

[34] R. Chartrand, W. Yin, Iteratively reweighted algorithms for compressive sensing, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, 2008, pp. 3869–3872.

[35] D. Wipf, S. Nagarajan, Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions, IEEE J. Select. Top. Signal Process. 4 (2010) 317–329.

[36] A.L. Goldberger, L.A. Amaral, L. Glass, J.M. Hausdorff, P.C. Ivanov, R.G. Mark, J.E. Mietus, G.B. Moody, C.-K. Peng, H.E. Stanley, Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals, Circulation 101 (2000) e215.

[37] G.B. Moody, R.G. Mark, A.L. Goldberger, PhysioNet: a web-based resource for the study of physiologic signals, IEEE Eng. Med. Biol. Mag. 20 (2001) 70–75.

[38] P.S. Addison, Wavelet transforms and the ECG: a review, Physiol. Meas. 26 (2005) R155.

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

[40] M. Guerquin-Kern, M. Häberlin, K. Pruessmann, M. Unser, A fast wavelet-based reconstruction method for magnetic resonance imaging, IEEE Trans. Med. Imaging 30 (2011) 1649–1660.

[41] D.L. Donoho, A. Maleki, A. Montanari, Message-passing algorithms for compressed sensing, Proc. Natl. Acad. Sci. 106 (2009) 18914–18919.