Journal of Scientific Computing (2021) 87:21
https://doi.org/10.1007/s10915-021-01440-z

Smoothing Strategy Along with Conjugate Gradient Algorithm for Signal Reconstruction

Caiying Wu[1] · Jing Wang[1] · Jan Harold Alcantara[2,3] · Jein-Shan Chen[2]

Received: 20 January 2020 / Revised: 1 February 2021 / Accepted: 17 February 2021

Abstract
In this paper, we propose a new smoothing strategy along with conjugate gradient algorithm for the signal reconstruction problem. Theoretically, the proposed conjugate gradient algorithm along with the smoothing functions for the absolute value function is shown to possess some nice properties which guarantee global convergence. Numerical experiments and comparisons suggest that the proposed algorithm is an efficient approach for sparse recovery. Moreover, we demonstrate that the approach has some advantages over some existing solvers for the signal reconstruction problem.

Keywords l_p-norm regularization · Signal recovery · Conjugate gradient algorithm · Sparse solution

Mathematics Subject Classification 90C33

1 Introduction

The target problem of this paper is the signal reconstruction problem, which has wide applications in compressive sensing [6,7,9,16]. Tremendous amount of articles related to this topic can be found in the literature, and hence we do not intend to repeat its importance and various applications here. We shall directly look into its mathematical model and show our idea for tackling it. Mathematically, the signal reconstruction problem model is described as follows:

$$\begin{aligned} &\min \ \|x\|_0 \\ &\text{s.t.} \ \ b = Ax, \end{aligned} \tag{1}$$

Caiying Wu, Jing Wang, research is supported by the Natural Science Foundation of Inner Mongolia Autonomous Region (2018MS01016). Jan Harold Alcantara, Jein-Shan Chen, research is supported by Ministry of Science and Technology, Taiwan.

✉ Jein-Shan Chen
jschen@math.ntnu.edu.tw

1 College of Mathematics Science, Inner Mongolia University, Hohhot 010021, China

2 Department of Mathematics, National Taiwan Normal University, Taipei 11677, Taiwan

3 Mathematics and Statistics Department, De La Salle University, Manila 1004, Philippines

Published online: 02 March 2021

where $x \in \mathbb{R}^n$ is the original sparse signal that needs to be recovered, $A \in \mathbb{R}^{m\times n}$ ($m \ll n$) is the measurement matrix, $b \in \mathbb{R}^m$ is the observed vector, and $\|x\|_0$ represents the l_0-norm of x, which is defined as the number of nonzero components of x. Note that b could also be contaminated with noise, that is, $b = Ax + e$, where $e \in \mathbb{R}^m$ denotes the noise.

Unfortunately, the l_0-norm minimization problem (1) is an NP-hard combinatorial optimization problem [24]. In order to avoid this difficulty, one popular approach is to replace l_0-norm by l_1-norm in model (1) and obtain the following l_1-minimization problem:

$$\begin{aligned} &\min \ \|x\|_1 \\ &\text{s.t.} \ \ b = Ax, \end{aligned} \tag{2}$$

where $\|x\|_1$ denotes the l_1-norm of x and $\|x\|_1 = \sum_{i=1}^n |x_i|$. More specifically, under the Restricted Isometry Property (RIP), the l_1-minimization (2) was shown to possess the same solution as l_0-minimization (1). Please refer to [1,4,5,7,8,10,11,16] for more details and survey.

Related to (2) are two regularized unconstrained minimization problems. The first one is given by

$$\text{Model (I)}: \quad \min_{x\in\mathbb{R}^n} \lambda\|x\|_1 + \frac{1}{2}\|Ax - b\|_2^2.$$

For instance, the studies in [7,16,18,21] follow model (I). The second one is

$$\text{Model (II)}: \quad \min_{x\in\mathbb{R}^n} \lambda_1\|x\|_1 + \lambda_2\|x\|_2^2 + \frac{1}{2}\|Ax - b\|_2^2,$$

which is the subject of investigation done in [35].

An alternative approach to compute the sparse solutions of l_0-minimization (1) is proposed by Gribnoval and Nielsen in [19], which is given by the constrained model

$$\begin{aligned} &\min \ \|x\|_p^p \\ &\text{s.t.} \ \ Ax = b, \end{aligned} \tag{3}$$

where $0 < p < 1$ and $\|x\|_p^p = \sum_{j=1}^n |x_j|^p$. The problem (3) is called l_p-minimization, which is motivated by the special property of l_p-quasi-norm:

$$\lim_{p\to 0^+} \|x\|_p^p = \|x\|_0.$$

Similarly, the l_p-minimization problem (3) is NP-hard [12]. In order to deal with this problem, three more regularized unconstrained minimization models are studied in the literature. The first one is

$$\text{Model (III)}: \quad \min_{x\in\mathbb{R}^n} \lambda\|x\|_p^p + \frac{1}{2}\|Ax - b\|_2^2, \tag{4}$$

where $\lambda > 0$ is a parametric factor. The studies in [14,17,22,26,33] are based on the model (III). The second one is

$$\text{Model (IV)}: \quad \min_{x\in\mathbb{R}^n} \lambda_1\|x\|_p^p + \lambda_2\|x\|_2^2 + \frac{1}{2}\|Ax - b\|_2^2,$$

where $\lambda_1, \lambda_2 > 0$ are two parameters. The model (IV) is studied in [34]. The third model based on (3) is

$$\text{Model (V)}: \quad \min_{x\in\mathbb{R}^n} \lambda_1\|x\|_p^p + \lambda_2\|x\|_1 + \frac{1}{2}\|Ax - b\|_2^2,$$

