Hindawi
Computational Intelligence and Neuroscience
Volume 2021, Article ID 9943519, 8 pages
https://doi.org/10.1155/2021/9943519

Research Article

An Inexact Penalty Decomposition Method for Sparse Optimization

Zhengshan Dong,[1] **Geng Lin**,[1] **and Niandong Chen**[2]

[1]*College of Mathematics and Data Science, Minjiang University, Fuzhou 350108, China*
[2]*New Huadu Business School of Minjiang University, Minjiang University, Fuzhou 350108, China*

Correspondence should be addressed to Niandong Chen; 171944938@qq.com

Received 23 March 2021; Accepted 30 June 2021; Published 15 July 2021

Academic Editor: Henry Man Fai Leung

The penalty decomposition method is an effective and versatile method for sparse optimization and has been successfully applied to solve compressed sensing, sparse logistic regression, sparse inverse covariance selection, low rank minimization, image restoration, and so on. With increase in the penalty parameters, a sequence of penalty subproblems required being solved by the penalty decomposition method may be time consuming. In this paper, an acceleration of the penalty decomposition method is proposed for the sparse optimization problem. For each penalty parameter, this method just finds some inexact solutions to those subproblems. Computational experiments on a number of test instances demonstrate the effectiveness and efficiency of the proposed method in accurately generating sparse and redundant representations of one-dimensional random signals.

1. Introduction

In this paper, we consider solving the following sparse optimization problem by an inexact penalty decomposition (iPD) method:

$$\begin{aligned} \min_{x\in\mathcal{X}} \quad & l(x)+\lambda\|x\|_0, \\ \text{s.t.} \quad & g(x)\le 0, h(x)=0, \end{aligned} \tag{1}$$

where $\lambda \ge 0$ controls the sparsity of the solution, $\mathcal{X} \subset \mathbb{R}^n$ is a closed convex set in the n-dimensional Euclidean space $\mathbb{R}^n$, l: $\mathbb{R}^n \longrightarrow \mathbb{R}$, $g(x)$: $\mathbb{R}^n \longrightarrow \mathbb{R}^p$ are continuously differentiable convex functions, h: $\mathbb{R}^n \longrightarrow \mathbb{R}^m$ is an affine function, and $\|x\|_0$ denotes the number of nonzero components of x.

Sparse optimization is to solve some problems whose solutions are sparse or compressed. And it has attracted considerable attention in the past ten years since its broad applications, such as signal (image) processing [1–3], linear regression [4], inverse problem [5], model selection [6], and machine learning [6, 7]. In those applications, most information of interest has or can be coded by much low dimension though its own dimension is high.

However, problem (1) is NP hard even though for some simple special cases [8]. Even so, many methods have been proposed for some special cases of problem (1). These methods can be classified into four categories: (1) greedy methods: matching pursuit [9, 10] and greedy coordinate descent [11]; (2) l_1-norm relaxation methods: gradient projection [12, 13], iterative shrinkage-thresholding [5, 14], iterative reweighted method [15], alternating direction method [16], and homotopy method [17–20]; (3) l_p-norm $(0 < p < 1)$ relaxation methods [1, 2, 21]; and (4) l_0-norm based methods, e.g., penalty decomposition method [22], block decomposition method [23], iterative hard thresholding method [22, 24–29], and so on. In this paper, we mainly discuss the PD method.

The PD method was proposed for solving the general l_0-norm minimization problem (1) by Lu and Zhang in [22]. And it had been successfully applied to solve compressed sensing [22], sparse logistic regression [22], sparse inverse covariance selection [22], low rank minimization [30], image restoration [3] problems, and so on. Moreover, the PD method is theoretically sound. Lu et al. stated that any accumulation point of the sequence generated by the PD

method satisfies the first-order optimality conditions of problem (1) when the Robinson condition holds. Hence, the PD method is an effective and versatile method for sparse optimization. However, since the PD method found exact solutions of subproblems for each penalty parameter, it may be time consuming in practice.

In this paper, an inexact penalty decomposition (iPD) method is proposed for the sparse optimization problem (1). The iPD method just finds some inexact solutions to those subproblems for each penalty parameter. In more detail, for the first convex subproblem, the iPD method just takes one gradient step and then goes to solve the second nonconvex subproblem. The second subproblem can be solved by the iterative hard thresholding method [26]. After the two steps, the penalty parameter is updated. Computational experiments on a number of random instances demonstrate the effectiveness of the proposed method in accurately generating sparse and redundant representations of one-dimensional random signals.

