Hindawi
Mathematical Problems in Engineering
Volume 2019, Article ID 5261830, 17 pages
https://doi.org/10.1155/2019/5261830

Research Article

A Modified Spectral PRP Conjugate Gradient Projection Method for Solving Large-Scale Monotone Equations and Its Application in Compressed Sensing

Jie Guo and Zhong Wan

School of Mathematics and Statistics, Central South University, Hunan, Changsha, China

Correspondence should be addressed to Zhong Wan; wanmath@163.com

Received 24 November 2018; Accepted 24 March 2019; Published 8 April 2019

Academic Editor: Higinio Ramos

In this paper, we develop an algorithm to solve nonlinear system of monotone equations, which is a combination of a modified spectral PRP (Polak-Ribière-Polyak) conjugate gradient method and a projection method. The search direction in this algorithm is proved to be sufficiently descent for any line search rule. A line search strategy in the literature is modified such that a better step length is more easily obtained without the difficulty of choosing an appropriate weight in the original one. Global convergence of the algorithm is proved under mild assumptions. Numerical tests and preliminary application in recovering sparse signals indicate that the developed algorithm outperforms the state-of-the-art similar algorithms available in the literature, especially for solving large-scale problems and singular ones.

1. Introduction

In many fields of sciences and engineering, solution of a nonlinear system of equations is a fundamental problem. For example, in [1, 2], both a Nash economic equilibrium problem and a signal processing problem were formulated into a nonlinear system of equations. Owing to complexity, in the past five decades, numerous algorithms and some software packages in virtue of those powerful algorithms have been developed for solving the nonlinear system of equations. See, for example, [1, 3–16] and the references therein. Nevertheless, in practice, no any algorithm can efficiently solve all the systems of equations arising from sciences and engineering. It is significant to develop a specific algorithm to solve the problems with different analytic and structural features [17, 18].

In this paper, we consider the following nonlinear system of monotone equations:

$$F(x) = 0, \tag{1}$$

where $F : R^n \longrightarrow R^n$ is a continuous and monotone function; that is to say, F satisfies

$$(F(x) - F(y))^T (x - y) \geqslant 0 \quad \forall x, y \in R^n. \tag{2}$$

It has been shown that the solution set of problem (1) is convex if it is nonempty [3]. In addition, throughout the paper, the space R^n is equipped with the Euclidean norm $\|\cdot\|$ and the inner product $\langle x, y\rangle = x^T y$, for $x, y \in R^n$.

Aiming at solution of problem (1), many efficient methods were developed recently. Only by incomplete enumeration, we here mention the trust region method [19], the Newton and the quasi-Newton methods [4–6, 19], the Gauss-Newton methods [7, 8], the Levenberg-Marquardt methods [20–22], the derivative-free methods and its modified versions [9–16, 23–27], the derivative-free conjugate gradient projection method [14], the modified PRP (Polak-Ribière-Polyak) conjugate gradient method [11], the TPRP method [10], the PRP-type method [28], the projection method [23], the FR-type method [9], and the modified spectral conjugate gradient projection method [13]. Summarily, the spectral gradient methods and the conjugate gradient methods are more popular in solving a large-scale nonlinear system of equations than the Newton and the quasi-Newton methods. One of the former's advantages lies in that there is no requirement of computing and storing the Jacobian matrix or its approximation.

Specifically, Li introduced a class of methods for large-scale nonlinear monotone equations in [10], which include the SG-liked method, the MPRP method, and the TPRP method. Chen [28] proposed a PRP method for large-scale nonlinear monotone equations. A descent modified PRP method and FR-type methods were presented in [9, 11], respectively. Liu and Li proposed a projection method for convex constrained monotone nonlinear equations in [23]. Two derivative-free conjugate gradient projection methods were presented for such a system in [14]. Three extensions of conjugate gradient algorithms were developed in [24–26], respectively. Based on the projection technique in [29], Wan and Liu proposed a modified spectral conjugate gradient projection method (MSCGP) to solve a nonlinear monotone system of symmetric equations in [13]. Then, in [2], MSCGP was successfully applied into recovering sparse signals and restoring blurred images.

It is noted that the main idea of [13, 23] is to construct a search direction by projection technique such that it is sufficiently descent. In virtue of derivative-free and low storage properties, numerical experiments indicated that the developed algorithm in [13] is more efficient to solve large-scale nonlinear monotone systems of equations than the similar ones available in the literature.