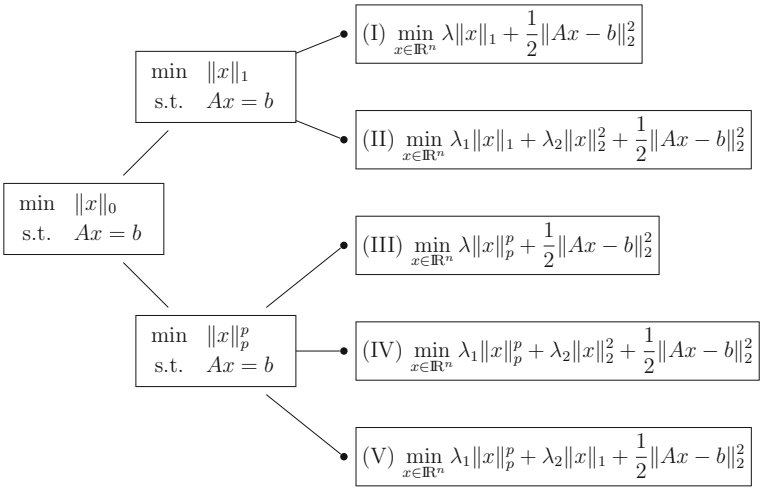

Fig. 1 Five models for sparse reconstruction

where $\lambda_1, \lambda_2 > 0$ are two parameters. The model (V) is recently investigated in [30]. To sum up, we present in Fig. 1 all the five models (I)–(V) together.

There are many optimization algorithms that can be applied to solve the above models (I)–(V). For the sake of simplicity and lower storage requirements, conjugate gradient algorithms are suitable for large scale problems. In this paper, like [27,29,32,36], we are interested in applying the conjugate gradient method for signal recovery. To employ the conjugate gradient algorithm, we need to check the differentiability of the objective function that we aim to minimize. However, we observe that there is a common feature among all the aforementioned models (I)–(V). Not only they are regularized unconstrained minimizations, but also all the objective functions are nonsmooth. The key part causing this is the non-differentiability of the absolute value function $|\cdot|$ inside l_1-norm and l_p-norm. In view of this feature, we shall consider smoothing strategy to approximate the absolute value function $|\cdot|$. Albeit the idea is not new, we propose new smoothing function and compare different kinds of smoothing approaches along with conjugate gradient algorithm, which is the main contribution of this paper.

Recently, there are six smoothing functions studied in [25] to approximate the absolute value function $|\cdot|$. Inspired by this article, we adapt them to work along with our proposed conjugate gradient which will be recalled in the next section. First, we present them out as below.

$$\varphi_1(\mu, t) = \mu[\ln(1 + e^{-\frac{t}{\mu}}) + \ln(1 + e^{\frac{t}{\mu}})],$$

$$\varphi_2(\mu, t) = \begin{cases} t & \text{if } t \geq \frac{\mu}{2}, \\ \frac{t^2}{\mu} + \frac{\mu}{4} & \text{if } -\frac{\mu}{2} < t < \frac{\mu}{2}, \\ -t & \text{if } t \leq -\frac{\mu}{2}, \end{cases}$$

$$\varphi_3(\mu, t) = \sqrt{4\mu^2 + t^2},$$

$$\varphi_4(\mu, t) = \begin{cases} \frac{t^2}{2\mu} & \text{if } |t| \leq \mu, \\ |t| - \frac{\mu}{2} & \text{if } |t| > \mu, \end{cases}$$

$$\varphi_5(\mu,t)=\begin{cases} t & \text{if } t>\mu,\\ -\frac{t^4}{8\mu^3}+\frac{3t^2}{4\mu}+\frac{3\mu}{8} & \text{if } -\mu\le t\le\mu,\\ -t & \text{if } t<-\mu,\end{cases}$$

$$\varphi_6(\mu,t)=t\,\mathrm{erf}\left(\frac{t}{\sqrt{2}\mu}\right)+\sqrt{\frac{2}{\pi}}\mu e^{-\frac{t^2}{2\mu^2}}.$$

where the error function is defined as follows:

$$\mathrm{erf}(t)=\frac{2}{\sqrt{\pi}}\int_0^t e^{-u^2}du,\quad \forall t\in\mathbb{R}.$$

With these smoothing functions, there are several smooth models that can be considered, which are given as below.

$$\text{[SF(I)]}\quad \min_{x\in\mathbb{R}^n}\lambda\sum_{i=1}^n\varphi_j(\mu,x_i)+\frac{1}{2}\|Ax-b\|_2^2,$$

$$\text{[SF(II)]}\quad \min_{x\in\mathbb{R}^n}\lambda\sum_{i=1}^n\varphi_j(\mu,x_i)+\lambda_2\|x\|_2^2+\frac{1}{2}\|Ax-b\|_2^2,$$

$$\text{[SF(III)]}\quad \min_{x\in\mathbb{R}^n}\lambda\sum_{i=1}^n\varphi_j^p(\mu,x_i)+\frac{1}{2}\|Ax-b\|_2^2,$$

$$\text{[SF(IV)]}\quad \min_{x\in\mathbb{R}^n}\lambda_1\sum_{i=1}^n\varphi_j^p(\mu,x_i)+\lambda_2\|x\|_2^2+\frac{1}{2}\|Ax-b\|_2^2,$$

$$\text{[SF(V)]}\quad \min_{x\in\mathbb{R}^n}\lambda_1\sum_{i=1}^n\varphi_j^p(\mu,x_i)+\lambda_2\sum_{i=1}^n\varphi_j(\mu,x_i)+\frac{1}{2}\|Ax-b\|_2^2,$$

Here, $\varphi_j(\mu,x_i)\approx|x_i|$ for $j=1,2,\ldots,6$. Moreover, "[SF(I)]" stands for smoothing function for model (I). Likewise, the others represent similar meanings.