The rest of this paper is organized as follows. Section 2 is the preliminary, in which some notations and the basic method are described. Section 3 presents the iPD method. Computational experiments are presented in Section 4, and conclusions are drawn in Section 5.

2. Preliminaries

2.1. Notations. In this subsection, some notations are presented to simplify presentation. The transpose of a vector $x \in \mathbb{R}^n$ is denoted by x^T. If without special statement, all norms used are the Euclidean norm, denoted by $\|\cdot\|_2$. $\mathcal{P}_{\mathcal{X}}(\cdot)$ denotes projection on a set $\mathcal{X}$. Given a vector $x \in \mathbb{R}^n$, the nonnegative part of x is denoted by x^+, i.e., $x^+ = \max(x, 0)$. The index of nonzero components of a vector x is denoted by $S(x) = \{i : x_i \neq 0\}$ (called support set) and $S_k := S(x^k)$. The size of $S(x)$ is denoted as $s = |S(x)|$.

Now, let us consider problem (1). It is easy to verify that problem (1) is equivalent to the following problem:

$$\begin{aligned} &\min_{x,y\in\mathcal{X}} \ l(x) + \lambda\|y\|_0, \\ &\text{s.t.} \quad g(x) \le 0, h(x) = 0, \quad x = y. \end{aligned} \tag{2}$$

And the relative penalty function of problem (2) is defined as

$$\begin{aligned} p_\rho(x, y) = l(x) + \lambda\|y\|_0 \\ + \frac{\rho}{2}\left(\left\|[g(x)]^+\right\|_2^2 + \|h(x)\|_2^2 + \|x - y\|_2^2\right), \end{aligned} \tag{3}$$

where $\rho > 0$ is the penalty parameter.

For simplicity, we also denote

$$\begin{aligned} F_\rho(x) &= l(x) + \lambda\|x\|_0 + \frac{\rho}{2}\left(\left\|[g(x)]^+\right\|_2^2 + \|h(x)\|_2^2\right), \\ f(x) &= \frac{1}{2}\left(\left\|[g(x)]^+\right\|_2^2 + \|h(x)\|_2^2\right), \\ q_\rho(x, y) &= l(x) + \frac{\rho}{2}\left(\left\|[g(x)]^+\right\|_2^2 + \|h(x)\|_2^2 + \|x - y\|_2^2\right). \end{aligned} \tag{4}$$

2.2. The PD Method. In this subsection, we show the PD method proposed in [22]. First, the outline of the PD method is as presented in Algorithm 1. Then, we explain why the PD method is time consuming by a random example.

Remark 1

(i) The termination condition in Step 8 of Algorithm 1 is used to establish the global convergence of the PD method. In practice, the termination criterion is based on the relative change of the sequence $\{(x^{k,i}, y^{k,i})\}$ such as the sequence satisfying

$$\max\left\{\frac{\left\|x^{k,i} - x^{k,i-1}\right\|_\infty}{\max\left(\left\|x^{k,i}\right\|_\infty, 1\right)}, \frac{\left\|y^{k,i} - y^{k,i-1}\right\|_\infty}{\max\left(\left\|y^{k,i}\right\|_\infty, 1\right)}\right\} \le \epsilon_I, \tag{5}$$

for some $\epsilon_I > 0$. In addition, the PD method terminates the outer iterations when

$$\left\|x^k - y^k\right\|_\infty \le \epsilon_O, \tag{6}$$

holds for some $\epsilon_O > 0$.

(ii) The second subproblem, i.e., in Step 6 of Algorithm 1,

$$y^{k,i+1} \in \arg\min_y \lambda\|y\|_0 + \frac{\rho_k}{2}\left\|y - x^{k,i+1}\right\|_2^2, \tag{7}$$

has a closed-form solution [26].

$$y_j^{k,i+1} = \begin{cases} x_j^{k,i+1}, & \text{if } \left|x_j^{k,i+1}\right| > \sqrt{\dfrac{2\lambda}{\rho_k}}, \\ 0, & \text{otherwise}, \end{cases} \tag{8}$$

where $[\cdot]_j$ denotes the j-th entry of a vector, $j \in \{1, 2, \ldots, n\}$.

In Step 5 of Algorithm 1, minimizing the function $p_{\rho_k}(x, y^k)$ with respect to x is a convex problem. There exist many efficient methods for this purpose if $\mathcal{X}$ is simple. However, for each penalty parameter, the PD method solves the penalty subproblems a few times until some termination conditions are reached, which is time consuming.

