Numer. Math. Theor. Meth. Appl. Vol. **2**, No. 4, pp. 377-402
doi: 10.4208/nmtma.2009.m9002s November 2009

A Coordinate Gradient Descent Method for Nonsmooth Nonseparable Minimization

Zheng-Jian Bai[1], Michael K. Ng[2,*] and Liqun Qi[3]

[1] *School of Mathematical Sciences, Xiamen University, Xiamen 361005, China.*
[2] *Centre for Mathematical Imaging and Vision and Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong.*
[3] *Department of Applied Mathematics, The Hong Kong Polytechnic University, Hung Hom, Hong Kong.*

Received 18 February 2009; Accepted (in revised version) 9 June 2009

Abstract. This paper presents a coordinate gradient descent approach for minimizing the sum of a smooth function and a nonseparable convex function. We find a search direction by solving a subproblem obtained by a second-order approximation of the smooth function and adding a separable convex function. Under a local Lipschitzian error bound assumption, we show that the algorithm possesses global and local linear convergence properties. We also give some numerical tests (including image recovery examples) to illustrate the efficiency of the proposed method.

AMS subject classifications: 65F22, 65K05

Key words: Coordinate descent, global convergence, linear convergence rate.

1. Introduction

We consider a nonsmooth optimization problem of minimizing the sum of a smooth function and a convex nonseparable function as follows.

$$\min_x F_c(x) \overset{\text{def}}{=} f(x) + cP(x), \tag{1.1}$$

where $c > 0$, $P : \mathbb{R}^n \to (-\infty, \infty]$ is proper, convex, lower semicontinuous (lsc) function, and f is smooth (i.e., continuously differentiable) on an open subset of $\mathbb{R}^n$ containing $\mathrm{dom}P = \{x | P(x) < \infty\}$. In this paper, we assume that P is a nonseparable function in the form $P(x) := \|Lx\|_1$, where $L \neq I$ is preferred to be a sparse matrix. In particular, we focus on a special case of (1.1) defined by

$$\min_x F_c(x) \overset{\text{def}}{=} f(x) + c\|Lx\|_1, \tag{1.2}$$

*Corresponding author. *Email addresses:* `zjbai@xmu.edu.cn` (Z.-J. Bai), `mng@math.hkbu.edu.hk` (M. K. Ng), `maqilq@polyu.edu.hk` (L. Qi) where L is the first order or second order differentiation matrix. Problem (1.1) with $P(x) = \|x\|_1$ and Problem (1.2) arise in many applications, including compressed sensing [9,13,24], signal/image restoration [5,19,23], data mining/classification [3,14,21], and parameter estimation [8,20].

There has been considerable discussion on the problem (1.1), see for instance [2,6,7,11,15]. If P is also smooth, then a coordinate gradient descent based on Armijo-type rule was well developed for the unconditional minimization problem (1.1) in Karmanov [10, pp. 190–196 and pp. 246–250], where the global convergence and geometrical convergence are provided if $F_c(x)$ is assumed to be strongly convex. Recently, Tseng and Yun [22] gave a coordinate gradient descent method with stepsize chosen by an Armijo-type rule for the problem (1.1) under the assumption that P is (block) separable, where the coordinates are updated in either the Gauss-Seidel rule or the Gauss-Southwell-r rule or the Gauss-Southwell-q rule. Moreover, the global convergence and linear convergence for this method were established. However, the method cannot be employed to solve (1.2) directly since $P(x) = \|Lx\|_1$ is no longer a (block) separable function.

Recently, various methods have been considered for image restoration / deblurring/ denoising problems with ℓ_1-regularization, see for instance [5,17,19,23,25]. In particular, Fu *et al.* [5] gave a primal-dual interior point method for solving the following optimization problem with $\ell 1$ regularization:

$$\min_x \|Ax - b\|_2^2 + c\|Lx\|_1, \tag{1.3}$$

where A is a linear blurring operator, x is the original true image, and b is the observed blurred image. In each interior point iteration, the linear system is solved by a preconditioned conjugate gradient method. However, the number of conjugate gradient iterations are still large since the linear system is ill-conditioned and the performance of the preconditioner depends on the support of the blurring function and on how fast such function decays in spatial domain. Osher *et al.* [17,25] presented the Bregman iterative algorithm for solving (1.3) with L being the identity matrix or the first order differentiation matrix. In each Bregman iteration, we need to solve an unconstrained convex subproblem.

In this paper, we aim to provide a coordinate gradient descent method with stepsize chosen by an Armijo-type rule to solve the problem (1.2) and (1.3) efficiently, especially when the problem dimension is large. Our idea is to find a coordinate-wise search direction by finding a minimum in a subproblem, which is obtained by a strictly convex quadratic approximate of f and adding a separable function term (derived from $P(x) = \|Lx\|_1$). Then, we update the current iterate in the direction of the coordinate-wise minimizer. We will show that the coordinate-wise minimizer can be sufficient close to the coordinate-wise minimizer of the subproblem of the sum of the same strictly convex quadratic approximate of f and $P(x) = \|Lx\|_1$ if the parameter c is small enough. This approach can be implemented simply and is capable to solve large-scale problems. We show that our algorithm converges globally if the coordinates are chosen by either the Gauss-Seidel rule or the Gauss-Southwell-r rule or the Gauss-Southwell-q rule. Moreover, we prove that our approach with Gauss-Southwell-q rule converges at least linearly based on a local Lipschitzian error bound assumption. Numerical tests (including image recovery examples) show the efficiency of the proposed method.

Throughout the paper, we use the following notations. $\mathbb{R}^n$ denotes the space of n-dimensional real column vectors, and T denotes transpose. For any $x \in \mathbb{R}^n$ and nonempty $\mathscr{J} \subseteq \mathscr{N} \overset{\text{def}}{=} \{1, \cdots, n\}$, x_j denotes the j-th component of x, and $\|x\|_p = (\sum_{j=1}^n |x_j|^p)^{1/p} = 1$ for $1 \le p \le \infty$. For simplicity, we write $\|x\| = \|x\|_2$. For $n \times n$ real symmetric matrices A and B, we write $A \succeq B$ (respectively, $A \succ B$) to mean that $A - B$ is positive semidefinite (respectively, positive definite). $A_{\mathscr{J}\mathscr{J}} = [A_{ij}]_{i,j \in \mathscr{J}}$ denotes the principal submatrix of A indexed by $\mathscr{J}$. $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ denote the minimum and maximum eigenvalues of A. We denote by I the identity matrix and by 0_n the $n \times n$ matrix of zero entries. Unless otherwise specified, $\{x^k\}$ denotes the sequence x^0, $x^1, \cdots$ and, for any integer $\ell \ge 0$, $\{x^{k+\ell}\}_{\mathscr{K}}$ denotes a subsequence $\{x^{k+\ell}\}_{k \in \mathscr{K}}$ with $\mathscr{K} \subseteq \{0, 1, \cdots\}$.

The paper is organized as follows. In Section 2, we give a coordinate gradient descent method for solving our problem. In Section 3, we establish the convergence of our method. In Section 4 numerical examples are presented to illustrate the efficiency of our proposed method and apply our approach to the image restoration problems in Section 5. Finally, the concluding remarks are given in Section 6.

2. The coordinate gradient descent method

In this section, we present the coordinate gradient descent algorithm for solving the problems (1.2) and (1.3). Since it is assumed that f is smooth, we will replace f by a second-order approximation based on $\nabla f(x)$ at x. Then we generate a search direction d by a coordinate descent. In particular, given a nonempty index subset $\mathscr{J} \in \mathscr{N}$ and a symmetric matrix $H \succ 0_n$ (approximating the Hessian $\nabla^2 f(x)$), we determine a search direction $d = d_H(x; \mathscr{J})$ by

$$d_H(x;\mathscr{J}) \overset{\text{def}}{=} \arg\min_d \left\{ \nabla f(x)^T d + \frac{1}{2} d^T H d + c\|L(x+d)\|_1 \,|\, d_j = 0 \, \forall j \notin \mathscr{J} \right\}. \tag{2.1}$$

Then, we compute the new iterate:

$$x^+ := x + \alpha d,$$

where $\alpha > 0$ is a stepsize. For simplicity, we select the stepsize α by the Armijo rule as in [22].

We point out that $d_H(x;\mathscr{J})$ depends on H only through $H_{\mathscr{J}\mathscr{J}}$. It is still difficult to solve (2.1) since $\|Lx\|_1$ is nonseparable. Therefore, it is desirable to find an approximation of $d_H(x;\mathscr{J})$ via replacing $\|Lx\|_1$ in (2.1) by a separable convex function. In particular, for the problem (1.2), we may approximate $d_H(x;\mathscr{J})$ by

$$\tilde{d}_H(x;\mathscr{J}) \overset{\text{def}}{=} \arg\min_d \left\{ \nabla f(x)^T d + \frac{1}{2} d^T H d + c\|L\|_1 \|x+d\|_1 \,|\, d_j = 0 \, \forall j \notin \mathscr{J} \right\}. \tag{2.2}$$

Remark 2.1. Suppose that $H \succ 0_n$ is a diagonal matrix. It follows from (2.2) that the j-th components of $\tilde{d}_H(x;\mathscr{J})$ is given by

$$\tilde{d}_H(x;\mathscr{J})_j = -\mathrm{mid}\{(\nabla f(x)_j - c\|L\|_1)/H_{jj}, x_j, (\nabla f(x)_j + c\|L\|_1)/H_{jj}\},\ j \in \mathscr{J},$$

where $\mathrm{mid}\{e, f, g\}$ means the median of the scalars e, f, g.

On the closeness between $\tilde{d}_H(x;\mathscr{J})$ and $d_H(x;\mathscr{J})$, we establish the following result.

Proposition 2.1. *Given $x \in \mathbb{R}^n$, nonempty $\mathscr{J} \subseteq \mathscr{N}$ and $H \succ \underline{\lambda} I \succ 0_n$. For the problem* (1.2), *let $d = d_H(x;\mathscr{J})$ and $\tilde{d} = \tilde{d}_H(x;\mathscr{J})$ be the solutions of* (2.1) *and* (2.2), *respectively. Then*

$$\|\tilde{d} - d\| \le \frac{2\sqrt{n}\|L\|_1}{\underline{\lambda}} c. \tag{2.3}$$

Proof. Let $g = \nabla f(x)$. By the definition of d and $\tilde{d}$ and Fermat's rule [18, Theorem 10.1],

$$\begin{aligned} d &\in \arg\min_w (g + Hd)^T w + c\|L(x+w)\|_1, \\ \tilde{d} &\in \arg\min_w (g + H\tilde{d})^T w + c\|L\|_1\|(x+w)\|_1. \end{aligned}$$

Thus

$$\begin{aligned} &(g + Hd)^T d + c\|L(x+d)\|_1 \le (g + Hd)^T \tilde{d} + c\|L(x+\tilde{d})\|_1, \\ &(g + H\tilde{d})^T \tilde{d} + c\|L\|_1\|(x+\tilde{d})\|_1 \le (g + H\tilde{d})^T d + c\|L\|_1\|(x+d)\|_1. \end{aligned}$$

Adding the above two inequalities and simplifying give rise to

$$\begin{aligned} &(\tilde{d} - d)^T H(\tilde{d} - d) \\ &\le c(\|L(x+\tilde{d})\|_1 - \|L(x+d)\|_1) + c\|L\|_1\left(\|(x+d)\|_1 - \|(x+\tilde{d})\|_1\right) \\ &\le c\|L(\tilde{d} - d)\|_1 + c\|L\|_1\|\tilde{d} - d\|_1 \\ &\le 2c\|L\|_1\|\tilde{d} - d\|_1 \le 2c\sqrt{n}\|L\|_1\|\tilde{d} - d\|. \end{aligned}$$

It follows from $H \succ \underline{\lambda} I$ and (2.12) that

$$\underline{\lambda}\|(\tilde{d} - d)\|^2 \le 2c\sqrt{n}\|L\|_1\|\tilde{d} - d\|.$$

Dividing both sides by $\underline{\lambda}\|(\tilde{d} - d)\|$ yields (2.3). □

Remark 2.2. From Proposition 2.1, we see that $\tilde{d}_H(x;\mathscr{J})$ is sufficiently close to $d_H(x;\mathscr{J})$ as the parameter c is small enough. Therefore, in practice, we may replace $d_H(x;\mathscr{J})$ by $\tilde{d}_H(x;\mathscr{J})$ since it is easily computable.