Now, we are ready to present our model. Theoretically, we only focus on model (III) given as in (4). In particular, our main idea is inspired by the re-weighted techniques used in [22,23,34]. To proceed, notice that we can express the objective function in model (III) as

$$\begin{aligned}\lambda\|x\|_p^p+\frac{1}{2}\|Ax-b\|_2^2&=\lambda\sum_{i=1}^n|x_i|^p+\frac{1}{2}\|Ax-b\|_2^2\\&=\lambda\sum_{i=1}^n|x_i|^{p-1}\cdot|x_i|+\frac{1}{2}\|Ax-b\|_2^2.\end{aligned}$$

In light of this expression, for $j=1,2,\ldots,6$, we construct the below smoothing function

$$H_j(x):=\lambda\sum_{i=1}^n({x_i}^2+\overline{\mu}^2)^{\frac{p-1}{2}}\cdot\varphi_j(\mu,x_i)+\frac{1}{2}\|Ax-b\|_2^2,\tag{5}$$

where $\overline{\mu}>0$ and $\mu>0$ are two parameters. Indeed, the weight $({x_i}^2+\overline{\mu}^2)^{\frac{p-1}{2}}$ for $i=1,2,\ldots,n$ in (5) is regarded as a continuous variable, which is slightly different from that used in the literature. Then, we have a new smoothing strategy for model (III):

$$\text{[ReSF(III)]}\quad \min_{x\in\mathbb{R}^n}\lambda\sum_{i=1}^n({x_i}^2+\overline{\mu}^2)^{\frac{p-1}{2}}\cdot\varphi_j(\mu,x_i)+\frac{1}{2}\|Ax-b\|_2^2.\tag{6}$$

"[ReSF(III)]" stands for re-weighted smoothing function for model (III). We apply a version of the conjugate gradient algorithm presented in the next section to the smooth model (6).

The remaining parts of this paper are organized as follows. In Sect. 2, we shall introduce a version of nonlinear conjugate gradient method to solve the model (III) by using (5) and (6). We prove the boundedness of the level sets of the objective function, as well as the Lipschitz continuity of its gradient. On the basis of these properties, we provide a theorem that shows the global convergence of the algorithm. In Sect. 3, we report some experimental results to demonstrate the efficiency of our proposed method. Numerical comparisons with other popular algorithms are shown as well, which reveal that our new model has a satisfactory numerical performance, which strongly suggests that it is a good choice for the target problem. We discuss our conclusions in Sect. 4.

2 Algorithm and Convergence Analysis

In this section, we first describe the nonlinear conjugate gradient (CG) algorithm, which was proposed by Chen and Zhou and for image restoration in [13]. Here, we employ it for signal recovery. Then, we discuss the boundedness property of the level set, the Lipschitz continuity property of the objective function's gradient and the global convergence. There are many versions of CG algorithms, and this one can be viewed as a version of the CG combined with limited memory BFGS method.

Algorithm 1

Step 0. Choose initial point $x^0 \in \mathbb{R}^n$, $\varepsilon_0 > 0$, $\overline{\mu} > 0$, $\mu > 0$, $r \geq 0$, $\delta \in (0, 1)$ and $\rho \in (0, 1)$. Set $k = 0$.

Step 1. Compute the search direction

$$d^k = \begin{cases} -\nabla H_j(x^k) & \text{if } k = 0, \\ -\nabla H_j(x^k) + \beta_k d^{k-1} + \theta_k z^{k-1} & \text{if } k > 0, \end{cases}$$

where

$$\begin{aligned} \beta_k &= \frac{\nabla^T H_j(x^k) z^{k-1}}{(d^{k-1})^T z^{k-1}} - \frac{2\|z^{k-1}\|^2 \nabla^T H_j(x^k) d^{k-1}}{((d^{k-1})^T z^{k-1})^2}, \\ \theta_k &= \frac{\nabla^T H_j(x^k) d^{k-1}}{(d^{k-1})^T z^{k-1}}, \\ z^{k-1} &= y^{k-1} + t_k s^{k-1}, \\ t_k &= \varepsilon_0 \left\| \nabla H_j(x^k) \right\|^r + \max\left\{0, -\frac{(s^{k-1})^T y^{k-1}}{\|s^{k-1}\|^2}\right\}, \end{aligned}$$

with

$$\begin{aligned} y^{k-1} &= \nabla H_j(x^k) - \nabla H_j(x^{k-1}), \\ s^{k-1} &= x^k - x^{k-1} = \alpha_{k-1} d^{k-1}. \end{aligned}$$

Step 2. Compute $\alpha_k = \max\{\rho^0, \rho^1, \ldots\}$ satisfying

$$H_j(x^k + \rho^i d^k) \leq H_j(x^k) + \delta \rho^i \left(\nabla H_j(x^k)\right)^T d^k.$$

Step 3. Set $x^{k+1} = x^k + \alpha_k d^k$.
Step 4. Set $k = k + 1$ and go to Step 1.