Input: $\rho_0 > 0$, x^0, $\sigma > 1$;
Output: $\widehat{x}$;
(1) initialization $k \longleftarrow 0$, $y^{0,0} = x^0$;
(2) **repeat**
(3) $\quad i \longleftarrow 0$, $y^{k,0} = y^k$;
(4) $\quad$ **repeat**
(5) $\quad\quad x^{k,i+1} \in \arg\min_x p_{\rho_k}(x, y^{k,i})$;
(6) $\quad\quad y^{k,i+1} \in \arg\min_y p_{\rho_k}(x^{k,i+1}, y)$;
(7) $\quad\quad i \longleftarrow i + 1$;
(8) $\quad$ **until** $\|\mathscr{P}_{\mathscr{X}}(x^{k,i} - \nabla_x q_{\rho_k}(x^{k,i}, y^{k,i}))\|_2 \le \epsilon_k$
(9) $\quad y^{k+1} \longleftarrow y^{k,i}$;
(10) $\quad x^{k+1} \longleftarrow x^{k,i}$;
(11) $\quad \rho_{k+1} \longleftarrow \sigma\rho_k$;
(12) $\quad k \longleftarrow k + 1$;
(13) **until** some termination conditions reach
(14) $\widehat{x} \longleftarrow y^k$;

Algorithm 1: The PD method [22].

Consider a special case—compressed sensing [31]. One important task of compressed sensing is to find the sparsest solution to the underdetermined linear system, which is formulated as

$$\begin{aligned} \min_x \quad & \|x\|_0, \\ \text{s.t.} \quad & Ax = b, \end{aligned} \tag{9}$$

where $A \in \mathbb{R}^{m\times n}$ is the sensing matrix and $b \in \mathbb{R}^m$ is the observation data. For this special problem, $f(x) = (1/2)\|Ax - b\|_2^2$ and $F_\rho(x) = \|x\|_0 + (\rho/2)\|Ax - b\|_2^2$. The value of $f(x)$ is called data fidelity, and it can measure the feasibility of a solution x. $F_\rho(x)$ is the penalty function of problem (9).

Example 1. We generate a sparse vector $\overline{x}^*$ with length $n = 5000$ and $s = 100$ nonzero components. These components independently follow the standard Gaussian distribution, and their locations are assigned randomly to $\overline{x}^*$. Then, we create a Gaussian random matrix A with size 1000×5000, and let $b = A\overline{x}^*$. Then, we solve this instance by the PD method package, and the process data are presented as Figure 1.

Figure 1 shows that the value of $f(x)$ decreases slowly. It decreases steep just at the first few steps for each penalty parameter. There are many almost null steps during the process. And the value of the penalty function $F_\rho(x)$ increases too much when updating the penalty parameter. Hence, we can just take one or a few iterations for each penalty parameter to save some time. In Section 3, we will improve the PD method by the above observations.

3. The Proposed Method

In this section, we describe the process of the iPD method. From the outline of Algorithm 1, we find that, for each penalty parameter ρ_k, the block coordinate descent method needs to alternately solve two minimization subproblems many times, and an example in Section 2 shows that there are many almost null step for each penalty parameter. Hence, the original PD method may be time consuming if convergence speed of the block coordinate descent is slow.

Motivated by the analysis in Section 2 and the above demonstration, we accelerate the PD method by alteratively solving the two penalty subproblems once a time after updating the penalty parameter. For solving the first penalty subproblem, a gradient step is taken, and its step-length is searched by the backtracking line search method.

Now, we present the outline of the accelerated penalty decomposition method as follows.

Remark 2. A practical termination criterion in Step 11 of Algorithm 2 can be

$$\frac{\left\|x^k - y^k\right\|_\infty}{\max\left(\left\|x^k\right\|_\infty, 1\right)} \le \text{tol}, \tag{10}$$

for some $\text{tol} > 0$.

Theorem 1. *If the gradient of the function $p_{\rho_k}(x, y^k)$ with respect to x is Lipschitz continuous (its Lipschitz constant is denoted as L_p), then the line search between Steps 3–6 can be terminated in a finite number of iterations.*

Proof. Since $p_{\rho_k}(x^{k+1}, y^k)$ satisfies

$$p_{\rho_k}\left(x^{k+1}, y^k\right) \le p_{\rho_k}\left(y^k, y^k\right) + \nabla_x p_{\rho_k}\left(y^k, y^k\right)^T\left(x^{k+1} - y^k\right) + \frac{L_p}{2}\left\|x^{k+1} - y^k\right\|_2^2, \tag{11}$$