Based on the convexity of the function $\|Lx\|_1$, we have the following descent lemma.

Lemma 2.1. *For any $x \in \mathbb{R}^n$, nonempty $\mathscr{J} \subseteq \mathscr{N}$ and $H \succ 0_n$, let $d = d_H(x;\mathscr{J})$ and $g = \nabla f(x)$. Then*

$$F_c(x+\alpha d) \le F_c(x) + \alpha\left(g^T d + c\|L(x+d)\|_1 - c\|Lx\|_1\right) + o(\alpha) \quad \forall \alpha \in (0,1], \tag{2.4}$$
$$g^T d + c\|L(x+d)\|_1 - c\|Lx\|_1 \le -d^T H d. \tag{2.5}$$

Proof. It follows from a similar proof in [22, Lemma 2.1]. □

Next, we state the coordinate gradient descent (CGD) method for solving (1.2) as follows:

Algorithm 2.1. (CGD method)

Step 0. Give $x^0 \in \mathbb{R}^n$. Let $k := 0$.

Step 1. Choose a nonempty $\mathscr{J}^k \subseteq \mathscr{N}$ and an $H^k \succ 0_n$.

Step 2. Solve (2.1) for $d^k = d_{H^k}(x^k;\mathscr{J}^k)$ where $x = x^k, \mathscr{J} = \mathscr{J}^k, H = H^k$.

Step 3. Choose $\alpha^k_{\text{init}} > 0$ and let α^k be the largest element of $\{\alpha^k_{\text{init}}\beta^j\}_{j=0,1,\cdots}$ satisfying

$$F_c(x^k + \alpha^k d^k) \le F_c(x^k) + \alpha^k \theta \Delta^k, \tag{2.6}$$

where $0 < \beta < 1, 0 < \theta < 1, 0 \le \gamma < 1$, and

$$\Delta^k \stackrel{\text{def}}{=} \nabla f(x^k)^T d^k + \gamma {d^k}^T H^k d^k + c\|L(x^k + d^k)\|_1 - c\|Lx^k\|_1. \tag{2.7}$$

Step 4. Define

$$x^{k+1} := x^k + \alpha^k d^k.$$

Then replace k by $k+1$ and go to Step 1.

In Algorithm 2.1, we need to choose an appropriate index subset $\mathscr{J}^k$. As in [22], we may choose the index subset $\mathscr{J}^k$ in a Gauss-Seidel manner. Based on the definition of L in (1.2), let $\mathscr{J}^0, \mathscr{J}^1, \cdots$ cover $1, 2, \cdots, n$ for every s consecutive iterations, i.e.,

$$\mathscr{J}^k \cup \mathscr{J}^{k+1} \cup \cdots \cup \mathscr{J}^{k+s-1} = \{1,2,\cdots,n\}, \quad k = 0,1,\cdots,$$

where $\mathscr{J}^k$ includes the linear indices corresponding to the nonzero entries of the row vectors of the matrix L. We may also choose the index subset $\mathscr{J}^k$ in a Gauss-Southwell-r rule. That is, we select $\mathscr{J}^k$ such that

$$\|d_{D^k}(x^k;\mathscr{J}^k)\|_\infty \ge v\|d_{D^k}(x^k)\|_\infty,$$

where $0 < v \le 1$, $D^k \succ 0_n$ is diagonal, and $d_D(x) \stackrel{\text{def}}{=} d_D(x;\mathscr{N})$. Finally, we can use the Gauss-Southwell-q rule to choose the index subset $\mathscr{J}^k$, i.e.,

$$q_{D^k}(x^k;\mathscr{J}^k) \le v\, q_{D^k}(x^k), \tag{2.8}$$

where $0 < \nu \le 1$, $D^k \succ 0_n$ is diagonal and

$$q_D(x;\mathcal{J}) \overset{\text{def}}{=} \left(\nabla f(x)^T d + \frac{1}{2} d^T D d + c\|L(x+d)\|_1 - c\|Lx\|_1\right)_{d=d_D(x;\mathcal{J})}. \tag{2.9}$$

Then $q_D(x;\mathcal{J})$ gives an estimate for the descent in F_c from x to $x + d_D(x;\mathcal{J})$. By Lemma 2.1, we have that

$$q_D(x) \overset{\text{def}}{=} q_D(x;\mathcal{N}) \le -\frac{1}{2} d_D(x)^T D d_D(x) \le 0.$$

Remark 2.3. We observe that it is still difficult to find the index subset $\mathcal{J}^k$ satisfying the condition (2.8) since $\|Lx\|_1$ is not separable. From Proposition 2.1, we know that $\tilde{d}_D(x;\mathcal{J}) \to d_D(x;\mathcal{J})$ as $c \to 0$. Therefore, for the small $c > 0$, we can choose the index subset $\mathcal{J}^k$ such that

$$\tilde{q}_D(x^k;\mathcal{J}^k) \le \mu \tilde{q}_D(x^k) \quad \text{for some} \quad 0 < \mu \le 1, \tag{2.10}$$

where

$$\tilde{q}_D(x;\mathcal{J}) \overset{\text{def}}{=} \left(\nabla f(x)^T d + \frac{1}{2} d^T D d + c\|L\|_1(\|x+d\|_1 - \|x\|_1)\right)_{d=\tilde{d}_D(x;\mathcal{J})} \tag{2.11}$$

and then check whether the condition (2.8) holds for the selected $\mathcal{J}^k$ and $d^k = \tilde{d}_{D^k}(x^k;\mathcal{J}^k)$. Then the CGD method with the Gauss-Southwell-r rule is very simple and can be used for large-scale applications in signal/image restoration, etc. From the numerical experiments in Sections 4 and 5, we can observe that it is very efficient to do so when the parameter c come near to $zero$ (but not necessarily too small).

Remark 2.4. In Step 2 of Algorithm 2.1, we need to solve (2.1) for $d^k = d_{H^k}(x^k;\mathcal{J}^k)$. By Proposition 2.1, we may approximate $d_{H^k}(x^k;\mathcal{J}^k)$ by $\tilde{d}_{H^k}(x^k;\mathcal{J}^k)$ defined in (2.2) if the parameter c is sufficiently small. The solution $\tilde{d}_{H^k}(x^k;\mathcal{J}^k)$ to (2.2) has an explicit expression if $H^k \succ 0_n$ is diagonal, see Remark 2.1. From the numerical tests in Sections 4 and 5, one can see that $\tilde{d}_{H^k}(x^k;\mathcal{J}^k)$ is an effective approximation to $d_{H^k}(x^k;\mathcal{J}^k)$ when c is as small as practice-acceptable.

Finally, in Step 3 of Algorithm 2.1, we use the Armijo rule for choosing the stepsize α^k. In this step, we need only function evaluations. In practice, we can keep the number of function evaluations small if α^k_{init} is chosen based on the previous stepsize α^{k-1}.

We will see that all the above three rules yield global convergence of the CGD method for the problem (1.1). We will also show that the CGD method with the Gauss-Southwell-q rule gives rise to at least a linear convergence rate under a local Lipschitizian error bound assumption.

2.1. Properties of search direction

In this section, we shall discuss the properties of the search direction $d_H(x;\mathcal{J})$. These properties can be employed for the proof of global convergence and local convergence rate of the CGD method.

On the sensitivity of $d_H(x;\mathcal{J})$ with respect to the quadratic coefficients H, we have the following lemma. Since the proof is similar to that of Lemma 3 in [22], we omit it.

Lemma 2.2. *For any $x\in\mathbb{R}^n$, nonempty $\mathcal{J}\subseteq\mathcal{N}$, and $H\succ 0_n$, $\widetilde{H}\succ 0_n$, let $d=d_H(x;\mathcal{J})$ and $\tilde{d}=d_{\widetilde{H}}(x;\mathcal{J})$. Then*

$$\|\tilde{d}\|\le\frac{1}{2}\left(1+\lambda_{\max}(R)+\sqrt{1-2\lambda_{\min}(R)+\lambda_{\max}(R)^2}\right)\frac{\lambda_{\max}(H_{\mathcal{J}\mathcal{J}})}{\lambda_{\min}(\widetilde{H}_{\mathcal{J}\mathcal{J}})}\|d\|,$$

where $R=H_{\mathcal{J}\mathcal{J}}^{-1/2}\widetilde{H}_{\mathcal{J}\mathcal{J}}H_{\mathcal{J}\mathcal{J}}^{-1/2}$. If $H_{\mathcal{J}\mathcal{J}}\succ\widetilde{H}_{\mathcal{J}\mathcal{J}}$, then also

$$\|d\|\le\sqrt{\frac{\lambda_{\max}(H_{\mathcal{J}\mathcal{J}}-\widetilde{H}_{\mathcal{J}\mathcal{J}})}{\lambda_{\min}(H_{\mathcal{J}\mathcal{J}}-\widetilde{H}_{\mathcal{J}\mathcal{J}})}}\|\tilde{d}\|.$$

The following result concerns stepsizes satisfying the Armijo descent condition (2.6), which can be proved by following the similar proof lines of [22, Lemma 5 (b)].

Lemma 2.3. *For any $x\in\mathbb{R}^n$, $H\succ 0_n$, and nonempty $\mathcal{J}\subseteq\mathcal{N}$, let $d=d_H(x;\mathcal{J})$ and $g=\nabla f(x)$. For any $\gamma\in[0,1)$, let $\Delta=g^Td+\gamma d^THd+c\|L(x+d)\|_1-c\|Lx\|_1$. If f satisfies*

$$\|\nabla f(y)-\nabla f(z)\|\le\zeta\|y-z\|\quad\forall\, y,z\in\mathbb{R}^n \tag{2.12}$$

for some $\zeta>0$, and $H\succeq\underline{\lambda}I$, where $\underline{\lambda}>0$, then the descent condition

$$F_c(x+\alpha d)-F_c(x)\le\theta\alpha\Delta$$

is satisfied for any $\theta\in(0,1)$ whenever $0\le\alpha\le\min\{1,2\underline{\lambda}(1-\theta+\theta\gamma)/\zeta\}$.

3. Convergence analysis

In this section, we establish the global and linear convergence of Algorithm 2.1. As in [22], we know that $x\in\mathbb{R}^n$ is called a stationary point of F_c if $x\in\mathrm{dom}F_c$ and $F_c'(x;d)\ge 0$ for all $d\in\mathbb{R}^n$.

Assumption 3.1. *$\underline{\lambda}I\le H^k\le\bar{\lambda}I$ for all k, where $0<\underline{\lambda}\le\bar{\lambda}$.*

By following similar arguments in [22, Theorem 1], we can show the following global convergence of Algorithm 2.1.