Lemma 2.1 *Let H_j be defined as in (5). For any $\overline{\mu} > 0$ and $\mu > 0$, the level set $\mathcal{L}(x^0) = \{x \in \mathbb{R}^n \mid H_j(x) \leq H_j(x^0)\}$ is bounded.*

Proof From the structure of the function $\varphi_j(\mu, t)$, it is not hard to verify that

$$\lim_{t\to\infty} (t^2 + \bar{\mu}^2)^{\frac{p-1}{2}} \varphi_j(\mu, t) = +\infty.$$

Hence, for any $\mu > 0$, we obtain

$$\lim_{\|x\|\to\infty} H_j(x) = +\infty. \tag{7}$$

Assume that $\mathcal{L}(x^0)$ is unbounded. Then, there exists an index set K_1, such that $\|x^k\| \to \infty$, as $k \to \infty$ and $k \in K_1$. By applying (7), it yields

$$H_j(x^k) \to +\infty, \quad \text{as } k \to \infty \text{ and } k \in K_1,$$

which contradicts $H_j(x^k) \leq H_j(x^0)$. Thus, the level set $\mathcal{L}(x^0)$ is bounded. □

Noting that the function $(t^2+\bar{\mu}^2)^{\frac{p-1}{2}}$ is continuously differentiable of arbitrary order while the smoothing functions $\varphi_j(\mu, t)$ have Lipschitz continuous gradient, then we conclude that the first term of H_j given by (5) has Lipschitz continuous gradient over bounded sets. In particular, since the level set $\mathcal{L}(x^0)$ is bounded by Lemma 2.1, it follows that the first term of H_j has Lipschitz continuous gradient over $\mathcal{L}(x^0)$. Since the gradient of $\frac{1}{2}\|Ax - b\|_2^2$ is also Lipschitz continuous over $\mathbb{R}^n$, then we get the following result.

Lemma 2.2 *The function H_j defined in (5) has Lipschitz continuous gradient, that is, there exists a constant $L > 0$, such that*

$$\left\|\nabla H_j(x) - \nabla H_j(y)\right\| \leq L\|x - y\|, \quad \forall\, x, y \in \mathcal{L}(x^0). \tag{8}$$

Lemma 2.3 *Let H_j be defined as in (5) and $\{x^k\}$, $\{d^k\}$ be generated by Algorithm 1. Then, there holds*

$$\left(\nabla H_j(x^k)\right)^T d^k \leq -\frac{1}{2}\|\nabla H_j(x^k)\|^2. \tag{9}$$

Proof With Lemma 2.1 and Lemma 2.2, the proof is exactly the same as that in [13, Lemma 2.2]. We omit it. □

Theorem 2.1 *Let H_j be defined as in (5) and $\{x^k\}$, $\{d^k\}$ be generated by Algorithm 1. Then, we have*

$$\liminf_{k\to\infty} \left\|\nabla H_j(x^k)\right\| = 0, \tag{10}$$

for $j = 1, 2, \ldots, 6$.

Proof Suppose that the conclusion (10) is not true. Then, there exists a constant $r_1 > 0$ such that

$$\left\|\nabla H_j(x^k)\right\| \geq r_1, \quad \forall k,\ j = 1, 2, \ldots, 6.$$

Applying Lemma 2.3, it implies that $\{H_j(x^k)\}$ is decreasing, which implies $x^k \in \mathcal{L}(x^0)$. From Lemma 2.1, the level set $\mathcal{L}(x^0)$ is bounded. Thus, $\{H_j(x^k)\}$ is convergent. In addition, from the Armijo line search in Step 2, we know

$$\lim_{k\to\infty} \alpha_k \left(\nabla H_j(x^k)\right)^T d^k = 0. \tag{11}$$

On the other hand, applying (9) in Lemma 2.3 gives

$$\alpha_k \left(\nabla H_j(x^k)\right)^T d^k \leq -\frac{1}{2}\alpha_k \|\nabla H_j(x^k)\|^2 \leq -\frac{r_1^2}{2}\alpha_k < 0. \tag{12}$$

Putting (11) and (12) together yields

$$\lim_{k\to\infty} \alpha_k = 0.$$

Now, by [13, Lemma 2.3], there exists a constant $\overline{\varepsilon} > 0$ such that

$$\|d^k\| \leq \overline{\varepsilon}\|\nabla H_j(x^k)\|, \quad \forall k \geq 0.$$

Since $\mathcal{L}(x^0)$ is bounded, there is a constant $\overline{r} > 0$ such that

$$\|\nabla H_j(x^k)\| \leq \overline{r}, \quad \forall k \geq 0.$$

Thus, there holds $\|d^k\| \leq \overline{\varepsilon}\overline{r}$, and hence

$$\lim_{k\to\infty} \alpha_k d^k = 0.$$

Note that from the Armijo line search, we have

$$\frac{H_j(x^k) - H_j(x^k + \frac{\alpha_k}{\rho} d^k)}{\alpha_k/\rho} < -\delta \left(\nabla H_j(x^k)\right)^T d^k. \tag{13}$$

Since $\{x^k\}$ and $\{d^k\}$ are bounded, there exist subsequences $\{x^k\}_{k\in K}$ and $\{d^k\}_{k\in K}$ such that

$$x^k \to \overline{x} \quad \text{and} \quad d^k \to \overline{d}, \quad \text{as} \quad k \in K,\ k \to \infty.$$

Then, taking the limit on both sides of (13) with $k \in K$ leads to

$$-\left(\nabla H_j(\overline{x})\right)^T \overline{d} \leq -\delta \left(\nabla H_j(\overline{x})\right)^T \overline{d} \quad \Longrightarrow \quad \left(\nabla H_j(\overline{x})\right)^T \overline{d} \geq 0,$$

which contradicts Lemma 2.3. Therefore, the proof is complete. □

3 Numerical Experiments

In this section, we report the results of our experiments demonstrating the applicability, efficiency and merits of our algorithm. All tests are carried out on a PC (3.20GHz, 32GB of RAM) with the use of Matlab R2020a.