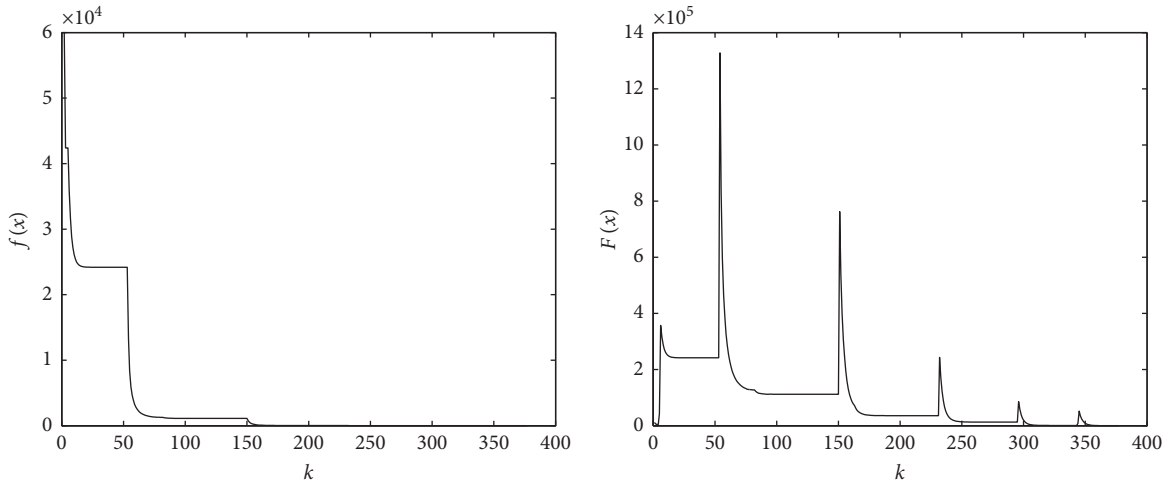

Figure 1: Iteration process of penalty decomposition for solving compressed sensing with size $m = 1000, n = 5000$, and $s = 100$: (a) data fidelity at each iteration; (b) penalty function value at each iteration.

Input: $\rho_0 > 0$, x^0, $\sigma_0 > 1$, $L_0 > 0$, $\gamma_{\text{inc}} > 1$, $\gamma_{\text{dec}} > 1$;
Output: $\widehat{x}$;
(1) initialization $k \longleftarrow 0$, $y^k = x^k$;
(2) **repeat**
(3) **while** $p_{\rho_k}(x^{k+1}, y^k) + (\eta/2)\|x^{k+1} - y^k\|_2^2 > p_{\rho_k}(y^k, y^k)$ **do**
(4) $L_k = \min(\gamma_{\text{inc}} L_k, L_{\max})$
(5) $x^{k+1} = y^k - (\nabla_x p_{\rho_k}(y^k, y^k)/L_k)$;
(6) **end while**
(7) $y^{k+1} \in \operatorname{argmin}_y p_{\rho_k}(x^{k+1}, y)$;
(8) $\rho_{k+1} = \sigma_k \rho_k$
(9) $L_{k+1} = \min((L_k/\gamma_{\text{dec}}), L_{\min})$
(10) $k \longleftarrow k + 1$;
(11) **until** some termination conditions reach
(12) $\widehat{x} \longleftarrow y^k$;

Algorithm 2: The inexact PD method.

it together with $x^{k+1} = y^k - (\nabla_x p_{\rho_k}(y^k, y^k)/L_k)$ implies that

$$p_{\rho_k}\left(x^{k+1}, y^k\right) \le p_{\rho_k}\left(x, y^k\right) - \left(L_k - \frac{L_p}{2}\right)\left\|x^{k+1} - x^k\right\|_2^2. \quad (12)$$

Then, if $L_k \ge ((L_p + \eta)/2)$,

$$p_{\rho_k}\left(x^{k+1}, y^k\right) + \frac{\eta}{2}\left\|x^{k+1} - y^k\right\|_2^2 \le p_{\rho_k}\left(y^k, y^k\right), \quad (13)$$

holds, which means that the while loop in Algorithm 2 terminates if $L_k \ge ((L_p + \eta)/2)$. Let $\overline{L}_k$ be the final value of L_k after the while loop. Then, $(\overline{L}_k/\gamma_{\text{inc}}) \le ((L_p + \eta)/2)$ holds, i.e., $\overline{L}_k \le (\gamma_{\text{inc}}(L_p + \eta)/2)$. Let $\widehat{n}_k$ be the number of iterations in the while loop at the k-th iteration. Then, one can get that

$$L_{\min}\gamma_{\text{inc}}^{\widehat{n}_k - 1} \le L_k^0 \gamma_{\text{inc}}^{\widehat{n}_k - 1} \le \overline{L}_k \le \frac{\gamma_{\text{inc}}\left(L_p + \eta\right)}{2}, \quad (14)$$

where L_k^0 is the initial value of L_k in the line search. Therefore,

$$\widehat{n}_k \le N := \left\lceil \frac{\log\left(L_p + \eta\right) - \log\left(2L_{\min}\right)}{\log(\gamma)} + 2 \right\rceil. \quad (15)$$