Lemma 3.1. *Let $\{x^k\}$, $\{d^k\}$, $\{H^k\}$ be sequences generated by the CGD method under Assumption 3.1, where $\inf_k\alpha^k_{\text{init}}>0$. Then the following results hold.*

(a) *$\{F_c(x^k)\}$ is nonincreasing and Δ^k given by (2.7) satisfies*

$$-\Delta^k\ge(1-\gamma)d^{k^T}H^kd^k\ge(1-\gamma)\underline{\lambda}\|d^k\|^2\quad\forall k, \tag{3.1}$$

$$F_c(x^{k+1})-F_c(x^k)\le\theta\alpha^k\Delta^k\le 0\quad\forall k. \tag{3.2}$$

(b) *If $\{x^k\}_{\mathscr{K}}$ is a convergent subsequence of $\{x^k\}$, then $\{\alpha^k\Delta^k\} \to 0$ and $\{d^k\}_{\mathscr{K}} \to 0$. If in addition $\underline{\delta} I \le D^k \le \bar{\delta} I$ for all k, where $0 < \underline{\delta} \le \bar{\delta}$, then $\{d_{D^k}(x^k;\mathscr{J}^k)\}_{\mathscr{K}} \to 0$.*

(c) *If $\{\mathscr{J}^k\}$ is chosen by the Gauss-Southwell-q rule* (2.8)*, $\underline{\delta} I \le D^k \le \bar{\delta} I$ for all k, where $0 < \underline{\delta} \le \bar{\delta}$, and either* (1) *$\|Lx\|_1$ is continuous on $\mathbb{R}^n$ or* (2) *$\inf_k \alpha^k > 0$ or* (3) *$\alpha^k_{\text{init}} = 1$ for all k, then every cluster point of $\{x^k\}$ is a stationary point of F_c.*

(d) *If f satisfies* (2.12) *for some $\zeta \ge 0$, then $\inf_k \alpha^k > 0$. If $\lim_{k\to\infty} F_c(x^k) > -\infty$ furthermore, then $\{\Delta^k\} \to 0$ and $\{d^k\} \to 0$.*

Now we establish the convergence rate of the CGD method for $\{\mathscr{J}^k\}$ chosen by the Gauss-Southwell-q rule (2.8). We need the following assumption as in [22]. In the following, $\bar{X}$ denotes the set of stationary points of F_c and

$$\text{dist}(x,\bar{X}) = \min_{\bar{x}\in\bar{X}} \|x - \bar{x}\| \quad \forall x \in \mathbb{R}^n.$$

Assumption 3.2. (a) *$\bar{X} \neq \emptyset$ and, for any $\xi \ge \min_x F_c(x)$, there exist scalars $\tau > 0$ and $\epsilon > 0$ such that*

$$\text{dist}(x,\bar{X}) \le \tau \|d_I(x)\| \quad \textit{whenever } F_c(x) \le \xi, \|d_I(x)\| \le \epsilon.$$

(b) *There exists a scalar $\delta > 0$ such that*

$$\|x - y\| \ge \delta \quad \textit{whenever } x, y \in \bar{X}, F_c(x) \neq F_c(y).$$

On the asymptotic convergence rate of the CGD method under Assumption 3.2, we have the following theorem. The proof technique is taken from [22] but noting the non-separability of $\|Lx\|_1$.

Theorem 3.1. *Assume that f satisfies* (2.12) *for some $\zeta \ge 0$. Let $\{x^k\}$, $\{H^k\}$, $\{d^k\}$ be sequences generated by the CGD method satisfying Assumption* 3.1*, where $\{\mathscr{J}^k\}$ is chosen by Gauss-Southwell-q rule* (2.8) *and $\underline{\delta} I \le D^k \le \bar{\delta} I$ for all k ($0 \le \underline{\delta} \le \bar{\delta}$). If F_c satisfies Assumption* 3.2 *and $\sup_k \alpha^k_{\text{init}} \le 1$ and $\inf_k \alpha^k_{\text{init}} > 0$, then either $\{F_c(x^k)\} \downarrow -\infty$ or $\{F_c(x^k)\}$ converges at least Q-linearly and $\{x^k\}$ converges at least R-linearly.*

Proof. For each $k = 0, 1, \cdots$ and $d^k = d_{H^k}(x^k;\mathscr{J}^k)$, by (2.7)-(2.9), we have

$$\begin{aligned}
\Delta^k + \left(\frac{1}{2} - \gamma\right) {d^k}^T H^k d^k &= {g^k}^T d^k + \frac{1}{2}{d^k}^T H^k d^k + c\|L(x^k + d^k)\|_1 - c\|Lx^k\|_1 \\
&\le {g^k}^T \tilde{d}^k + \frac{1}{2}(\tilde{d}^k)^T H^k \tilde{d}^k + c\|L(x^k + \tilde{d}^k)\|_1 - c\|Lx^k\|_1 \\
&\le q_{D^k}(x^k;\mathscr{J}^k) + \frac{1}{2}(\tilde{d}^k)^T (H^k - D^k)\tilde{d}^k \\
&\le q_{D^k}(x^k;\mathscr{J}^k) + (\bar{\lambda} - \underline{\delta})\|\tilde{d}^k\|^2, \qquad (3.3)
\end{aligned}$$

where $\tilde{d}^k := d_{D^k}(x^k; \mathcal{J}^k)$. For each k,

$$\frac{\underline{\delta}}{\bar{\lambda}} I \preceq \underline{\delta}(H^k_{\mathcal{J}^k\mathcal{J}^k})^{-1} \preceq (H^k_{\mathcal{J}^k\mathcal{J}^k})^{-1/2} D^k_{\mathcal{J}^k\mathcal{J}^k} (H^k_{\mathcal{J}^k\mathcal{J}^k})^{-1/2} \preceq \bar{\delta}(H^k_{\mathcal{J}^k\mathcal{J}^k})^{-1} \preceq \frac{\bar{\delta}}{\underline{\lambda}} I.$$

Then, by Lemma 2.2, we obtain

$$\|\tilde{d}^k\| \le \frac{1 + \bar{\delta}/\underline{\lambda} + \sqrt{1 - 2\underline{\delta}/\bar{\lambda} + (\bar{\delta}/\underline{\lambda})^2}}{2} \frac{\bar{\lambda}}{\underline{\delta}} \|d^k\|.$$

This, together with (3.3), implies that

$$\Delta^k + \left(\frac{1}{2} - \gamma\right) d^{k^T} H^k d^k \le q_{D^k}(x^k; \mathcal{J}^k) + \omega \|d^k\|^2, \tag{3.4}$$

where ω is a constant depending on $\bar{\lambda}$, $\underline{\lambda}$, $\bar{\delta}$, and $\underline{\delta}$.

By Lemma 3.1 (a), $\{F_c(x^k)\}$ is nonincreasing. Then either $\{F_c(x^k)\} \downarrow -\infty$ or $\lim_{k\to\infty} F_c(x^k) > -\infty$. Suppose the latter. Since $\inf_k \alpha^k_{\text{init}} > 0$, Lemma 3.1 (d) implies that $\inf_k \alpha^k > 0$, $\{\Delta^k\} \to 0$, and $\{d^k\} \to 0$. Since $\{H^k\}$ is bounded by Assumption 3.1, by (3.4), we get $\lim_{k\to\infty} \inf q_{D^k}(x^k; \mathcal{J}^k) \ge 0$. By (2.9) and (2.5) in Lemma 2.1, we have that

$$q_{D^k}(x^k) \le -\frac{1}{2} d_{D^k}(x^k)^T D^k d_{D^k}(x^k) \le 0.$$

Also, using (2.8), we obtain $\{d_{D^k}(x^k)\} \to 0$.

By Lemma 2.2 with $H = D^k$ and $\tilde{H} = I$,

$$\|d_I(x^k)\| \le \frac{1 + 1/\underline{\delta} + \sqrt{1 - 2/\bar{\delta} + (1/\underline{\delta})^2}}{2} \bar{\delta} \|d_{D^k}(x^k)\| \quad \forall k. \tag{3.5}$$

Thus $\{d_I(x^k)\} \to 0$. Since $\{F_c(x^k)\}$ is nonincreasing, it follows that $F_c(x^k) \le F_c(x^0)$ and $\|d_I(x^k)\| \le \epsilon$ for all $k \ge$ some $\bar{k}$. Then, by Assumption 3.2 (a), we get

$$\|x^k - \bar{x}^k\| \le \tau \|d_I(x^k)\| \quad \forall k \ge \bar{k}, \tag{3.6}$$

where $\tau > 0$ and $\bar{x}^k \in \bar{X}$ satisfies $\|x^k - \bar{x}^k| = \text{dist}(x^k, \bar{X})$. Since $\{d_I(x^k)\} \to 0$, this implies that $\{x^k - \bar{x}^k\} \to 0$. Since $\{x^{k+1} - x^k\} = \{\alpha^k d^k\} \to 0$, this and Assumption 3.2 (b) imply that $\{\bar{x}^k\}$ eventually settles down at some isocost surface of F_c, i.e., there exist an index $\hat{k} \ge \bar{k}$ and a scalar $\bar{v}$ such that

$$F_c(\bar{x}^k) = \bar{v} \quad \forall k \ge \hat{k}. \tag{3.7}$$

Fix any index k with $k \ge \hat{k}$. Since $\bar{x}^k$ is a stationary point of F_c, we obtain

$$\nabla f(\bar{x}^k)^T (x^k - \bar{x}^k) + c\|Lx^k\|_1 - c\|L\bar{x}^k\|_1 \ge 0.$$

By the Mean Value Theorem, there exists some ψ^k lying on the line segment joining x^k with $\bar{x}^k$ such that

$$f(x^k) - f(\bar{x}^k) = \nabla f(\psi^k)^T (x^k - \bar{x}^k).$$

Since $x^k, \bar{x}^k \in \mathbb{R}^n$, so does ψ^k. Combing these two relations and using (3.7), we get

$$\begin{aligned}\bar{v} - F_c(x^k) &\le \left(\nabla f(\bar{x}^k) - \nabla f(\psi^k)\right)^T (x^k - \bar{x}^k)\\ &\le \|\nabla f(\bar{x}^k) - \nabla f(\psi^k)\| \, \|x^k - \bar{x}^k\|\\ &\le \zeta \|x^k - \bar{x}^k\|^2,\end{aligned}$$

where the last inequality uses (2.12), the convexity of $\mathbb{R}^n$, and $\|\psi^k - \bar{x}^k\| \le \|x^k - \bar{x}^k\|$. This together with $\{x^k - \bar{x}^k\} \to 0$ shows that

$$\liminf_{k\to\infty} F_c(x^k) \ge \bar{v}. \tag{3.8}$$

Fix any $k \ge \hat{k}$. Letting $\mathscr{J} = \mathscr{J}^k$, we have

$$\begin{aligned}&F_c(x^{k+1}) - \bar{v}\\ &= f(x^{k+1}) + c\|L(x^{k+1})\|_1 - f(\bar{x}^k) - c\|L\bar{x}^k\|_1\\ &= \nabla f(\tilde{x}^k)^T(x^{k+1} - \bar{x}^k) + c\|L(x^{k+1})\|_1 - c\|L\bar{x}^k\|_1\\ &= (\nabla f(\tilde{x}^k) - g^k)^T(x^{k+1} - \bar{x}^k) - (D^k d^k)^T(x^{k+1} - \bar{x}^k) + \gamma_k,\end{aligned}$$

where the Mean Value Theorem with $\tilde{x}^k$ a point lying on the segment joining x^{k+1} with $\bar{x}^k$, and