The parameters of our algorithm are summarized as follows:

$$x^0 = A^T b, \quad \lambda = \lambda_0 = 0.001\|A^T b\|_\infty, \quad \rho = 0.2,$$
$$\delta = 0.3, \quad \varepsilon_0 = 10^{-10}, \quad r = 0, \quad \overline{\mu} = 10^{-3}.$$

For the smoothing parameter μ, we apply a continuation approach which has been widely used in the literature (see [3] for instance) in order to improve the quality of solution obtained. First, we set an initial value of the smoothing parameter as $\mu_0 = 10^{-3}$ and use Algorithm

1 to solve (6). When running Algorithm 1, we used the stopping criterion implemented in NESTA [3] which is based on the relative variation of the objective function. In particular, we stop Algorithm 1 when

$$\frac{|H_j(x^{k+1}) - H_j(x^k)|}{|H_j(x^{k+1})|} < 10^{-5}, \quad j = 1, 2, 3, 4, 5, 6. \tag{14}$$

For higher accuracy of the solution, a lower threshold value for the relative variation can be used. Since we are applying a continuation scheme, we reduce the value of μ then implement again Algorithm 1 as above with stopping criterion given by (14). In particular, for $k \geq 0$, we set $\mu_{k+1} = 0.1\mu_k$. In addition to this, we also reduce the regularization parameter λ in each continuation step according to the formula $\lambda_{k+1} = 0.1\lambda_k$ for $k \geq 0$. This procedure is inspired by the approach used in fixed-point continuation (FPC) method [20]. We repeat the continuation procedure until we obtain a solution within a desired relative error, or until we have reached a pre-specified maximum number of continuation steps. In our simulations, we set the maximum number of continuation steps to 4 which we found to be enough to obtain a good approximate solution for the problems we have considered.

3.1 Comparison of Smoothing Functions and Choosing *p*

Meanwhile, the success and performance of our algorithm are indeed dependent on the parameter p and the choice of smoothing function. Thus, we explore these issues first before we present our comparisons with other algorithms.

Experiment 1 In this experiment, we determine which values of p will result to an algorithm which has a good frequency of success and fast convergence time. We let $A \in \mathbb{R}^{m\times n}$ be a Gaussian matrix with $n = 2^{15} = 32768$, $m = \frac{n}{2}$, $x^* \in \mathbb{R}^n$ is a $K-$sparse original signal with $K = \lfloor \frac{n}{40} \rfloor$, whose nonzero components are sampled from $\mathcal{N}(0, 1)$. We consider two cases where $b = Ax^*$ (noiseless case) and $b = Ax^* + e$ where e is a Gaussian noise with zero mean and $\sigma = 0.01$.

For each test problem, the result is reported by running the algorithm 10 times and taking the average CPU time. Note that just like the numerical experiment in [29], the smoothing function φ_1 is very unstable, so we do not include it in the numerical results. In the sequel, the frequency of successful reconstructions means the percentage of all test instances that reach the required relative error which we set to 10^{-3}, i.e.

$$\frac{\|x - x^*\|}{\|x^*\|} \leq 10^{-3}$$

where x is the solution obtained by the algorithm. The summary of the results are presented in Tables 1 and 2 .

The summary of results for the noiseless and noisy cases are presented in Tables 1 and 2, respectively. From these, we see that the algorithm has a very low frequency of success when a small value of p is used, particularly when $p = 0.3$. The frequency of success improves as the value of p increases. In terms of CPU time of convergence, we observe the same pattern that a larger value of p provides faster convergence. In particular, we obtain the fastest convergence time when $p = 0.9$. We can also see that the smoothing function φ_3 with $p = 0.9$ has the optimal performance among all the smoothing functions and values of p considered. We confirm that this is the case for different dimensions in the next experiment.

Table 1 Average CPU time and frequency of success for different values of p and different smoothing functions φ_i

p	Smoothing functions									
	φ_2		φ_3		φ_4		φ_5		φ_6	
	CPU time	Success rate	CPU time	Success rate	CPU time	Success rate	CPU time	Success rate	CPU time	Success rate
0.3	–	0	492.96	0.7	–	0	1440.03	0.1	–	0
0.5	550.40	0.2	213.78	1	547.32	0.4	526.04	0.3	271.77	1
0.7	285.47	1	192.48	1	237.29	0.9	252.07	1	209.16	1
0.9	187.35	1	143.85	1	167.48	1	179.99	1	156.40	1

The table shows the results for the noiseless case where $n = 32768$, $m = 16384$ and $K = \lfloor n/40 \rfloor$ and the entries of A are sampled from $\mathcal{N}(0, 1)$

Table 2 Average CPU time and frequency of success for different values of p and different smoothing functions φ_i

p	Smoothing functions									
	φ_2		φ_3		φ_4		φ_5		φ_6	
	CPU time	Success rate	CPU time	Success rate	CPU time	Success rate	CPU time	Success rate	CPU time	Success rate
0.3	–	0	495.181	0.8	–	0	–	0	–	0
0.5	586.03	0.4	211.681	1	525.22	0.5	527.27	0.3	296.24	0.9
0.7	285.53	0.9	197.75	1	237.70	1	256.40	1	186.95	1
0.9	187.01	1	144.03	1	167.70	1	178.19	1	155.81	1

The table shows the results for the noisy case where $n = 32768$, $m = 16384$ and $K = \lfloor n/40 \rfloor$ and the entries of A are sampled from $\mathcal{N}(0, 1)$

Experiment 2 We compare the performance of different smoothing functions with $p = 0.9$, which is the optimal value of p found in the previous experiment. We generate A, b and x^* as in Experiment 1 but we consider different dimensions $n \in \{2^{11}, 2^{12}, 2^{13}, 2^{14}, 2^{15}, 2^{16}\}$ and let $m = n/2$ and $K = \lfloor n/40 \rfloor$. The results are shown in Table 3. With $p = 0.9$, all of the smoothing functions were effective in solving the problem as we obtained 100% success rate from 10 independent simulations. Moreover, from Table 3, we see that the function φ_3 results to a more efficient algorithm both in the noiseless and noisy case.