□

4. Experiments

In this section, we implement the proposed accelerated PD method to solve the compressed sensing problem. To verify the efficiency of PD empirically, a large number of computational experiments are performed on one-dimensional random signals. We mainly compare the performance of our improved PD method with that of the original PD method [22]. All experiments were performed on a personal computer with an Intel(R) Core(TM)i7-7700HQ CPU (2.80 GHz) and 8 GB memory, using a MATLAB toolbox (version R2018b).

We compare the performance of the compared methods by the CPU time (in seconds) required, the size of the

Table 1: Parameter settings in the acceleration of the PD method.

Parameter	Value
x_0	0
ρ	$\rho_0 = 10, \rho_{k+1} = \min(1.1\rho_k, 10^{15})$
tol	10^{-6}
η	1
L	$L_0 = 0.1 \max(\|A_j\|_2^2)$ $\gamma_{\text{inc}} = 2, \gamma_{\text{dec}} = 3$

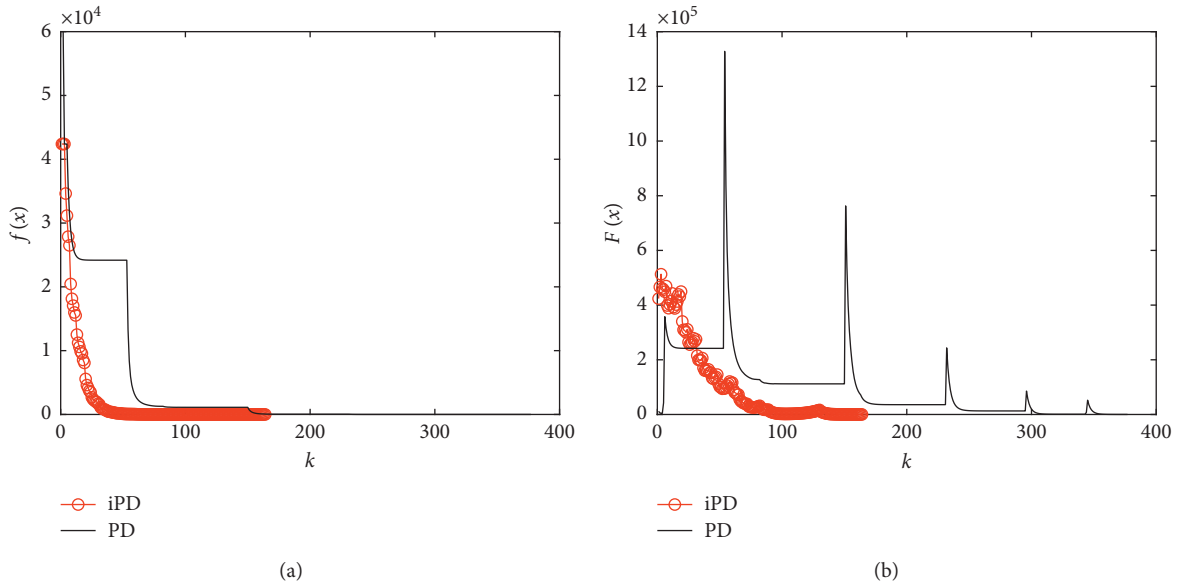

Figure 2: Iteration process of the compared methods for solving compressed sensing with size $m = 1000, n = 5000$, and $s = 100$: (a) data fidelity at each iteration; (b) penalty function value at each iteration.

support set of the reconstructed data $\hat{x}$, and the mean squared error (MSE) with respect to $\overline{x}^*$, which is defined as

$$\text{MSE} = \frac{1}{n}\left\|\hat{x} - \overline{x}^*\right\|, \tag{16}$$

and the data fidelity of $A\hat{x} - y$ is defined as

$$\text{DF} = \frac{1}{2}\|A\hat{x} - y\|^2, \tag{17}$$

and NS as the number of successfully recovered instances. We say a signal $\hat{x}$ is successfully recovered if the positions of the nonzero components of $\hat{x}$ are the same as $\overline{x}^*$ and the corresponding MSE value is less than 10^{-4}.