$$\begin{aligned}\gamma_k &= (g^k + D^k d^k)^T(x^{k+1} - \bar{x}^k) + c\|L(x^{k+1})\|_1 - c\|L\bar{x}^k\|_1\\ &= (g^k + D^k d^k)^T(x^k - \bar{x}^k) + \alpha^k (g^k + D^k d^k)^T d^k + c\|L(x^{k+1})\|_1 - c\|L\bar{x}^k\|_1\\ &\le (g^k + D^k d^k)^T(x^k - \bar{x}^k) + \alpha^k c\|Lx^k\|_1 - \alpha^k c\|L(x^k + d^k)\|_1\\ &\quad + c\|L(x^{k+1})\|_1 - c\|L\bar{x}^k\|_1\\ &= (g^k + D^k d^k)^T(x^k - \bar{x}^k) + \alpha^k c\|Lx^k\|_1 - \alpha^k c\|L(x^k + d^k)\|_1\\ &\quad + c\left\|L\left(\alpha^k(x^k + d^k) + (1-\alpha^k)x^k\right)\right\|_1 - c\|L\bar{x}^k\|_1\\ &\le (g^k + D^k d^k)^T(x^k - \bar{x}^k) + \alpha^k c\|Lx^k\|_1 - \alpha^k c\|L(x^k + d^k)\|_1\\ &\quad + \alpha^k c\|L(x^k + d^k)\|_1 + (1-\alpha^k)c\|Lx^k\|_1 - c\|L\bar{x}^k\|_1\\ &= (g^k + D^k d^k)^T(x^k - \bar{x}^k) + c\|Lx^k\|_1 - c\|L\bar{x}^k\|_1\\ &= (D^k d^k)^T(x^k - \bar{x}^k) - \frac{1}{2}\left(D^k d_{D^k}(x^k)\right)^T(x^k - \bar{x}^k)\\ &\quad + \left(g^k + \frac{1}{2}D^k d_{D^k}(x^k)\right)^T(x^k - \bar{x}^k) - c\|L\bar{x}^k\|_1 + c\|Lx^k\|_1.\end{aligned}$$

In the second step above, we used $x^{k+1} - \bar{x}^k = x^k - \bar{x}^k + \alpha^k d^k$; the third step uses (2.5) in Lemma 2.1 (applied to x^k, D^k, and $\mathscr{J}^k$); the fifth step uses the convexity of $\|Lx\|_1$, $\alpha^k \le \alpha^k_{\text{init}} \le 1$.

Combining the above results, we obtain

$$\begin{aligned} & F_c(x^{k+1}) - \bar{v} \\ & \le \zeta \|\tilde{x}^k - x^k\| \|x^{k+1} - \bar{x}^k\| + \bar{\delta}\|d^k\|\|x^{k+1} - \bar{x}^k\| + \bar{\delta}\|d^k\|\|x^k - \bar{x}^k\| \\ & \quad + \frac{1}{2}\bar{\delta}\|d_{D^k}(x^k)\|\|x^k - \bar{x}^k\| - q_{D^k}(x^k), \end{aligned} \tag{3.9}$$

where (2.12), $0_n \prec D^k \preceq \bar{\delta} I$ and (2.9) have been used.

By the inequalities $\|\tilde{x}^k - x^k\| \le \|x^{k+1} - x^k\| + \|x^k - \bar{x}^k\|$, $\|x^{k+1} - \bar{x}^k\| \le \|x^{k+1} - x^k\| + \|x^k - \bar{x}^k\|$, and $\|x^{k+1} - x^k\| = \alpha^k \|d^k\|$, we observe from (3.6) and $D^k \succ 0_n$ that the right-hand side of (3.9) is bounded above by

$$C_1 \left(\|d^k\| + \|d_{D^k}(x^k)\| + \|d_I(x^k)\| \right)^2 - q_{D^k}(x^k) \tag{3.10}$$

for all $k \ge \hat{k}$, where $C_1 > 0$ is some constant depending on ζ, τ, $\bar{\delta}$ only. By (3.5), the quantity in (3.10) is bounded above by

$$C_2 \|d^k\|^2 + C_2 \|d_{D^k}(x^k)\|^2 - q_{D^k}(x^k) \tag{3.11}$$

for all $k \ge \hat{k}$, where $C_2 > 0$ is some constant depending on ζ, τ, $\underline{\delta}$, $\bar{\delta}$ only.

By (3.1), we have

$$\underline{\lambda}\|d^k\|^2 \le {d^k}^T H^k d^k \le -\frac{1}{1-\gamma}\Delta^k \quad \forall k. \tag{3.12}$$

By (2.9) and $D^k \succeq \underline{\delta} I$,

$$q_{D^k}(x^k) \le -\frac{1}{2} d_{D^k}(x^k)^T D^k d_{D^k}(x^k) \le -\frac{\underline{\delta}}{2}\|d_{D^k}(x^k)\|^2 \quad \forall k.$$

Thus, the quantity in (3.11) is bounded above by

$$C_3 \left(-\Delta^k - q_{D^k}(x^k) \right) \tag{3.13}$$

for all $k \ge \hat{k}$, where $C_3 > 0$ is some constant depending on ζ, τ, $\underline{\lambda}$, $\underline{\delta}$, $\bar{\delta}$, γ only.

By (2.8), we have

$$q_{D^k}(x^k; \mathscr{J}^k) \le \nu q_{D^k}(x^k). \tag{3.14}$$

By (3.4) and (3.12),

$$\begin{aligned} -q_{D^k}(x^k; \mathscr{J}^k) & \le -\Delta^k + \left(\gamma - \frac{1}{2} \right) {d^k}^T H^k d^k + \omega \|d^k\|^2 \\ & \le -\Delta^k - \max\left\{ 0, \gamma - \frac{1}{2} \right\} \frac{1}{1-\gamma}\Delta^k - \frac{\omega}{\underline{\lambda}(1-\gamma)}\Delta^k. \end{aligned} \tag{3.15}$$

Combining (3.14) and (3.15), the quantity in (3.13) is bounded above by

$$-C_4 \Delta^k \tag{3.16}$$

for all $k \geq \hat{k}$, where $C_4 > 0$ is some constant depending on ζ, τ, $\underline{\lambda}$, $\bar{\lambda}$, $\underline{\delta}$, $\bar{\bar{\delta}}$, γ, v only. Therefore, the right-hand side of (3.9) is bounded above by $-C_4\Delta^k$ for all $k \geq \hat{k}$. This, together with (3.2), (3.9), and $\inf_k \alpha^k > 0$ yields

$$F_c(x^{k+1} - \bar{v}) \leq C_5(F_c(x^k) - F_c(x^{k+1})) \quad \forall k \geq \hat{k},$$

where $C_5 = C_4/(\theta \inf_k \alpha^k)$. Upon rearranging terms and using (3.8), we get

$$0 \leq F_c(x^{k+1}) - \bar{v} \leq \frac{C_5}{1+C_5}(F_c(x^k) - \bar{v}) \quad \forall k \geq \hat{k},$$

and so $\{F_c(x^k)\}$ converges to $\bar{v}$ at least Q-linearly.

Finally, by (3.2), (3.12), and $x^{k+1} - x^k = \alpha^k d^k$, we obtain

$$\theta(1-\gamma)\underline{\lambda}\frac{\|x^{k+1} - x^k\|^2}{\alpha^k} \leq F_c(x^k) - F_c(x^{k+1}) \quad \forall k \geq \hat{k}.$$

It follows that

$$\|x^{k+1} - x^k\| \leq \sqrt{\frac{\sup_k \alpha^k}{\theta(1-\gamma)\underline{\lambda}}(F_c(x^k) - F_c(x^{k+1}))} \quad \forall k \geq \hat{k}.$$

Since $\{F_c(x^k) - F_c(x^{k+1})\} \to 0$ at least R-linearly and $\sup_k \alpha^k \leq 1$, $\{x^k\}$ converges at least R-linearly. □

4. Numerical results

In this section, we will present some numerical tests to illustrate the efficiency of Algorithm 2.1. All runs are carried out in MATLAB 7.6 running on a notebook PC of 2.5 GHz Intel(R) Core(TM)2 Duo CPU. We implement the CGD Algorithm 2.1 to solve the following minimization problems with ℓ_1-regularization:

$$\min_x F_c(x) \overset{\text{def}}{=} f(x) + c\|Lx\|_1, \tag{4.1}$$

where the matrix L is given by

$$L = \begin{bmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & \ddots & \\ & & & 1 & -1 \end{bmatrix} \in \mathbb{R}^{(n-1)\times n} \tag{4.2}$$

or

$$L = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix} \in \mathbb{R}^{n\times n}. \tag{4.3}$$

Table 1: Nonlinear least square test functions.

Name	n	Description
BT	1000	Broyden tridiagonal function, nonconvex, with sparse Hessian
BAL	1000	Brown almost-linear function, nonconvex, with dense Hessian
TRIG	1000	Trigonometric function, nonconvex, with dense Hessian
LR1	1000	$f(x)=\sum_{i=1}^{n}\left(i\left(\sum_{j=1}^{n} jx_j\right)-1\right)^2$, convex, with dense Hessian
LFR	1000	$f(x)=\sum_{i=1}^{n}\left(x_i-\frac{2}{n+1}\sum_{j=1}^{n}x_j-1\right)^2+\left(\frac{2}{n+1}\sum_{j=1}^{n}x_j+1\right)^2$, strongly convex, with dense Hessian

For the function f in (4.1), we choose several nonconvex and convex test functions with $n=1000$ from the set of nonlinear least square functions used by Moré *et al.* [16]. These functions are shown in Table 1.

In our experiments, we choose the diagonal Hessian approximation

$$H^k=\operatorname{diag}\left[\min\{\max\{\nabla^2 f(x^k)_{jj},10^{-2}\},10^9\}\right]_{j=1,\cdots,n},$$

for Problem (4.1). When $\nabla^2 f$ is ill-conditioned, the Armijo descent condition (2.6) eventually cannot be satisfied by any $\alpha^k>0$ due to cancelation error in the function evaluations. Also, in MATLAB 7.6, floating point substraction is accurate up to 15 digits only. In these cases, no further progress is possible so we exit the method when (2.6) remains unsatisfied after α^k reaches 10^{-15}.

To use the CGD approach to solve our problems, we need to solve (2.1), which is not separable and then poses the computational challenges, especially for large-scale cases. By Remark 2.4, $\tilde{d}_H(x;\mathcal{J})$ defined in (2.2) is a good approximation to $d_H(x;\mathcal{J})$ defined in (2.1) if c is close to zero. Thus we can replace $d_H(x;\mathcal{J})$ by $\tilde{d}_H(x;\mathcal{J})$ in practice. In addition, since the Hessian approximation $H\succ 0_n$ is chosen to be diagonal, $\tilde{d}_H(x;\mathcal{J})$ is given explicitly, see Remark 2.1.