3.2 Comparisons with Other Solvers for Sparse Recovery Problems

With the findings from the previous section, we now compare our algorithm with φ_3 and $p = 0.9$ to other famous algorithms in the literature. We consider popular solvers FISTA [2], NESTA [3], and FPC-BB [20] which are designed to solve the convex model (I). We also look at comparisons with those algorithms based on the nonconvex model (III) with $p = 0.5$ including HALF [31], IRUcLq-v [23] and alternating direction method of multipliers (ADMM) [28].

Experiment 3 We consider the same A, b and x^* as in Experiment 1 and compare our algorithm with FISTA, NESTA, FPC-BB, HALF, IRUcLq-v and ADMM. For this experiment, we also consider the two cases where b is noiseless or corrupted with noise. In addition, we explore the capability of the solvers to recover signals of different sparsity levels, namely $K \in \{\lfloor 0.05m \rfloor, \lfloor 0.1m \rfloor, \lfloor 0.2m \rfloor, \lfloor 0.3m \rfloor\}$. The results are summarized in Tables 4 and 5 .

We observe from both the noiseless and noisy cases that FPC-BB is the fastest solver for the case when $K = \lfloor 0.05m \rfloor$, $K = \lfloor 0.1m \rfloor$ and $K = \lfloor 0.2m \rfloor$ with 100% success rate. For these sparsity levels, the second fastest solver is our CG Algorithm with the function φ_3 and $p = 0.9$, followed by the HALF and FISTA algorithms. On the other hand, the solvers IRUcLq-v and ADMM were able to solve all these test problems but with very high computation time compared with other solvers.

While FPC-BB is the fastest for the aforementioned sparsity levels, it failed to solve all the problems for the highest signal density level considered which is $K = \lfloor 0.3m \rfloor$. For this case, CG Algorithm has the best performance in terms of CPU time and frequency of success among all the solvers considered. On the other hand, FISTA, HALF and IRUcLq-v algorithms were able to solve all the problems as well but the computation times required are larger, especially for IRUcLq-v. We point out that the methods IRUcLq-v and ADMM are the slowest solvers in all instances which is expected since these algorithms require the inversion of an $n \times n$ matrix, which is extremely costly for high dimensional problems.

To summarize, this experiment reveals that our method is efficient in handling different sparsity levels. In particular, it can recover signals which are dense (i.e. not strictly sparse) which is not achieved by FPC-BB, NESTA and ADMM.

Experiment 4 In this experiment, we look at the measurement matrix $A \in \mathbb{R}^{m \times n}$ considered in [15,23] where the entries are sampled from $\mathcal{N}(0, \frac{1}{m})$ and we let $n = 32768$, $m = n/4 = 8192$. The signal x^* and the vector b are generated as in the previous experiments.

The results are shown in Table 6. It is evident that our method is very efficient and effective in solving the problems for different sparsity levels. In fact, it is the fastest solver among all the algorithms considered followed by the FPC-BB and HALF algorithm. Notice, however, that the time required by the HALF method to solve the problem is almost three times longer than the time needed by our algorithm. With this, we see that the CG algorithm is very efficient

Table 3 Average CPU time of the algorithm using different smoothing functions with $p = 0.9$

n	Noiseless case					Noisy case				
	φ_2	φ_3	φ_4	φ_5	φ_6	φ_2	φ_3	φ_4	φ_5	φ_6
2048	0.32	0.27	0.29	0.35	0.31	0.33	0.29	0.30	0.37	0.31
4096	2.46	2.02	2.19	2.36	2.28	2.49	2.03	2.22	2.36	2.29
8192	10.84	8.12	9.37	10.18	8.80	10.45	8.01	9.28	10.09	8.67
16384	47.94	34.78	41.66	43.54	36.81	49.25	34.24	43.66	47.31	37.93
32768	183.71	144.51	167.88	177.72	154.49	187.21	148.41	168.69	178.98	158.69
65536	747.38	529.53	611.02	623.19	636.89	740.50	524.09	592.11	624.18	631.07

The frequency of success in all cases is 1. For each n, we set $m = n/2$ and $K = \lfloor n/40 \rfloor$. The entries of A are sampled from $\mathcal{N}(0, 1)$

Table 4 Average CPU time and frequency of success of different solvers for different sparsity levels with $n = 32768$ and $m = 16384$

Solver	Sparsity level (K)							
	$\lfloor 0.05m \rfloor$		$\lfloor 0.1m \rfloor$		$\lfloor 0.2m \rfloor$		$\lfloor 0.3m \rfloor$	
	CPU time	Success rate	CPU time	Success rate	CPU time	Success rate	CPU time	Success rate
CG algorithm	157.57	1	184.18	1	226.86	1	299.70	1
FISTA	259.09	1	283.90	1	308.37	1	345.23	1
NESTA	420.54	1	–	0	–	0	–	0
FPC-BB	106.97	1	121.15	1	175.65	1	–	0
HALF	187.59	1	224.99	1	335.86	1	543.17	1
IRUcLq-v	457.92	1	554.29	1	716.53	1	929.47	1
ADMM	578.85	1	665.60	1	967.74	1	–	0

The table shows the results for the noiseless case from 10 independent trials. The entries of A are sampled from $\mathcal{N}(0, 1)$