In [30], Yang et al. proposed a modified spectral PRP conjugate gradient method for solving unconstrained optimization problem. It was proved that the search direction at each iteration is a descent direction of objective function and global convergence was established under mild conditions. Our research interest in this paper is to study how to extend this method to solution of problem (1). Specifically, we should address the following issues:

(1) Without need of derivative information of the function F, how to determine the spectral and conjugate parameters to get a sufficiently descent search direction at each iteration?

(2) To ensure global convergence of algorithm, how to choose an appropriate step length for the given search direction? Particularly, monotonicity of F should be utilized to design a new iterate scheme?

(3) What about the numerical performance of new iteration scheme? Especially, whether it is more efficient or not than the similar algorithms available in the literature.

Note that a new line search rule was proposed in [31] for solving nonlinear monotone equations with convex constraints, and it was shown that, in virtue of this line search, the developed algorithm has good numerical performance. However, the presented line search is involved with choice of a weight. Since it may be difficult to choose an appropriate weight in the practical implementation of algorithm, we attempt to overcome this difficulty in this paper.

Summarily, we intend to propose a modified spectral PRP conjugate gradient derivative-free projection method for solving large-scale nonlinear equations. Global convergence of this method will be proved, and numerical tests will be conducted by implementing the developed algorithm to solve benchmark large-scale test problems and to reconstruct sparse signals in compressive sensing.

The rest of this paper is organized as follows. In Section 2, we first state the idea to propose a new spectral PRP conjugate gradient method. Then, a new algorithm is developed. Global convergence is established in Section 3. Section 4 is devoted to numerical experiments. Preliminary application of the algorithm is presented in Section 5. Some conclusions are drawn in Section 6.

2. Development of New Algorithm

In this section, we will state how to develop a new algorithm in detail.

2.1. Projection Method. Generally, to solve (1), we need to construct an iterative format as follows:

$$x_{k+1} = x_k + \alpha_k d_k, \tag{3}$$

where $\alpha_k > 0$ is called a step length and d_k is a search direction. Let $z_k = x_k + \alpha_k d_k$. If z_k satisfies

$$F(z_k)^T (x_k - z_k) > 0, \tag{4}$$

then a projection method can be obtained for solving Problem (1). Actually, by monotonicity of F, it holds that

$$F(z_k)^T (x^* - z_k) = (F(z_k) - F(x^*))^T (x^* - z_k) \le 0, \tag{5}$$

for any solution of (1), x^*. With such a z_k, we define a hyperplane:

$$H_k = \{x \in R^n \mid F(z_k)^T (x - z_k) = 0\}. \tag{6}$$

From (4) and (5), it is clear that H_k strictly separates the iterate point x_k from the solution x^*. Thus, the projection of x_k onto H_k is closer to x^* than x_k. Consequently, the iterative format

$$x_{k+1} = x_k - \frac{F(z_k)^T (x_k - z_k)}{\|F(z_k)\|^2} F(z_k) \tag{7}$$

is referred to as the projection method proposed in [29]. Both analytic properties and numerical results have shown efficiency and robustness of the projection-based algorithms for monotone system of equations [10, 13, 14]. In this paper, we intend to propose a new spectral conjugate gradient method also in virtue of the above projection technique.

2.2. A Modified Spectral PRP Conjugate Gradient Method. In the projection method (7), it is noted that z_k must satisfy (4). That is to say, d_k should be a search direction satisfying

$$F(z_k)^T d_k < 0. \tag{8}$$

Very recently, Wan et al. [13] proposed a modified spectral conjugate gradient projection method for solving nonlinear monotone symmetric equations, where d_k was chosen by

$$d_k = \begin{cases} -F_k, & \text{if } k = 0, \\ -\theta_k F_k + \beta_k d_{k-1}, & \text{if } k > 0, \end{cases} \tag{9}$$

and β_k and θ_k are computed by

$$\begin{aligned} \theta_k &= \frac{d_{k-1}^T \left(y_{k-1} - F_k F_k^T s_{k-1} / \|F_k\|^2 \right)}{d_{k-1}^T \left(I - F(x_k) F(x_k)^T / \|F(x_k)\|^2 \right) y_{k-1}}, \\ \beta_k &= \frac{F_k^T (y_{k-1} - s_{k-1})}{d_{k-1}^T \left(I - F(x_k) F(x_k)^T / \|F(x_k)\|^2 \right) y_{k-1}}, \end{aligned} \tag{10}$$

respectively, $s_{k-1} = x_k - x_{k-1}$, and $y_{k-1} = F(x_k) - F(x_{k-1})$. It was proved in [13] that d_k given by (9) and (10) is sufficiently descent and satisfies $F_k^T d_k = -\|F_k\|^2$.