On the other hand, the index subset $\mathcal{J}^k$ is chosen by either (i) the Gauss-Seidel rule, where J^k cycles through $\{1,2\}$, $\{2,3\}$,$\cdots$,$\{n-1,n\}$ in that order for B as in (4.2) (for B as in (4.3), J^k cycles through $\{1,2,3\}$, $\{2,3,4\}$,$\cdots$,$\{n-2,n-1,n\}$) or (ii) the Gauss-Southwell-r rule ($D^k=H^k$)

$$\mathcal{J}^k=\left\{j\,\middle|\,|\tilde{d}_{D^k}(x^k;j)|\geq v^k\|\tilde{d}_{D^k}(x^k)\|_\infty\right\},\quad v^{k+1}=\begin{cases}\max\{10^{-4},v^k/10\} & \text{if } \alpha^k>10^{-3},\\ \min\{0.9,50v^k\} & \text{if } \alpha^k<10^{-6},\\ v^k & \text{else}\end{cases}\tag{4.4}$$

(initially $v^0=0.5$) or (iii) the Gauss-Southwell-q rule

$$\mathcal{J}^k=\left\{j\,\middle|\,\tilde{q}_{D^k}(x^k;j)\leq v^k\min_i \tilde{q}_{D^k}(x^k;i)\right\},\quad v^{k+1}=\begin{cases}\max\{10^{-4},v^k/10\} & \text{if } \alpha^k>10^{-3},\\ \min\{0.9,50v^k\} & \text{if } \alpha^k<10^{-6},\\ v^k & \text{else}\end{cases}\tag{4.5}$$

Table 2: Numerical results for the test functions ("*" means that CGD exited due to an Armijo stepsize reaching 10^{-15} and "‡" means that $v^{k+1} = \max\{10^{-5}, v^k/10\}$ if $\alpha^k \geq 10^{-3}$).

L is set to be (4.2)				
Name	c	Rule (i)	Rule (ii)	Rule (iii)
		it./obj./cpu.	it./obj./cpu.	it./obj./cpu.
BT	1	3991/11.5906/4.8	1/0.74036/0.08	‡8/0.73116/0.1
	0.1	6986/11.1572/8.0	1/0.04722/0.1	‡16/0.04722/0.1
	0.01	6986/13.5325/8.0	1/0.00433/0.1	‡25/0.00433/0.1
	0.001	6986/13.8458/8.0	1/0.00043/0.2	‡36/0.00043/0.2
BAL	1	5/122887/0.6	5/0.00135/0.5	5/0.00135/0.5
	0.1	5/122659/0.5	5/0.00018/0.4	5/0.00018/0.4
	0.01	5/122637/0.5	5/0.00001/0.5	5/0.00001/0.4
	0.001	5/122634/0.6	5/0.00000/0.4	5/0.00000/0.4
TRIG	1	*1/0.00008/0.3	1/0.00000/0.3	1/0.00000/0.2
	0.1	*1/0.00008/0.4	1/0.00000/0.3	1/0.00000/0.2
	0.01	*1/0.00008/0.4	1/0.00000/0.2	1/0.00000/0.3
	0.001	*1/0.00008/0.2	1/0.00000/0.3	1/0.00000/0.3
LR1	1	14/224236/0.5	7/978562/0.7	5/254.1/0.5
	0.1	14/22648/0.6	7/978558/0.6	5/251.2/0.6
	0.01	14/2489/0.5	7/6285090.8	5/251.0/0.6
	0.001	14/473/0.5	7/628509/0.7	5/250.9/0.7
LFR	1	999/1005/9.9	1/1001/0.08	1/1001/0.03
	0.1	999/15.2/9.7	1/11/0.05	1/11/0.06
	0.01	999/5.1/9.3	1/1.1/0.03	1/1.1/0.02
	0.001	999/5.0/9.2	1/1.0/0.05	1/1.0/0.03

(initially $\mu^0 = 0.5$) satisfying (2.8) with $d = \tilde{d}(x; \mathscr{J})$ for some $0 < \mu \leq 1$, where $\tilde{q}_D(x; \mathscr{J})$ is defined by (2.11). The Gauss-Southwell-q rule is based on Remark 2.3.

For simplicity, in Algorithm 2.1, we set $\theta = 0.1$, $\beta = 0.5$, $\gamma = 0$, $\alpha^0_{\text{init}} = 1$, and $\alpha^0_{\text{init}} = \min\{\alpha^{k-1}/\beta, 1\}$ for all $k \geq 1$. The stopping tolerance for Algorithm 2.1 is set to be

$$\|x^{k+1} - x^k\| \leq 10^{-4}.$$

Now, we report the performance of Algorithm 2.1 using the rules (i), (ii) and (iii). Tables 2-3 display the number of CGD iterations, the final objective function value, and the CPU time (in seconds) for four different values of c. From Tables 2 and 3, we observe that Rules (ii) and (iii) behaviors better than Rule (i) for the test functions BT, BAL, TRIG, and LFR. However, for the test function LR1 whose Hessian are far from being diagonally dominant, Rules (ii) and (iii) are slower than Rule (i) in CPU time but Rule (iii) performs better than Rules (i) and (ii) in minimizing the objective function value.

5. Application to image restoration

In this section, we apply the CGD algorithm to image restoration examples.

Table 3: Numerical results for the test functions ("*" means that CGD exited due to an Armijo stepsize reaching 10^{-15}.

L is set to be (4.3)				
Name	c	Rule (i)	Rule (ii)	Rule (iii)
		`it./obj./cpu.`	`it./obj./cpu.`	`it./obj./cpu.`
BT	1	1997/11.9552/2.7	*6/4.7574/0.02	*8/3.5911/0.02
	0.1	3994/10.3078/4.8	14/0.10141/0.03	36/0.13934/0.2
	0.01	5990/10.1676/7.3	23/0.00996/0.2	90/0.01118/0.4
	0.001	7973/10.1573/9.1	32/0.00099/0.1	150/0.00101/0.5
BAL	1	16/122307/1.5	6/2.0055/0.6	6/2.0055/0.7
	0.1	17/120820/1.5	5/0.20033/0.5	5/0.20033/0.4
	0.01	17/120671/1.5	5/0.02003/0.5	5/0.02003/0.5
	0.001	17/120657/1.6	5/0.00200/0.5	5/0.00200/0.3
TRIG	1	*1/0.00208/0.4	1/0.00000/0.3	1/0.00000/0.3
	0.1	*1/0.00028/0.4	1/0.00000/0.3	1/0.00000/0.2
	0.01	*1/0.00001/0.4	1/0.00000/0.3	1/0.00000/0.3
	0.001	2/0.00009/0.4	1/0.00000/0.3	1/0.00000/0.3
LR1	1	12/415602/0.4	7/978567/0.8	5/257.6/0.5
	0.1	12/41795/0.4	7/978558/0.7	5/251.6/0.6
	0.01	12/4414/0.4	7/784234/0.7	5/251.0/0.5
	0.001	13/676/0.4	7/628509/0.9	5/250.9/0.5
LFR	1	997/1007/9.5	1/1001/0.1	1/1001/0.1
	0.1	997/45.5/9.5	1/41.2/0.06	1/41.2/0.06
	0.01	997/5.5/9.3	1/1.4/0.02	1/1.4/0.05
	0.001	997/5.0/9.7	1/1.0/0.03	1/1.0/0.03

Example 1. We test the 128×128 Cameraman image (See Fig. 1(a)). The blurring function is given by

$$a(y,z)=\exp[-0.5*(y^2+z^2)].$$

The observed image is represented by the vector of the form of $b=Ax^*+\eta$, where A is a BTTB matrix and x^* is a vector formed by row ordering the original image.

We use Algorithm 2.1 to solve the following minimization problem with ℓ_1-regularization:

$$\min_x F_c(x) \overset{\text{def}}{=} \|Ax-b\|_2^2+c\|Lx\|_1, \tag{5.1}$$

where L is given as in (4.2) or (4.3).

Fu *et al.* [5] provided a primal-dual interior point method for image restoration by solving (5.1). In each interior point iteration, we need to solve a ill-conditioned linear system. Osher *et al.* [17, 25] gave the Bregman iterative algorithm for solving (5.1) with L being the identity matrix or the first order differentiation matrix. In each Bregman iteration, we need to solve an unconstrained convex subproblem. For our algorithm, if we choose a diagonal matrix $H\succ 0_n$ and the parameter c is sufficiently small, then, by Proposition 2.1, the search direction $= d_H(x;\mathcal{J})$ in (2.1) may be approximated by $\tilde{d}_H(x;\mathcal{J})$ defined in (2.2), which is easy to solve, see Remark 2.1. In addition, from Section 4, the index subset $\mathcal{J}$ can

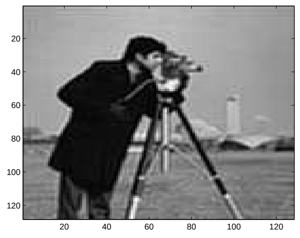
(a) Original Image

(b) ObservedImage

(c) Observed Image

(d) Initial Guess Image

(e) Initial Guess Image


(f) Restored Image

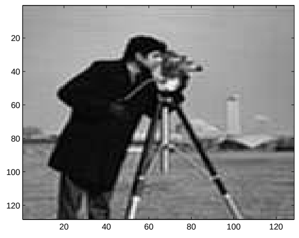

(g) Restored Image

Figure 1: Original, observed, initial guess, and restored images of Cameraman for L as in (4.2) (left) and (4.3) (right).

Table 4: Numerical results for the image restoration ("*" means that the condition (2.8) is not satisfied).

L is set to be (4.2)			
c	Rule (i)	Rule (ii)	Rule (iii)
	it./hd./ obj./ cpu./res0./res*.	it./hd./ obj./ cpu./res0./res*.	it./hd./ obj./cpu./res0./res*.
0.1	81/6.0/734265/131/0.0892/0.0888	170/0.49/19591/266/0.0892/0.0489	*20/3.95/33616/32.9/0.0891/0.0648
0.01	81/6.2/724597/131/0.0891/0.0887	170/0.51/6574/266/0.0892/0.0491	*131/0.65/7405/215/0.0892/0.0513
0.001	81/6.3/723219/131/0.0892/0.0888	169/0.52/5291/275/0.0892/0.0493	171/0.52/5277/278/0.0892/0.0492
0.0001	81/6.2/723265/134/0.0892/0.0888	170/0.54/5282/277/0.0891/0.0492	171/0.53/5133/277/0.0892/0.0494
L is set to be (4.3)			
c	Rule (i)	Rule (ii)	Rule (iii)
	it./hd./ obj./ cpu./res0./res*.	it./hd./ obj./ cpu./res0./res*.	it./hd./ obj./cpu./res0./res*.
0.1	208/7.4/724955/341/0.0891/0.0885	170/0.53/22096/274/0.0891/0.0489	*21/3.81/33322/34.2/0.0891/0.0647
0.01	209/6.0/715157/335/0.0891/0.0885	169/0.52/6743/271/0.0893/0.0492	*136/0.67/7318/221/0.0892/0.0512
0.001	209/6.1/714234/336/0.0891/0.0885	170/0.54/5450/279/0.0891/0.0491	170/0.55/5364/277/0.0892/0.0494
0.0001	209/5.9/714008/335/0.0892/0.0886	170/0.52/5219/271/0.0891/0.0492	171/0.51/5133/281/0.0892/0.0493

be chosen by Rules (i), (ii), or (iii) which is easy to check. Also, the main computational cost of our algorithm lies in the computations of $\nabla f(x)$ and function evaluations $F_c(x)$ (which is needed in the Armijo rule). This requires the matrix-vector products Ax and $A^T x$ which can be computed by fast transforms. In the following, we present some numerical tests to show that our approach is effective for image restoration.

In our tests, the noise η is set to Gaussian white noise with noise-to-signal ratio of 40 dB. We also set the initial guess image x^0 to be the solution of the following linear equation

$$(c_0 I + A^* A)x^0 = A^* b, \tag{5.2}$$

where c_0 is a suitable parameter. For simplicity, we fix $c_0 = 0.05$ and solve the above linear equation by the PCG method with the block-circulant preconditioner as in [12]. In the PCG method, we use the zero vector as the initial point and the stopping criteria is $\|r^i\|_2/\|A^* b\|_2 \le 10^{-7}$, where r^i is the residual after i iterations.