4.1. Data Generation and Parameter Setting. Each instance is generated randomly with size (m, n, s), where $m \times n$ is the dimension of matrix A and s is the sparsity level, such as $m = 1000$, $n = 5000$, and $s = 100$. The elements of matrix A follow the Gaussian distribution. The vector $\overline{x}^*$ is generated with the same distribution at s randomly chosen coordinates. Finally, the vector b is generated by $b = A\overline{x}^*$.

Unless otherwise stated, all parameters in the PD method are set as default, and parameters in the IPD package are set as in Table 1.

4.2. Compare with the Original PD Method. Firstly, we compare the iteration process of the iPD method with that of the PD method on a random instance. All parameters are set as before, and the problem size is $m = 1000, n = 5000$, and $s = 100$. Figure 2 describes the data fidelity and the penalty function value over the iteration process. From Figure 2(a), we find that the iPD method does not have many null steps, and the values of data fidelity generated by the iPD method decrease much fast than those of the original PD method. Furthermore, the iPD method just requires about 150 steps while the original PD method requires about 400 steps. And the running time of the iPD method is about 7 seconds, which is less than half of the time required by the original PD method. Moreover, the penalty function value generated by the iPD method is much stable than that by the original PD method.

In the second experiment, we compare the accelerated PD method with its original PD method at different sampling numbers. We fix the dimension $m = 5000$ and the sparsity level $s = 100$. For each sampling number m, 100 instances are generated, and the averaged performance of the two methods is presented in Figure 3.

From Figure 3(a), we see that the accelerated PD method requires not more than 10 seconds while the original PD method requires much more time. And the time required by