Table 5 Average CPU time and frequency of success of different solvers for different sparsity levels with $n = 32768$ and $m = 16384$

Solver	Sparsity level (K)							
	$\lfloor 0.05m \rfloor$		$\lfloor 0.1m \rfloor$		$\lfloor 0.2m \rfloor$		$\lfloor 0.3m \rfloor$	
	CPU time	Success rate	CPU time	Success rate	CPU time	Success rate	CPU time	Success rate
CG algorithm	159.44	1	183.83	1	227.61	1	294.96	1
FISTA	272.22	1	279.38	1	304.02	1	347.24	1
NESTA	422.98	1	–	0	–	0	–	0
FPC-BB	117.50	1	116.31	1	174.97	1	–	0
HALF	198.74	1	220.18	1	333.72	1	541.36	1
IRUcLq-v	454.91	1	642.26	1	704.10	1	912.16	1
ADMM	572.38	1	688.97	1	970.93	1	825.04	0.1

The table shows the results for the noisy case from 10 independent trials. The entries of A are sampled from $\mathcal{N}(0, 1)$

Table 6 Average CPU time and frequency of success of different solvers for different sparsity levels with $n = 32768$ and $m = 8192$

Solver	Sparsity level (K)							
	$\lfloor 0.05m \rfloor$		$\lfloor 0.1m \rfloor$		$\lfloor 0.2m \rfloor$		$\lfloor 0.3m \rfloor$	
	CPU time	Success rate	CPU time	Success rate	CPU time	Success rate	CPU time	Success rate
CG algorithm	24.66	1	33.00	1	51.29	1	96.44	1
FISTA	145.42	1	155.23	1	180.12	1	–	0
NESTA	–	0	–	0	–	0	–	0
FPC-BB	56.95	1	70.67	1	–	0	–	0
HALF	76.27	1	88.52	1	127.90	1	251.64	1
IRUcLq-v	469.05	1	616.98	1	978.63	1	1535.11	1
ADMM	585.93	1	760.49	1	806.68	0.1	–	0

The table shows the results for the noiseless case from 10 independent trials. The entries of the sensing matrix A are generated from $\mathcal{N}(0, 1/m)$

in handling the type of problem considered. In terms of the capability of the algorithms to solve the problem, it can also be seen that our algorithm (as well as HALF) is dominant as it was able to solve all the problems for all sparsity levels considered. On the other hand, FPC-BB algorithm failed to solve all the problems for sparsity levels $\lfloor 0.2m \rfloor$ and $\lfloor 0.3m \rfloor$.

To close this section, we summarize our numerical findings:

- The CG algorithm with the smoothing function φ_3 and $p = 0.9$ provides the best convergence time and success rate among all smoothing functions and values of p considered.
- Even though FPC-BB is usually the fastest solver among all, the difference between FPC-BB and our algorithm's CPU time of convergence is not large (see Experiment 3). Moreover, our algorithm obtains a good success rate for problems with relatively denser signals which is not achieved by FPC-BB. In this respect, our algorithm has a major advantage.
- Taking into account both the CPU time of convergence and the success rate of the algorithms in handling different problems, we believe that our algorithm provides a better performance among all algorithms considered as it can solve more problems at a reasonable computation time.
- While FPC-BB is often the fastest solver among other algorithms considered in this paper, we note that there are some problems where our method provides way better convergence time. We have shown this in Experiment 4 where we see that not only FPC-BB requires more than twice the time to obtain an approximate solution, but it also cannot solve problems with high density signals. On the other hand, the convergence times of other algorithms are much slower compared to our algorithm.

4 Conclusion

In this paper, we formulated a new model (6) which employs l_p-norm along with a special smoothing function that is motivated by re-weighted techniques. In general, the smoothing strategy along with a version of conjugate gradient is the main focus and contribution of this paper, both theoretically and and numerically. Not only we show global convergence, but also from various experiments and comparisons we have strong numerical evidence to verify our proposed algorithm is a good choice for tackling sparse signal reconstruction problem. In particular, our method can efficiently solve the sparse recovery problem even when the dimension of the problem is relatively high. Moreover, we have also demonstrated that dense signals can be recovered by our algorithm, which is not achievable by other algorithms. A future work that is worth considering is the theoretical and numerical study of the other smoothing models (SFI)-(SFV) along with the conjugate gradient algorithm, and numerical comparisons of these models for dealing with the sparse recovery problem.

Acknowledgements The authors would like to thank the anonymous referees for their valuable comments and suggestions which have significantly improved the original version of the paper.

Funding Caiying Wu and Jing Wang are supported by the Natural Science Foundation of Inner Mongolia Autonomous Region (2018MS01016). Jan Harold Alcantara and Jein-Shan Chen are supported by the Ministry of Science and Technology, Taiwan.