In our experiments, we choose the diagonal Hessian approximation

$$H^k = \mathrm{diag}\left[\min\{\max\{\nabla^2 f(x^k)_{jj}, 10^{-2}\}, 10^9\}\right]_{j=1,\cdots,n},$$

for Problem (5.1) with $f(x) = \|Ax - b\|_2^2$. For simplicity, in Algorithm 2.1, the index subset $\mathcal{J}^k$ is chosen by Rules (i), (ii), or (iii), the other parameters are set as in Section 4. In our numerical tests, we terminate Algorithm 2.1 when

$$\|x^{k+1} - x^k\|/\|b\| \le 10^{-4}$$

and the maximal number of CGD iterations is less than 500.

Our numerical results are included in Table 4, where `it.`, `obj.`, `hd.`, `cpu.`, `res0.` and `res*.` stand for the number of the CGD iterations, the final objective value, the value $\|H^k d_{H^k}(x^k)\|_\infty$ at the final iteration, the cpu time (in seconds), and the relative residuals $\|x^k - x^*\|/\|x^*\|$ at the starting point x^0 and at the final iterate of our algorithm, respectively.

From Table 4, it is obvious that Rules (ii) and (iii) perform typically better than Rule (i) in terms of reducing the objective function value and the relative residual. However, as the parameter c is larger, Algorithm 2.1 with Rules (iii) may stop before the convergence since $\tilde{d}_D(x;\mathcal{J})$ may be far away from $d_D(x;\mathcal{J})$. This agrees with our prediction.

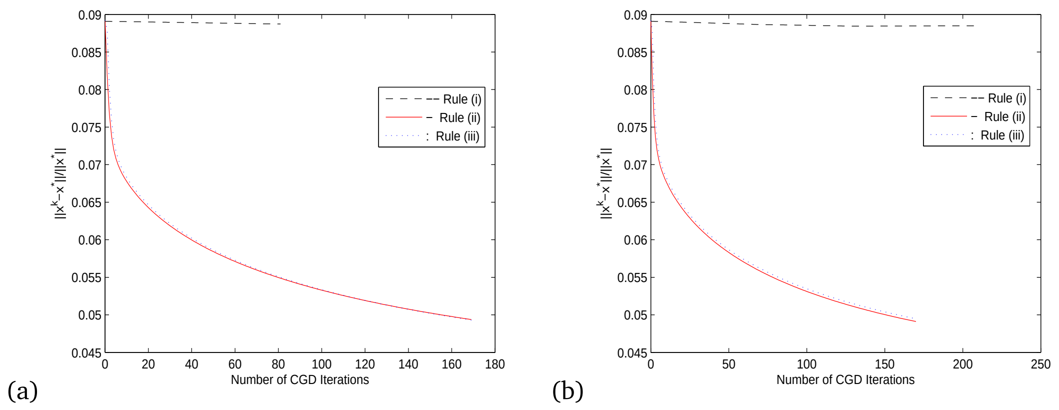


Figure 2: Convergence history of the CGD method for Problem (5.1). (a) L as in (4.2); (b) L as in (4.3).

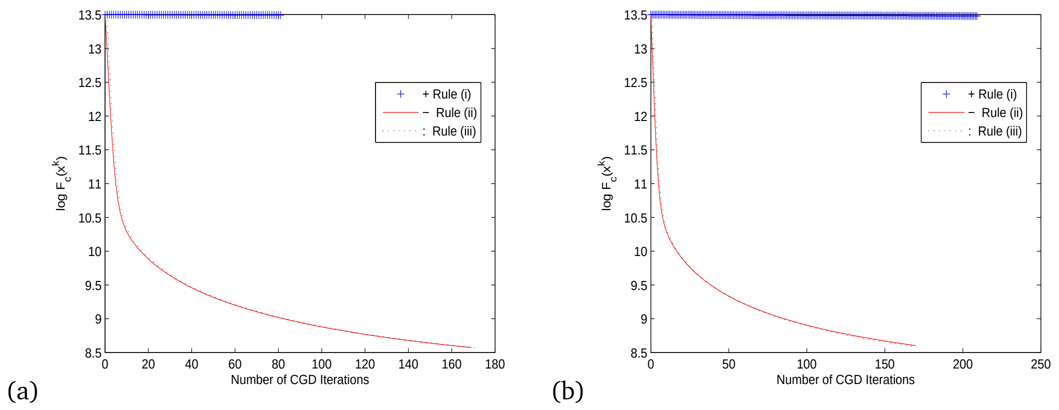


Figure 3: The objective function value $F_c(x^k)$ versus the number of the CGD iterations for Problem (5.1). (a) L as in (4.2); (b) L as in (4.3).

Fig. 1 shows the original, observed, initial guess, and restored images of Cameraman for Rule (iii) with $c = 0.001$.

To further illustrate the performance of the proposed method, in Figs. 2 and 3, we display the convergence history of Algorithm 2.1 for different rules with $c = 0.001$. We see from Fig. 2 that the proposed algorithm with Rules (ii) and (iii) works more efficiently. Also, we plot the natural logarithm of the objective function value versus the number of CGD iterations in Fig. 3 for different rules with $c = 0.001$. From Fig. 3, we can observe that the objective function value with Rule (i) is almost unchanged while the objective function value with Rules (ii) and (iii) decreases very fast. This shows that, the direction $\tilde{d}_D(x;\mathcal{J})$ is a feasible approximation to $d_D(x;\mathcal{J})$, especially when c is small.

Example 2. We consider the image deblurring problem. Let $x \in \mathbb{R}^n$ be the underlying image. Then the observed blurred image $b \in \mathbb{R}^n$ is given by

$$b = Ax + \eta, \tag{5.3}$$

where $\eta \in \mathbb{R}^n$ is a vector of noise and $A \in \mathbb{R}^{n\times n}$ is a linear blurring operator, see for instance [5].

We solve (5.3) in a tight frame domain. Let F be an $n \times m$ matrix whose column vectors form a tight frame in $\mathbb{R}^n$, i.e., $FF^T = I$. Moreover, we suppose that the original image x has a sparse approximation under F. Thus (5.3) turns into

$$b = AFu + \eta.$$

Then the underlying solution is given by $x = Fu$.

For the purpose of demonstration, the tight frame F is generated from the piecewise linear B-spline framelet and the coarsest decomposition level is set to be 4. The three filters are given by

$$h_0 = \frac{1}{4}[1,2,1], \quad h_1 = \frac{\sqrt{2}}{4}[1,0,-1], \quad h_2 = \frac{1}{4}[-1,2,-1].$$

Furthermore, the matrix F is the reconstruction operator and F^T is the decomposition operator of underlying tight framelet system, see [4] for the generation of the matrix F.

We apply the CGD algorithm to solve the following minimization problem:

$$\min_u F_c(u) \overset{\text{def}}{=} \|AFu - b\|_2^2 + c_1\|LFu\|_1 + c_2\|u\|_1, \tag{5.4}$$

where $c_1, c_2 > 0$ and L is given as in (4.2) or (4.3). Similarly, the index subset $\mathcal{J}^k$ is chosen by Rules (i), (ii), and (iii) in Section 4 with $x^k = u^k$ and $f(u) := \|AFu - b\|_2^2$, where the approximate search direction $\tilde{d}_D(u;\mathcal{J})$ is defined by

$$\tilde{d}_D(u;\mathcal{J}) \overset{\text{def}}{=} \arg\min_d \left\{ \nabla f(u)^T d + \frac{1}{2}d^T D d + (c_1\|B\|_1\|F\|_2 + c_2)\|u+d\|_1 \,|\, d_j = 0 \,\forall\, j \notin \mathcal{J} \right\}, \tag{5.5}$$

and $\tilde{q}(u)$ is given by

$$\tilde{q}_D(u;\mathcal{J}) \overset{\text{def}}{=} \left(\nabla f(u)^T d + \frac{1}{2}d^T D d + (c_1\|B\|_1\|F\|_2 + c_2)(\|u+d\|_1 - \|u\|_1) \right)_{d=\tilde{d}_D(u;\mathcal{J})}. \tag{5.6}$$

The other parameters in Algorithm 2.1 are set as in Section 4.

Remark 5.1. Following the similar proof of Proposition 2.1 in Section 2, we can also show that, for $D \succeq \underline{\delta} I \succ 0_n$,

$$\|\tilde{d}_D(u;\mathcal{J}) - d_D(u;\mathcal{J})\| \le \frac{2\sqrt{n}}{\underline{\delta}}(\|L\|_1\|F\|_2 c_1 + c_2).$$

This shows that $\tilde{d}_D(x;\mathcal{J}) \to d_D(x;\mathcal{J})$ as $c_1, c_2 \to 0$.

Table 5: Numerical results for the 256×256 Cameraman image convolved by a 15×15 Gaussian kernel of $\sigma_b = 2$ and contaminated by a Gaussian white noise of variance σ^2 ("*" means that the condition (2.8) is not satisfied).

$\sigma = 2$ and L is set to be (4.2)				
c_1	c_2	Rule (i)	Rule (ii)	Rule (iii)
		`it./ nz./ obj./cpu./res0./res*.`	`it./ nz./ obj./cpu./res0./res*.`	`it./ nz./ obj./cpu./res0./res*.`
1	1	153/479/1.1e9/264/1/0.999	237/153326/1.1e7/517/1/0.134	*66/264107/1.2e7/240/1/0.146
0.1	0.1	153/2142/1.1e9/265/1/0.999	85/817471/1.7e6/190/1/0.132	84/679553/1.7e6/310/1/0.132
0.001	0.1	153/2736/1.1e9/265/1/0.999	82/1223976/1.7e6/187/1/0.132	82/927371/1.7e6/300/1/0.132
0.0001	0.01	153/4249/1.1e9/264/1/0.999	82/1916809/6.2e5/186/1/0.132	83/1226224/6.3e5/306/1/0.131
0.001	0.001	153/4753/1.1e9/265/1/0.999	82/2059906/5.1e5/189/1/0.131	83/1273851/5.1e5/309/1/0.132
$\sigma = 2$ and L is set to be (4.3)				
c_1	c_2	Rule (i)	Rule (ii)	Rule (iii)
		`it./ nz./ obj./cpu./res0./res*.`	`it./ nz./ obj./cpu./res0./res*.`	`it./ nz./ obj./cpu./res0./res*.`
1	1	500/1253/1.1e9/863/1/0.997	297/106844/1.1e7/637/1/0.142	*68/193909/1.3e7/240/1/0.154
0.1	0.1	500/6710/1.1e9/870/1/0.997	92/622699/1.7e6/204/1/0.132	89/552058/1.7e6/316/1/0.132
0.001	0.1	500/10281/1.1e9/867/1/0.997	81/1216652/1.7e6/184/1/0.132	82/923758/1.7e6/299/1/0.132
0.0001	0.01	500/14772/1.1e9/869/1/0.997	82/1914421/6.2e5/188/1/0.132	82/1230000/6.3e5/301/1/0.132
0.001	0.001	500/15396/1.1e9/873/1/0.997	82/2018360/5.2e5/189/1/0.132	83/1264086/5.1e5/304/1/0.131

In our numerical tests, the noise η is set to be a Gaussian white noise of variance σ^2. The initial guess is given by $u^0 = 0$. As in [7], we choose $D^k = \rho^k I$, where

$$\rho^k = \begin{cases} 100, & k \le 10, \\ 20, & k > 10. \end{cases}$$

For our problem, Algorithm 2.1 is stopped when

$$\|u^{k+1} - u^k\| / \|b\| \le 5.0 \times 10^{-4}$$

and the maximal number of CGD iterations is set to be 500.