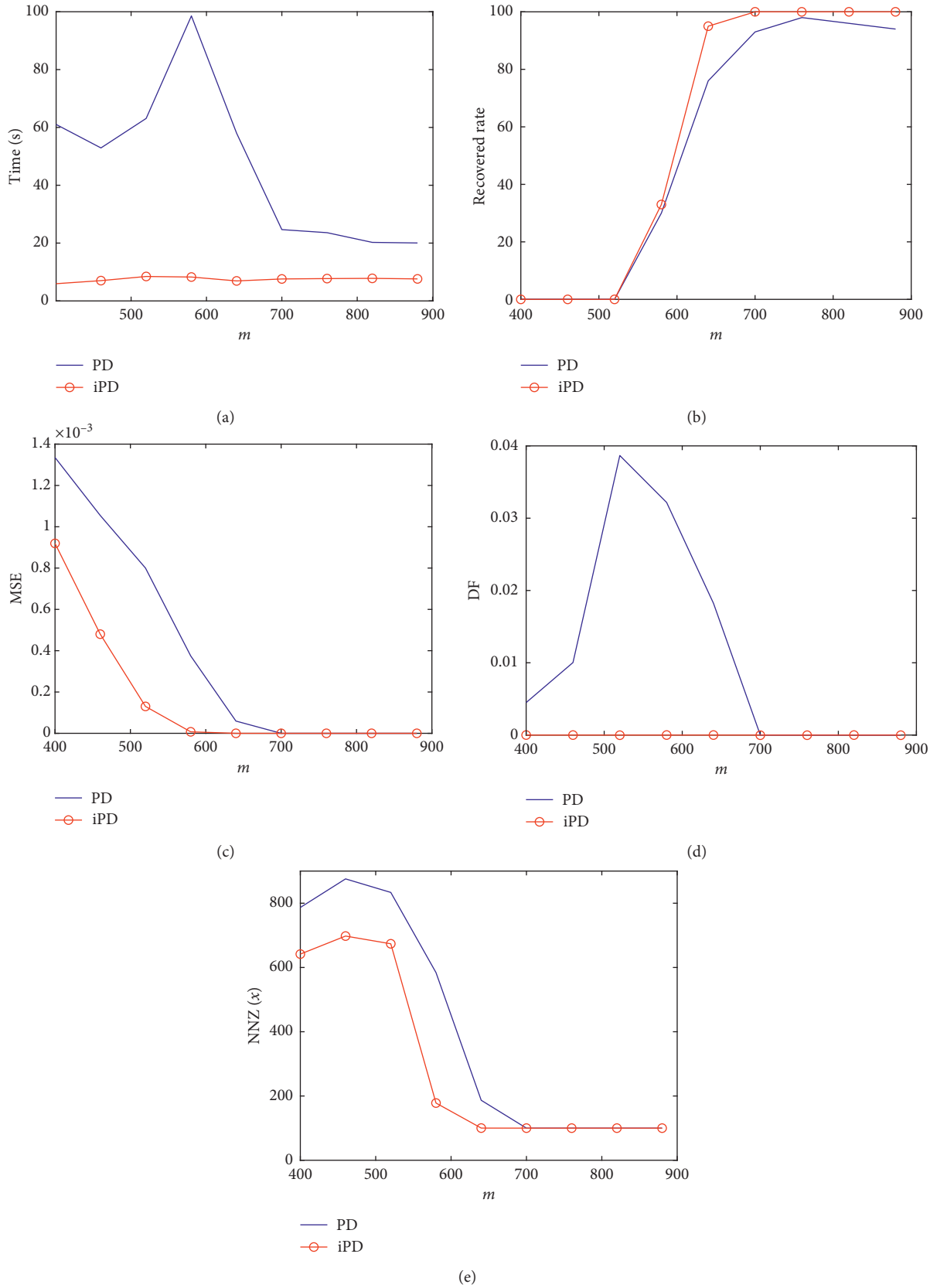

Figure 3: Averaged results of the penalty decomposition methods for the compressed sensing problem with different sampling numbers on 100 instances: (a) CPU time over sampling number; (b) recovered rate over sampling number; (c) MSE over sampling number; (d) data fidelity over sampling number; (e) number of nonzero components over sampling number.

Table 2: Averaged results on 100 instances with size $m = 1000$ and $n = 5000$ for each sparsity level s

Algorithm	s	Time (s)	NNZ	MSE	DF	NS
PD	50	18.2	50	5.62×10^{-9}	5.95×10^{-7}	100
	100	21.69	99.95	1.06×10^{-8}	2.61×10^{-6}	95
	150	26.36	149.88	2.50×10^{-8}	3.50×10^{-5}	89
	200	76.38	297.65	6.72×10^{-5}	8.19×10^{-2}	73
	250	102.54	1225.21	1.22×10^{-3}	3.53×10^{-1}	0
	270	105.18	1283.92	1.58×10^{-3}	1.93×10^{-1}	0
iPD	50	6.72	50	3.22×10^{-9}	1.77×10^{-7}	100
	100	8.56	99.99	5.20×10^{-9}	3.62×10^{-7}	99
	150	9.78	150	7.31×10^{-9}	5.85×10^{-7}	100
	200	10.32	200	9.32×10^{-9}	7.66×10^{-7}	100
	250	11.75	656.87	5.50×10^{-5}	1.03×10^{-6}	30
	270	12.75	1235.9	3.67×10^{-4}	9.72×10^{-7}	1

the accelerated PD method is stable at different sampling numbers. Figure 3(b) shows that the recovered rate by the accelerated PD method is higher than that by the original PD method when m is bigger than 600. When the sampling number is bigger than 700, the accelerated PD method can recover all signals successfully. We find that the MSE value and the DF value generated by the accelerated PD method are lower than those generated by the original PD method. The averaged number of nonzero components also shows that the accelerated PD method performs better.

In the next experiment, we compare the accelerated PD method with its original version for solving the compressed sensing problem with different sparsity levels s. All parameters are set as the same value as those stated before. The averaged computational results on 100 instances are presented in Table 2.

From Table 2, we find that the PD method not works well when the sparsity level is greater than 150, especially when it is greater than 200. However, the sparsity level recovered by the iPD method can reach 200. When the two methods can recover sparse signals, the iPD method just needs about one third of the time required by the PD method. Moreover, the recovered rate of the iPD method is higher than that of the original PD method. From MSE and DF value, we see that the signals recovered by the iPD method are more exact than those recovered by the PD method. When $s = 100$, there is one instance not recovered exactly by the iPD method since there exist several very small components and one of them is not recovered.

5. Conclusions

In this paper, we have proposed an acceleration of the penalty decomposition for the sparse approximation problem. The proposed method does not solve the penalty subproblems exactly and alternately solve penalty subproblems once a time after updating penalty parameters. We show that this method enhances the performance of the penalty decomposition method by computational experiments on a number of random instances for solving the compressed sensing problem. The experiments demonstrate that the proposed method indeed improves the original PD method since it recovers better solutions with less running time.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was funded partly by the Natural Science Foundation of Fujian Province of China, under grant 2020J01843, and the Science and Technology Project of the Education Bureau of Fujian, China, under grant JAT200403.