Note that a modified spectral PRP conjugate gradient method was proposed for solving unconstrained optimization problems in [30]. Similar to the idea in [13], we can extend the method in [30] to solution of problem (1). Specifically, we compute β_k and θ_k in (9) by

$$\begin{aligned} \theta_k &= \frac{d_{k-1}^T y_{k-1}}{\|F_{k-1}\|^2} - \frac{d_{k-1}^T F_k F_k^T F_{k-1}}{\|F_k\|^2 \|F_{k-1}\|^2}, \\ \beta_k &= \frac{F_k^T y_{k-1}}{\|F_{k-1}\|^2}, \end{aligned} \tag{11}$$

respectively. Although (11) gives different choices of β_k and θ_k from (10), we can also prove the following result.

Proposition 1. *Let d_k be given by (9) and (11). Then, for any $k \geq 0$, the following equality holds:*

$$F_k^T d_k = -\|F_k\|^2. \tag{12}$$

Proof. For $k = 0$, we have

$$F_0^T d_0 = F_0^T (-F_0) = -\|F_0\|^2. \tag{13}$$

For $k = 1$, we have

$$\begin{aligned} F_1^T d_1 &= F_1^T (-\theta_1 F_1 + \beta_1 d_0) \\ &= -\frac{d_0^T (F_1 - F_0)}{\|F_0\|^2} F_1^T F_1 + \frac{d_0^T F_1 F_1^T F_0}{\|F_1\|^2 \|F_0\|^2} F_1^T F_1 \\ &\quad + \frac{F_1^T (F_1 - F_0)}{\|F_0\|^2} d_0^T F_1 = \frac{d_0^T F_0}{\|F_0\|^2} F_1^T F_1 \\ &= \frac{\|F_1\|^2}{\|F_0\|^2} \left(-\|F_0\|^2 \right) = -\|F_1\|^2. \end{aligned} \tag{14}$$

We now prove that if

$$F_{k-1}^T d_{k-1} = -\|F_{k-1}\|^2, \tag{15}$$

holds for $k-1$ ($k > 1$), then (12) also holds for k.

Actually,

$$\begin{aligned} F_k^T d_k &= F_k^T (-\theta_k F_k + \beta_k d_{k-1}) \\ &= -\frac{d_{k-1}^T (F_k - F_{k-1})}{\|F_{k-1}\|^2} F_k^T F_k \\ &\quad + \frac{d_{k-1}^T F_k F_k^T F_{k-1}}{\|F_k\|^2 \|F_{k-1}\|^2} F_k^T F_k \\ &\quad + \frac{F_k^T (F_k - F_{k-1})}{\|F_{k-1}\|^2} d_{k-1}^T F_k = \frac{d_{k-1}^T F_{k-1}}{\|F_{k-1}\|^2} F_k^T F_k \\ &= \frac{-\|F_{k-1}\|^2}{\|F_{k-1}\|^2} \|F_k\|^2 = -\|F_k\|^2, \end{aligned} \tag{16}$$

where the forth equality follows condition (15). Consequently, by mathematical induction, (12) holds for any k. □

Proposition 1 ensures that the idea of projection method can be incorporated into design of iteration schemes to solve (1) as the search direction is determined by (9) and (11).

2.3. Modified Line Search Rule. Since it is critical to choose an appropriate step length to improve the performance of the iterate scheme (3), as well as determination of search directions, we now present an inexact line search rule to determine α_k in (3).