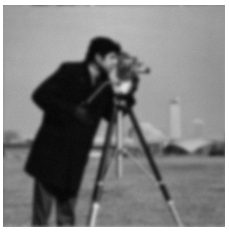
Now, we report the performance of Algorithm 2.1 with Rule (i), (ii), or (iii). Table 5 contains the numerical results for the 256×256 Cameraman image convolved by a 15×15 Gaussian kernel with $\sigma_b = 2$ (generated by the MATLAB-provided function `fspecial ('Gaussian',15,2)`) and contaminated by a Gaussian white noise of variance $\sigma = 2$ for different values of c_1 and c_2, where `it.`, `obj.`, `cpu.`, `res0.` and `res*.` are defined as above and `nz.` denotes the number of nonzero entries in the solution (an entry is regarded to be nonzero if its absolute value exceeds 10^{-6}). From Table 5, it is obvious that Algorithm 2.1 with Rules (ii) and (iii) performs more efficiently in terms of the objective function, the cpu time, and the relative residual. On the other hand, the solution to Problem (5.4) becomes more sparse if the values of c_1 and c_2 is growing larger. However, Algorithm 2.1 with Rule (iii) may stop as the parameters c_1 and c_2 is larger. This shows that the direction $\tilde{d}_D(x;\mathscr{J})$ may deviate from $d_D(x;\mathscr{J})$ when c_1, c_2 is too large, see Remark 5.1.

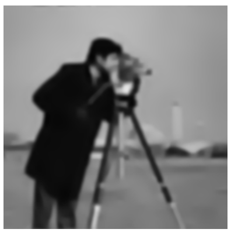
Table 6 lists the ratios (%) of between the number of nonzero entries (`nz.`) and the total number of entries (`n.`) in the solution obtained by Algorithm 2.1 for the 256×256 Cameraman image convolved by a 15×15 Gaussian kernel with $\sigma_b = 2$ and contaminated by a Gaussian white noise of variance $\sigma = 2$ for different values of c_1 and c_2. Here $\texttt{r1.} = \frac{\texttt{nz1.}}{\texttt{n.}} \times 100$, $\texttt{r2.} = \frac{\texttt{nz2.}}{\texttt{n.}} \times 100$, and $\texttt{r3.} = \frac{\texttt{nz3.}}{\texttt{n.}} \times 100$ `r2.`, where `nz1.`, `nz2.` and `nz3.` denote the number of nonzero entries in the solution when its entries larger than 10^{-4}, 10^{-5}, and 10^{-6}, respectively.

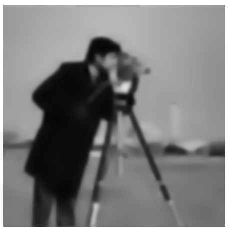
Table 6: Numerical results for the 256 × 256 Cameraman image convolved by a 15 × 15 Gaussian kernel of $\sigma_b = 2$ and contaminated by a Gaussian white noise of variance σ^2 ("*" means that the condition (2.8) is not satisfied).

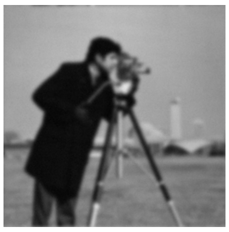
$\sigma = 2$ and L is set to be (4.2)				
c_1	c_2	Rule (i)	Rule (ii)	Rule (iii)
		`r1./r2./r3.` (%)	`r1./r2./r3.` (%)	`r1./r2./r3.` (%)
1	1	0.0221/0.0221/0.0221	7.0739/7.0823/7.0896	12.2093/12.2117/12.2120
0.1	0.1	0.0990/0.0990/0.0990	37.7247/37.7735/37.7988	31.4124/31.4209/31.4217
0.001	0.1	0.1262/0.1265/0.1265	56.4570/56.5484/56.5951	42.8772/42.8802/42.8805
0.0001	0.01	0.1932/0.1961/0.1965	88.4682/88.5923/88.6309	56.6983/56.6990/56.6991
0.001	0.001	0.2147/0.2190/0.2198	95.1364/95.2305/95.2475	58.9009/58.9012/58.9013

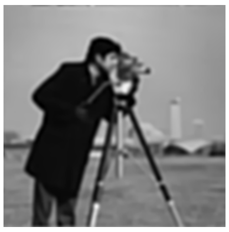
$\sigma = 2$ and L is set to be (4.3)				
c_1	c_2	Rule (i)	Rule (ii)	Rule (iii)
		`r1./r2./r3.` (%)	`r1./r2./r3.` (%)	`r1./r2./r3.` (%)
1	1	0.0579/0.0579/0.0579	4.9320/4.9371/4.9403	8.9642/8.9658/8.9661
0.1	0.1	0.3099/0.3103/0.3103	28.7400/28.7716/28.7928	25.5146/25.5252/25.5265
0.001	0.1	0.4745/0.4753/0.4754	56.1263/56.2103/56.2565	42.7092/42.7129/42.7134
0.0001	0.01	0.6778/0.6823/0.6830	88.3593/88.4825/88.5204	56.8730/56.8737/56.8737
0.001	0.001	0.7044/0.7109/0.7119	93.1963/93.3025/93.3265	58.4493/58.4497/58.4498


(a) Noise $\sigma = 2$


(b) using (4.2) $c_1 = c_2 = 1.0$


(c) using (4.3) $c_1 = c_2 = 1.0$


(d) Noise $\sigma = 2$


(e) using (4.2) $c_1 = c_2 = 0.1$

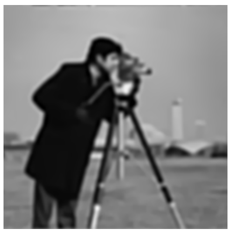
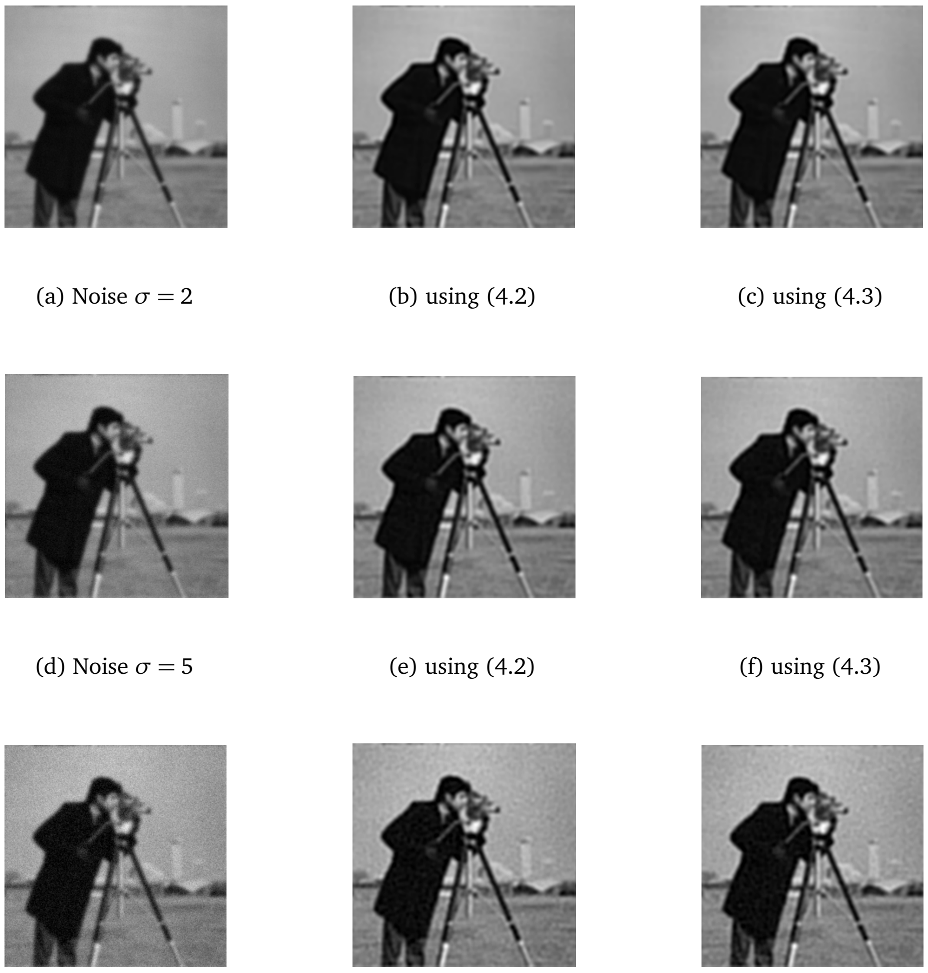

(f) using (4.3) $c_1 = c_2 = 0.1$

Figure 4: Noisy blurred (left) and deblurred images (center and right) of Cameraman convolved by a 15 × 15 Gaussian kernel of $\sigma_b = 2$ and contaminated by a Gaussian noise of variance σ^2.

Fig. 4 shows the results of the noisy blurred image and deblurred image for the 256 × 256 Cameraman image by Algorithm 2.1 with Rule (iii) for $c_1 = c_2 = 1.0$ (see Figs. 4(a), (b), (c)) and $c_1 = c_2 = 0.1$ (see Figs. 4(d), (e), (f)), where the blurring kernels are a



(a) Noise $\sigma = 2$ (b) using (4.2) (c) using (4.3)

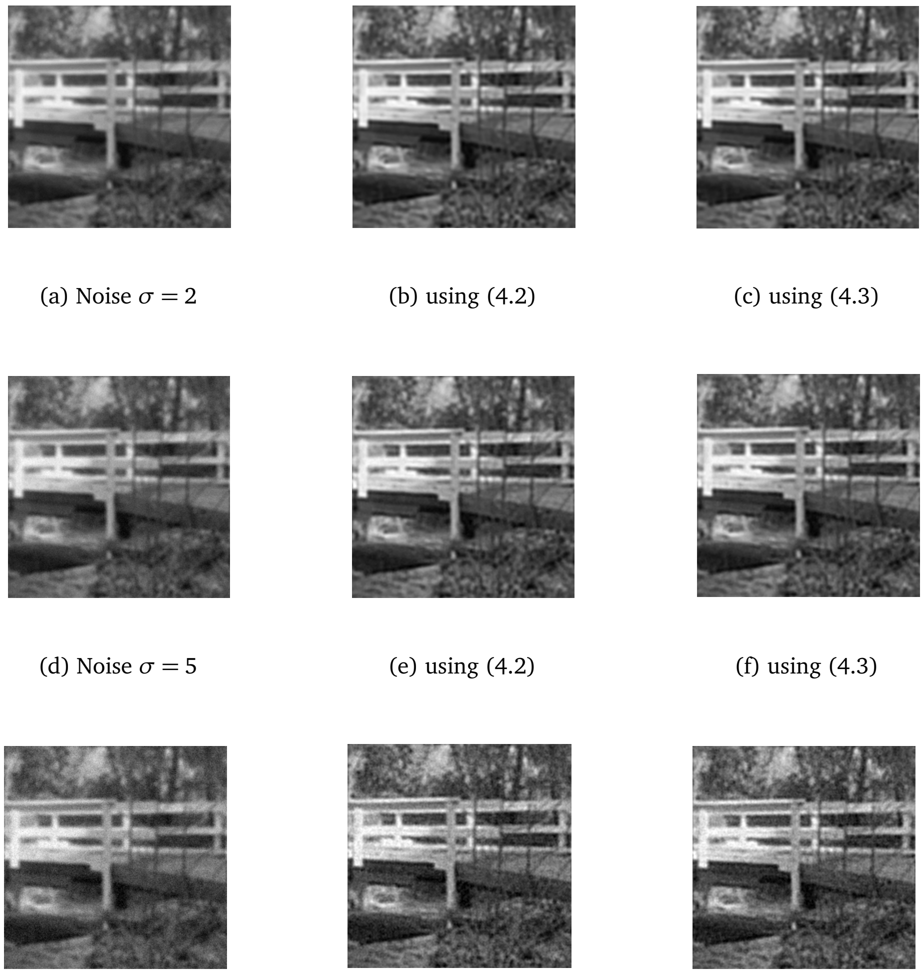
(d) Noise $\sigma = 5$ (e) using (4.2) (f) using (4.3)

(g) Noise $\sigma = 10$ (h) using (4.2) (i) using (4.3)

Figure 5: Noisy blurred (left) and deblurred images (center and right) of Cameraman convolved by a 15×15 Gaussian kernel of $\sigma_b = 2$ and contaminated by a Gaussian noise of variance σ^2.