References

[1] R. Chartrand, "Exact reconstruction of sparse signals via nonconvex minimization," *IEEE Signal Processing Letters*, vol. 14, no. 10, pp. 707–710, 2007.

[2] X. Xiaojun Chen, M. K. Ng, and C. Chao Zhang, "Non-lipschitz l_p-regularization and box constrained model for image restoration," *IEEE Transactions on Image Processing*, vol. 21, no. 12, pp. 4709–4721, 2012.

[3] Y. Zhang, B. Dong, and Z. Lu, "l_0 minimization for wavelet frame based image restoration," *Mathematics of Computation*, vol. 82, no. 282, pp. 995–1015, 2013.

[4] G. Mateos, J. A. Bazerque, and G. B. Giannakis, "Distributed sparse linear regression," *IEEE Transactions on Signal Processing*, vol. 58, no. 10, pp. 5262–5276, 2010.

[5] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[6] Z. Zechao Li, J. Jing Liu, Y. Yi Yang, X. Xiaofang Zhou, and H. Hanqing Lu, "Clustering-guided sparse structural learning for unsupervised feature selection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 9, pp. 2138–2150, 2014.

[7] Z. Noorie and F. Afsari, "Sparse feature selection: relevance, redundancy and locality structure preserving guided by pairwise constraints," *Applied Soft Computing*, vol. 87, 2020.

[8] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM Journal on Computing*, vol. 24, no. 2, pp. 227–234, 1995.

[9] S. G. Mallat and Z. Zhifeng Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.

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

[11] H. Fang, Z. Fan, Y. Sun, and M. Friedlander, "Greed meets sparsity: Understanding and improving greedy coordinate descent for sparse optimization," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, pp. 434–444, Palermo, Italy, June 2020.

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

[13] Y. Nesterov, "Gradient methods for minimizing composite functions," *Mathematical Programming*, vol. 140, no. 1, pp. 125–161, 2013.

[14] P. L. Combettes and V. R. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Modeling & Simulation*, vol. 4, no. 4, pp. 1168–1200, 2005.

[15] E. J. Candès, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted l_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5-6, pp. 877–905, 2008.

[16] J. Yang and Y. Zhang, "Alternating direction algorithms for ℓ_1-problems in compressive sensing," *SIAM Journal on Scientific Computing*, vol. 33, no. 1, pp. 250–278, 2011.

[17] D. M. Malioutov and A. S. Willsky, "Homotopy continuation for sparse signal representation," *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 5, pp. 733–736, 2005.

[18] S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo, "Sparse reconstruction by separable approximation," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2479–2493, 2009.

[19] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *The Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[20] E. T. Hale, W. Yin, and Y. Zhang, "Fixed-point continuation for ℓ_1-minimization: methodology and convergence," *SIAM Journal on Optimization*, vol. 19, no. 3, pp. 1107–1130, 2008.

[21] M.-J. Lai, Y. Xu, and W. Yin, "Improved iteratively reweighted least squares for unconstrained smoothed ℓ_q minimization," *SIAM Journal on Numerical Analysis*, vol. 51, no. 2, pp. 927–957, 2013.

[22] Z. Lu and Y. Zhang, "Sparse approximation via penalty decomposition methods," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2448–2478, 2013.

[23] G. Yuan, L. Shen, and W. Zheng, "A block decomposition algorithm for sparse optimization," in *KDD'20: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 275–285, CA, USA, July 2020.

[24] T. Blumensath and M. E. Davies, "Iterative thresholding for sparse approximations," *Journal of Constructive Approximation*, vol. 14, no. 5-6, pp. 629–654, 2008.

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

[26] Z. Lu, "Iterative hard thresholding methods for l_0 regularized convex cone programming," *Mathematical Programming*, vol. 147, no. 1-2, pp. 125–154, 2014.

[27] M. Nikolova, "Description of the minimizers of least squares regularized with ℓ_0-norm. Uniqueness of the global minimizer," *SIAM Journal on Imaging Sciences*, vol. 6, no. 2, pp. 904–937, 2013.

[28] T. Blumensath and M. E. Davies, "Normalized iterative hard thresholding: Guaranteed stability and performance," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 298–309, 2010.

[29] T. Blumensath, "Accelerated iterative hard thresholding," *Signal Processing*, vol. 92, no. 3, pp. 752–756, 2012.

[30] Z. Lu and Y. Zhang, "Penalty decomposition methods for rank minimization," *Optimization Method and Software*, 2015.

[31] R. Baraniuk, "Compressive Sensing," *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 118–121, 2007.