Very recently, Ou and Li [31] presented a line search rule as follows: find a nonnegative step length $\alpha_k = \max\{s\rho^i : i = 0, 1, 2, \ldots, \}$ such that the following inequality is as follows:

$$-F(x_k + \alpha_k d_k)^T d_k \geq \sigma \alpha_k \gamma_k \|d_k\|^2, \tag{17}$$

where d_k is a fixed search direction, $s > 0$ is a given initial step length, $\rho \in (0, 1)$ and $\sigma > 0$ are two given constants, and γ_k is specified by

$$\gamma_k = \lambda_k + (1 - \lambda_k) \|F(x_k + \alpha_k d_k)\|. \tag{18}$$

In [31], it was required that λ_k in (18) satisfies $\lambda_k \in [\lambda_{min}, \lambda_{max}] \subseteq (0, 1]$. Clearly, λ_k and $(1 - \lambda_k)$ are the weights for the values 1 and $\|F(x_k + \alpha_k d_k)\|$, respectively. In the practical implementation, it is may be difficult to choose an appropriate λ_k. To overcome this difficulty, we choose γ_k in (18) by

$$\gamma_k = \min\{1, \|F(x_k + \alpha_k d_k)\|\}. \tag{19}$$

It is sure that, for the new method as a combination of (9), (11), (17), and (19), we need to establish its convergence theory and to further test its numerical performance.

Remark 2. In [31], to ensure that well-defined the line search (17) is well-defined, it is assumed that d_k satisfies

$$F(x_k)^T d_k \leq -\tau \|F(x_k)\|^2, \tag{20}$$

where $\tau > 0$ is a given constant. By Proposition 1, it is clear that d_k chosen by (9) and (11) satisfies (20) as $\tau = 1$.

Input:
An initial point $x_0 \in R^n$, positive constants k_{max}, σ, ε, s and $\rho \in (0, 1)$.
Begin:
$k \longleftarrow 0$;
$F_0 \longleftarrow F(x_0)$;
While $(\|F_k\| \geq \varepsilon$ *and* $k < k_{max})$
Step 1. (Search direction)
Compute d_k by (9) and (11).
Step 2. (Step length)
$\alpha \longleftarrow s$;
Compute $F_k \longleftarrow F(x_k + \alpha d_k)$.
While $(-F_k^T d_k < \sigma \alpha \gamma_k \|d_k\|^2)$
$\alpha \longleftarrow \rho\alpha$;
Compute $F_k \longleftarrow F(x_k + \alpha d_k)$.
End While
$\alpha_k \longleftarrow \alpha$;
Step 3. (Projection and update)
$z_k \longleftarrow x_k + \alpha_k d_k$;
If $\|F(z_k)\| \leq \varepsilon$
$x_{k+1} \longleftarrow z_k$;
Break.
End If
Compute x_{k+1} by (7);
$F_{k+1} \longleftarrow F(x_{k+1})$;
$k \longleftarrow k + 1$.
End While
End

Algorithm 1: Modified spectral PRP derivative-free projection-based algorithm (MPPRP).

2.4. Development of New Projection-Based Algorithm. With the above preparation, we are in a position to develop an algorithm to solve problem (1) by combining the projection technique and the new methods to determine a search direction and a step length.

We now present the computer procedure of Algorithm 1 .

Remark 3. Since Algorithm 1 does not involve computing the Jacobian matrix of F or its approximation, both information storage and computational cost of the algorithm are lower. In virtue of this advantage, Algorithm 1 is helpful to solution of large-scale problems. In next section, we will prove that Algorithm 1 is applicable even if F is nonsmooth. Our numerical tests in Section 4 will further show that Algorithm 1 can find a singular solution of problem (1) (see problem 5 and Table 5).

Remark 4. Compared with the algorithm developed in [13], problem (1) is not assumed to be a symmetric system of equations.

3. Convergence

In this section, we are going to study global convergence of Algorithm 1.

Apart from different choices of search direction and step length, Algorithm 1 can be treated as a variant of the projection algorithm in [29]. So, similar to some critical points of establishing global convergence in [29], we attempt to prove that Algorithm 1 is globally convergent. Very recently, locally linear convergence was proved in [32] for some PRP-type projection methods.

We first state the following mild assumptions.

Assumption 5. The function F is monotone on R^n.

Assumption 6. The solution set of problem (1) is nonempty.

Assumption 7. The function F is Lipschitz continuous on R^n; namely, there exists a positive constant L such that for all $x, y \in R^n$,

$$\|F(x) - F(y)\| \leq L \|x - y\| . \tag{21}$$

Under these assumptions, we can prove that Algorithm 1 has the following nice properties.