15×15 Gaussian kernel with $\sigma_b = 2$. We observe Fig. 4 that the restored image becomes worse when the value of c_1 and c_2 is larger.

Figs. 5-6 display the results of the noisy blurred image and deblurred image for the 256×256 Cameraman and 260×260 Bridge images by Algorithm 2.1 with Rule (iii) for $c_1 = c_2 = 0.001$, where the blurring kernels are a 15×15 Gaussian kernel with $\sigma_b = 2$ and a 7×7 disk kernel (generated by the `MATLAB`-provided function `fspecial ('disk',3))`. From Figs. 5-6, it is easy to see that Algorithm 2.1 with Rule (iii) is very effective. It is also shown that the algorithm is robust to noise, since it still gives good restored images when the noise is as high as $\sigma = 10$.



(a) Noise $\sigma = 2$ (b) using (4.2) (c) using (4.3)

(d) Noise $\sigma = 5$ (e) using (4.2) (f) using (4.3)

(g) Noise $\sigma = 10$ (h) using (4.2) (i) using (4.3)

Figure 6: Noisy blurred (left) and deblurred images (center and right) of Bridge a 7×7 disk kernel and contaminated by a Gaussian white noise of variance σ^2.

To further illustrate the performance of the proposed method, in Figs. 7 and 8, we, respectively, plot the convergence history of Algorithm 2.1 and the natural logarithm of the objective function value for the 256×256 Cameraman image convolved by a 15×15 Gaussian kernel with $\sigma_b = 2$ and contaminated by a Gaussian white noise of variance $\sigma = 2$ with $c_1 = c_2 = 0.001$. From Fig. 7, we observe that the proposed algorithm with Rules (ii) and (iii) works more efficiently. Fig. 8 shows that the objective function value with Rule (i) is almost unchanged while the objective function value with Rules (ii) and (iii) decreases very fast. This indicates that the direction $\tilde{d}_D(u;\mathscr{J})$ is a effective approximation to $d_D(u;\mathscr{J})$ when both c_1 and c_2 are small.

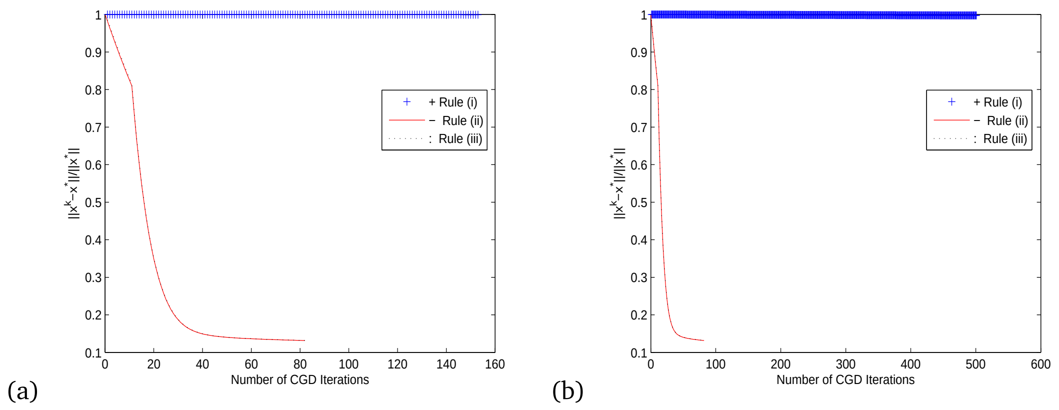


Figure 7: Convergence history of the CGD method for Problem (5.4). (a) L as in (4.2); (b) L as in (4.3).

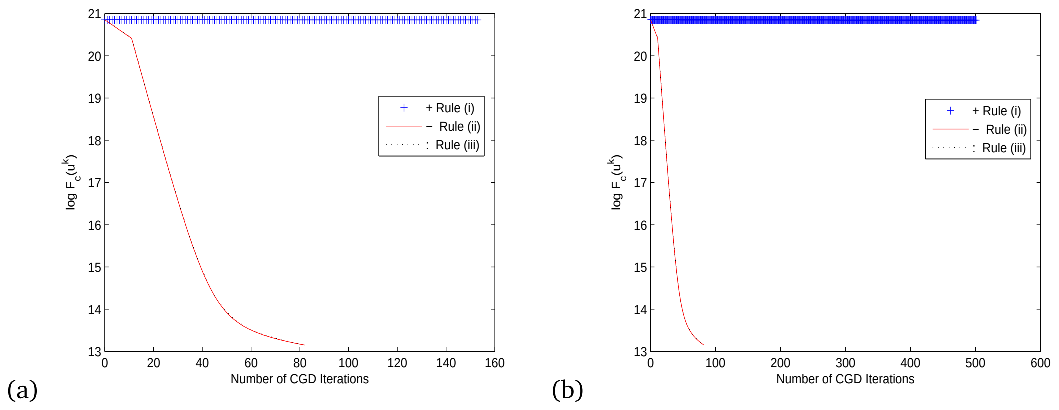


Figure 8: The objective function value $F_c(u^k)$ versus the number of the CGD iterations for Problem (5.4). (a) L as in (4.2); (b) L as in (4.3).

6. Concluding remarks

In conclusion, we have proposed an efficient coordinate gradient descent algorithm for solving the nonsmooth nonseparable minimization problems. We find the search direction from a subproblem obtained by a second-order approximation of the smooth function and adding a separable convex function. We show that our algorithm converges globally if the coordinates are chosen by either the Gauss-Seidel rule or the Gauss-Southwell-r rule or the Gauss-Southwell-q rule. We also prove that our approach with the Gauss-Southwell-q rule converges at least linearly based on a local Lipschitzian error bound assumption. We report some numerical tests to demonstrate the efficiency of the proposed method.

Acknowledgments The research of the first author is partially supported by NSFC Grant 10601043, NCETXMU, and SRF for ROCS, SEM. The research of the second author is supported in part by RGC 201508 and HKBU FRGs. The research of the third author is supported by the Hong Kong Research Grant Council.

References

[1] S. Alliney and S. Ruzinsky, *An algorithm for the minimization of mixed $\ell 1$ and $\ell 2$ norms with application to Bayesian estimation*, IEEE Trans. Signal Process., 42 (1994), pp. 618–627.

[2] A. Auslender, *Minimisation de fonctions localement lipschitziennes: applications à la programmation mi-convexe, mi-différentiable*, In: Mangasarian, O.L., Meyer, R.R., and Robinson, S.M. (eds.) Nonlinear Programming, vol. 3, pp. 429–460. Academic, New York, 1978.

[3] P. S. Bradley, U. M. Fayyad, and O. L. Mangasarian, *Mathematical programming for data mining: formulations and challenges*, INFORMS J. Comput. 11 (1999), pp. 217–238.

[4] J.-F. Cai, R. H. Chan, and Z. Shen, *A framelet-based image inpainting algorithm*, Appl. Comput. Harmon. Anal., 24 (2008), pp. 131–149.

[5] H. Y. Fu, M. K. Ng, M. Nikolova, and J. L. Barlow, *Efficient minimization methods of mixed $\ell 2$-$\ell 1$ and $\ell 1$-$\ell 1$ norms for image restoration*, SIAM J. Sci. Comput., 27 (2006), pp.1881–1902.

[6] M. Fukushima, *A successive quadratic programming method for a class of constrained nonsmooth optimization problems*, Math. Program., 49 (1990), pp. 231–251.

[7] M. Fukushima and H. Mine, *Ageneralized proximal point algorithm for certain non-convex minimization problems*, Int. J. Syst. Sci., 12 (1981), pp. 989–1000.

[8] A. Guitton and D. J. Verschuur, *Adaptive subtraction of multiples using the $\ell 1$-norm*, Geophys. Prospecting, 52 (2004), pp.1–27.

[9] E. T. Hale, W. Yin, and Y. Zhang, *A fixed-point continuation method for $\ell 1$-regularized minimization with applications to compressed sensing*, CAAM Tech- nical Report TR07-07, Department of Computational and Applied Mathematics, Rice University, July 2007.

[10] V. G. Karmanov, *Mathematical Programming*, Mir Pub., Moscow, 1989.

[11] K. C. Kiwiel, *A method for minimizing the sum of a convex function and a continuously differentiable function*, J. Optim. Theory Appl., 48 (1986), pp. 437–449.

[12] F. R. Lin, M. K. Ng, and W. K. Ching, *Factorized banded inverse preconditioners for matrices with Toeplitz structure*, SIAM J. Sci. Comput., 26 (2005), pp. 1852–1870.

[13] M. Lustig, D. Donoho, and J. Pauly, *Sparse MRI: The application of compressed sensing for rapid MR imaging*, Magnetic Resonance in Medicine, 58 (2007), pp. 1182–1195.

[14] O. L. Mangasarian and D. R. Musicant, *Large scale kernel regression via linear programming*, Mach. Learn. 46 (2002), pp.255–269.

[15] H. Mine and M. Fukushima, *A minimization method for the sum of a convex function and a continuously differentiable function*, J. Optim. Theory Appl., 33 (1981), pp. 9–23.

[16] J. J. Moré, B. S. Garbow, and K. E. Hillstrom, *Testing unconstrained optimization software*, ACM Trans. Math. Softw., 7 (1981), pp. 17–41.

[17] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin, *An iterated regularization method for total variation-based image restoration*, Multiscale Model. Simul., 4 (2005), pp. 460–489.

[18] R. T. Rockafellar and R. J.-B. Wets,*Variational Analysis*, Springer, New York, 1998.

[19] L. Rudin, S. J. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Phys. D, 60 (1992), pp. 259–268.

[20] S. Ruzinsky, *Sequential Least Absolute Deviation Estimation of Autoregressive Parameters*, Ph.D. thesis, Illinois Institute of Technology, Chicago, IL, 1989.

[21] S. Sardy, A. Bruce, and P. Tseng, *Robust wavelet denoising*, IEEE Trans. Signal Proc. 49 (2001), pp.1146–1152.

[22] P. Tseng and S. Yun, *A coordinate gradient descent method for nonsmooth separable minimization*, Math. Program., Ser. B, 117 (2008), pp. 387–423.

[23] C. R. Vogel and M. E. Oman, *Iterative methods for total variation denoising*, SIAM J. Sci. Comput., 17 (1996), pp. 227–238.

[24] M. Wakin, J. Laska, M. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. Kelly, and R. Baraniuk, *An architecture for compressiving image*, In Proceedings of the International Conference on Image Processing (ICIP), Atlanta, Georgia, 2006.

[25] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, *Bregman iterative algorithms for ℓ1-minimization with applications to compressed sensing*, SIAM J. Imaging Sci., 1 (2008), pp. 143–168.