Compliance with Ethical Standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Baraniuk, R.: Compressive sensing. IEEE Signal Process. Mag. **24**, 118–121 (2007)
2. Beck, A., Teboulle, M.: A fast iterative shrinkage thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. **2**(1), 183–202 (2009)
3. Becker, S., Bobin, J., Candès, E.: NESTA: a fast and accurate first-order method for sparse recovery. SIAM J. Imaging Sci. **4**(1), 1–39 (2011)
4. Bruckstein, A.M., Donoho, D.L., Elad, M.: From sparse solutions of systems of equations to sparse modeling of signals and images. SIAM Rev. **51**, 34–81 (2009)
5. Candès, E.J.: The restricted isometry property and its implications for compressed sensing. Comptes Rendus Mathematique **346**(9–10), 589–592 (2008)
6. Candès, E.J., Randall, P.A.: Highly robust error correction by convex programming. IEEE Trans. Inf. Theory **54**, 2829–2840 (2008)
7. Candès, E.J., Romberg, J., Tao, T.: Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Trans. Inf. Theory **52**, 489–509 (2006)
8. Candès, E.J., Romberg, J.K., Tao, T.: Stable signal recovery from incomplete and inaccurate measurements. Commun. Pure Appl. Math. **59**(8), 1207–1223 (2006)
9. Candès, E.J., Tao, T.: Near optimal signal recovery from random projections: universal encoding strategies. IEEE Trans. Inf. Theory **52**, 5406–5425 (2004)
10. Candès, E.J., Tao, T.: Decoding by linear programming. IEEE Trans. Inf. Theory **51**(12), 4203–4215 (2005)
11. Candès, E.J., Tao, T.: Near optimal signal recovery from random projections: Universal encoding strategies? IEEE Trans. Inf. Theory **52**(12), 5406–5425 (2006)
12. Chen, X., Ye, Y., Wang, Z., Ge, D.: Complexity of unconstrained $L_2 - L_p$ minimization. Math. Program. **143**, 371–383 (2014)
13. Chen, X., Zhou, W.: Smoothing nonlinear conjugate gradient method for image restoration using nonsmooth nonconvex minimization. SIAM J. Imaging Sci. **3**(4), 765–790 (2010)
14. Chen, X., Zhou, W.: Convergence of the reweighted l_1 minimization algorithm for $l_2 - l_p$ minimization. Comput. Optim. Appl. **59**, 47–61 (2014)
15. Daubechies, I., DeVore, R., Fornasier, M., Güntük, S.: Iteratively reweighted least squares minimization for sparse recovery. Commun. Pure Appl. Math. **63**, 1–38 (2010)
16. Donoho, D.: Compressed sensing. IEEE Trans. Inf. Theory **52**(4), 1289–1306 (2006)
17. Friedman, J., Hastie, T., Tibshirani, R.: A note on the group lasso and a sparse group lasso. Stat. Theory (math.ST). Submitted on 5 Jan (2010)
18. Ge, D., Jiang, X., Ye, Y.: A note on the complexity of L_p minimization. Math. Program. **129**, 285–299 (2011)
19. Gribnoval, R., Nielsen, M.: Sparse decompositions in unions of bases. IEEE Trans. Inf. Theory **49**, 3320–3325 (2003)
20. Hale, E.T., Yin, W., Zhang, Y.: Fixed-point continuation for ℓ_1-minimization: methodology and convergence. SIAM J. Optim. **19**, 1107–1130 (2008)
21. Koh, K., Kim, S., Boyd, S.: An interior-point method for large scale l_1 regularized logistic regression. J. Mach. Learn. Res. **8**, 1519–1555 (2007)
22. Lai, M.-J., Wang, J.: An unconstrained l_q minimization with $0 < q \leq 1$ for sparse solutions of underdetermined linear system. SIAM J. Optim. **21**(1), 82–101 (2011)
23. Lai, M.-J., Xu, Y., Yin, W.: Improved iteratively reweighted least squares for unconstrained smoothed l_q minimization. SIAM J. Numer. Anal. **51**(2), 927–957 (2013)
24. Natarajan, B.K.: Sparse approximate solutions to linear systems. SIAM J. Comput. **24**, 227–234 (1995)
25. Nguyen, C.T., Saheya, B., Chang, Y.-L., Chen, J.-S.: Unified smoothing functions for absolute value equation associated with second-order cone. Appl. Numer. Math. **135**, 206–227 (2019)
26. Simonn, N., Friedman, J., Hastieand, T., Tibshirani, R.: A sparse-group lasso. J. Comput. Graph. Stat. **22**, 231–245 (2013)
27. Wang, X., Liu, F., Jiao, L.C., Wu, J., Chen, J.: Incomplete variables truncated conjugate gradient method for signal reconstruction in compressed sensing. Inf. Sci. **288**, 387–411 (2014)
28. Wang, Y., Yin, W., Zheng, J.: Global convergence of ADMM in nonconvex nonsmooth optimization. J. Sci. Comput. **78**, 29–63 (2019)
29. Wu, C., Zhan, J., Lu, Y., Chen, J.-S.: Signal reconstruction by conjugate gradient algorithm based on smoothing l_1-norm. Calcolo **56**(4), 1–26 (2019)
30. Wu, C., Zhang, J., Chen, J.-S.: An elastic unconstrained $l_q - l_1$ minimization for finding sparse solution, submitted manuscript (2019)

31. Xu, Z., Chang, X., Xu, F., Zhang, H.: $L_{1/2}$ regularization: a thresholding representation theory and a fast solver. IEEE Trans. Neural Netw. Learn. Syst. **23**, 1013–1027 (2012)
32. Yin, K., Xiao, Y.H., Zhang, M.L.: Nonlinear conjugate gradient method for l_1-norm regularization problems in compressive sensing. J. Comput. Inf. Syst. **7**, 880–885 (2011)
33. Zhang, C.: Nearly unbiased variable selection under minimax concave penalty. Ann. Stat. **38**, 894–942 (2010)
34. Zhang, Y., Ye, W.: Sparse recovery by the iteratively reweighted l_1 algorithm for elastic $l_2 - l_q$ minimization. Optimization **66**(10), 1677–1687 (2017)
35. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. J. R. Stat. Soc. Ser. B-Stat. Methodol. **67**, 301–320 (2005)
36. Zhu, H., Xiao, Y.H., Wu, S.Y.: Large sparse signal recovery by conjugate gradient algorithm based on smoothing technique. Comput. Math. Appl. **66**, 24–32 (2013)