Lemma 8. *Let $\{x_k\}$ be a sequence generated by Algorithm 1. If Assumptions 5, 6, and 7 hold, then*

(1) for any x^*, such that $F(x^*) = 0$,

$$\|x_{k+1} - x^*\|^2 \leq \|x_k - x^*\|^2 - \|x_{k+1} - x_k\|^2 . \tag{22}$$

(2) The sequence $\{x_k\}$ is bounded.

(3) If $\{x_k\}$ is a finite sequence, then the last iterate point is a solution of problem (1); otherwise,

$$\sum_{i=1}^{n} \|x_{k+1} - x_k\|^2 < \infty, \tag{23}$$

and

$$\lim_{k\longrightarrow\infty}\|x_{k+1}-x_k\|=0. \quad (24)$$

(4) The sequence $\{\|F(x_k)\|\}$ is bounded. Hence, there exists a constant M_f such that $\|F(x_k)\|\leq M_f$.

Proof. (1) Let x^* be any point such that $F(x^*)=0$. Then, by monotonicity of F, we have

$$\langle F(z_k), x^*-z_k\rangle \leq 0. \quad (25)$$

From (7), it is also easy to verify that x_{k+1} is the projection of x_k onto the halfspace:

$$\{x\in R^n \mid \langle F(z_k), x-z_k\rangle\leq 0\}. \quad (26)$$

Thus, it follows from (25) that x^* belongs to this halfspace. From the basic properties of projection operator [33], we know that

$$\langle x_k-x_{k+1}, x_{k+1}-x^*\rangle \geq 0. \quad (27)$$

Consequently,

$$\begin{aligned}\|x_k-x^*\|^2 &= \|x_k-x_{k+1}\|^2+\|x_{k+1}-x^*\|^2\\ &\quad +2\langle x_k-x_{k+1}, x_{k+1}-x^*\rangle\\ &\geq \|x_k-x_{k+1}\|^2+\|x_{k+1}-x^*\|^2.\end{aligned} \quad (28)$$

The desired result (22) is directly obtained from (28).

(2) From (28), it is clear that the sequence $\{\|x_k-x^*\|\}$ is nonnegative and decreasing. Thus, $\{\|x_k-x^*\|\}$ is a convergent sequence. It is concluded that $\{x_k\}$ is bounded.

(3) From (28), we know

$$\|x_{k+1}-x_k\|^2\leq\|x_k-x^*\|^2-\|x_{k+1}-x^*\|^2. \quad (29)$$

Thus,

$$\begin{aligned}\sum_{k=1}^{n}\|x_{k+1}-x_k\|^2 &\leq \sum_{k=1}^{n}\left(\|x_k-x^*\|^2-\|x_{k+1}-x^*\|^2\right)\\ &= \|x_1-x^*\|^2-\|x_{n+1}-x^*\|^2.\end{aligned} \quad (30)$$

Since the sequence $\{x_k\}$ is bounded, the series $\{\sum_{k=1}^{n}\|x_{k+1}-x_k\|^2\}$ is convergent. Consequently,

$$\lim_{k\longrightarrow\infty}\|x_{k+1}-x_k\|^2=0. \quad (31)$$

The third result has been proved.

(4) For any x_k, by Lipschitz continuity, we have

$$\|F(x_k)\|=\|F(x_k)-F(x^*)\|\leq L\|x_k-x^*\|. \quad (32)$$

Since $\{\|x_k-x^*\|\}$ is convergent, we conclude that $\{\|F(x_k)\|\}$ is bounded. Consequently, for all $k\in N$, there exists a constant M_f such that $\|F(x_k)\|\leq M_f$. □

Lemma 8 indicates that, for the sequence $\{x_k\}$ generated by Algorithm 1, the sequence $\{x_k-x^*\}$ is decreasing and convergent, and the sequence $\{x_k\}$ is bounded, where x^* is any solution of problem (1).

Lemma 9. *Suppose that Assumptions 5, 6, and 7 hold. Let $\{d_k\}$ be a sequence generated by Algorithm 1. If there exists a constant $\varepsilon_0>0$ such that, for any positive integer k,*

$$\|F_k\|\geq\varepsilon_0. \quad (33)$$

Then, the sequence of directions $\{d_k\}$ is bounded; i.e., there exists a constant $M>0$ such that, for any positive integer k,

$$\|d_k\|\leq M. \quad (34)$$

Proof. From (9), (11), (12), and the results of Lemma 8, it follows that

$$\begin{aligned}\|d_k\| &= \|-\theta_kF_k+\beta_kd_{k-1}\| = \Bigg\|-\frac{d_{k-1}^T(F_k-F_{k-1})}{\|F_{k-1}\|^2}F_k\\ &\quad +\frac{d_{k-1}^TF_kF_k^TF_{k-1}}{\|F_k\|^2\|F_{k-1}\|^2}F_k+\frac{F_k^T(F_k-F_{k-1})}{\|F_{k-1}\|^2}d_{k-1}\Bigg\| = \Bigg\|-F_k\\ &\quad -d_{k-1}^TF_k\frac{F_k^T(F_k-F_{k-1})}{\|F_{k-1}\|^2\|F_k\|^2}F_k+\frac{F_k^T(F_k-F_{k-1})}{\|F_{k-1}\|^2}d_{k-1}\Bigg\|\\ &\leq \frac{2\|F_k\|}{\|F_{k-1}\|^2}\|d_{k-1}\|\|y_{k-1}\|+\|F_k\|\leq\frac{2LM_f}{\varepsilon_0^2}\|x_k\\ &\quad -x_{k-1}\|\|d_{k-1}\|+M_f.\end{aligned} \quad (35)$$

From (24), we know that there exist a positive integer k_1 and a positive number ε_1 $(0<\varepsilon_1<1)$ such that, for all $k\geq k_1$,

$$\frac{2LM_f}{\varepsilon_0^2}\|x_k-x_{k-1}\|<\varepsilon_1. \quad (36)$$

Hence,

$$\begin{aligned}\|d_k\| &\leq \varepsilon_1\|d_{k-1}\|+M_f\leq\varepsilon_1^2\|d_{k-2}\|+\varepsilon_1M_f+M_f\\ &\vdots\\ &\leq \varepsilon_1^{k_1}\|d_{k_1}\|+\varepsilon_1^{k_1-1}M_f+\cdots+\varepsilon_1M_f+M_f\leq\|d_{k_1}\|\\ &\quad +M_f\frac{1-\varepsilon_1^{k_1}}{1-\varepsilon_1}\leq\|d_{k_1}\|+\frac{M_f}{1-\varepsilon_1}.\end{aligned} \quad (37)$$

Let $M_1=\|d_{k_1}\|+M_f/(1-\varepsilon_1)$. Take

$$M=\max\{\|d_0\|,\|d_1\|,\ldots,\|d_{k_1}\|,M_1\}. \quad (38)$$

Then, $\|d_k\|\leq M$ holds for any positive integer k. □

Lemma 10. *Suppose that Assumptions 5, 6, and 7 hold. Let $\{x_k\}$ and $\{d_k\}$ be two sequences generated by Algorithm 1. Then, the line search rule (17) of Step 2 in Algorithm 1 is well-defined.*

Proof. Our aim is to show that the line search rule (17) terminates finitely with a positive step length α_k. By contradiction, suppose that, for some iterate indexes such as k^*, condition (17) does not hold. As a result, for all $m \in N$,

$$-F\left(x_{k^*} + s\rho^m d_{k^*}\right)^T d_{k^*} < \sigma s\rho^m \gamma_{k^*} \left\|d_{k^*}\right\|^2. \quad (39)$$

From (18) and the termination condition of Algorithm 1, it follows that, for all $m \in N$,

$$0 \le \varepsilon \le \gamma_{k^*} \le 1. \quad (40)$$

By taking the limit as $m \longrightarrow \infty$ in both sides of (39), we have

$$-F\left(x_{k^*}\right)^T d_{k^*} \le 0. \quad (41)$$

Equations (41) contradicts the fact that $-F_k^T d_k = -\|F_k\|^2 > 0$ for all k. That is to say, the line search rule terminates finitely with a positive step length α_k; i.e., the line search step of Algorithm 1 is well-defined. □

With the above preparation, we are now state the convergence result of Algorithm 1.

Theorem 11. *Suppose that Assumptions 5, 6, and 7 hold. Let $\{x_k\}$ be a sequence generated by Algorithm 1. Then,*

$$\liminf_{k\longrightarrow\infty} \left\|F_k\right\| = 0. \quad (42)$$

Proof. For the sake of contradiction, we suppose that the conclusion is not true. Then, by the definition of inferior limit, there exists a constant $\varepsilon_0 > 0$ such that, for any $k \in N$,

$$\left\|F_k\right\| \ge \varepsilon_0. \quad (43)$$

Consequently, from

$$\left\|F_k\right\|^2 = -F_k^T d_k \le \left\|F_k\right\| \left\|d_k\right\|, \quad (44)$$

it follows that $\|d_k\| \ge \varepsilon_0 > 0$ for any $k \in N$.

From (7), (17), and (40), we get

$$\begin{aligned}\left\|x_{k+1} - x_k\right\| &= \frac{\left|F\left(z_k\right)^T \left(x_k - z_k\right)\right|}{\left\|F\left(z_k\right)\right\|} = \frac{\left|\alpha_k F\left(z_k\right)^T d_k\right|}{\left\|F\left(z_k\right)\right\|} \\ &\ge \frac{\sigma\alpha_k^2 \gamma_k \left\|d_k\right\|^2}{\left\|F\left(z_k\right)\right\|} \ge \frac{\sigma\varepsilon\alpha_k^2 \left\|d_k\right\|^2}{M_f} \ge 0.\end{aligned} \quad (45)$$

Combining (24) and (45), we obtain

$$\lim_{k\longrightarrow\infty} \alpha_k \left\|d_k\right\| = 0. \quad (46)$$

Since $\|d_k\| \ge \varepsilon_0 > 0$ for any $k \in N$, we have

$$\lim_{k\longrightarrow\infty} \alpha_k = 0. \quad (47)$$

Clearly, $\alpha_k^* = \rho^{-1}\alpha_k$ does not satisfy in (17). It says that

$$-F\left(x_k + \alpha_k^* d_k\right)^T d_k < \sigma\alpha_k^* \gamma_k \left\|d_k\right\|^2. \quad (48)$$

By Lemmas 8 and 9, we know that the two sequences $\{x_k\}$ and $\{d_k\}$ are bounded. Without loss of generality, we choose a subset $K \in N$ such that

$$\begin{aligned}\lim_{k\longrightarrow\infty, k\in K} x_k &= x^*, \\ \lim_{k\longrightarrow\infty, k\in K} d_k &= d^*.\end{aligned} \quad (49)$$

Taking the limit in the two sides of (48) as $k \longrightarrow \infty$ ($k \in K$), it holds that

$$F\left(x^*\right)^T d^* \ge 0. \quad (50)$$

On the other hand, from (43), we know that

$$F\left(x_k\right)^T d_k = -\left\|F_k\right\|^2 \le -\varepsilon_0^2 < 0. \quad (51)$$

By taking the limit $k \longrightarrow \infty$ in the two sides of (51) for $k \in K$, we get

$$F\left(x^*\right)^T d^* \le -\varepsilon_0 < 0. \quad (52)$$

It contradicts (50). Thus, the proof of Theorem 11 has been completed. □

Remark 12 (only with γ_k being generated by (19)). As γ_k is determined by (18), the proofs are similar.

Since the proof of Theorem 11 does not involve differentiability of F, let alone nonsingularity of its Jacobian matrix, we know that the following result holds.

Corollary 13. *For any nonsmooth or singular function F, let $\{x_k\}$ be a sequence generated as Algorithm 1 is used to solve problem (1). Under Assumptions 5, 6, and 7, it holds that*

$$\liminf_{k\longrightarrow\infty} \left\|F_k\right\| = 0. \quad (53)$$

It should be pointed out that the global convergence of Algorithm 1 depends on assumption on monotonicity of F. For nonmonotonic function F, Algorithm 1 may be not applicable.

4. Numerical Experiments

In this section, by numerical experiments, we are going to study the effectiveness and robustness of Algorithm 1 for solving large-scale system of equations.

We first collect some benchmark test problems available in the literature.

Problem 14 (see [5]). The elements of $F(x)$ are given by

$$F_i(x) = 2x_i - \sin\left(x_i\right), \quad i = 1, 2, \ldots, n-1. \quad (54)$$

Problem 15 (see [5]). The elements of $F(x)$ are given by

$$F_i(x) = 2x_i - \left|\sin\left(x_i\right)\right|, \quad i = 1, 2, \ldots, n-1. \quad (55)$$

Problem 16 (see [15]). The elements of $F(x)$ are given by

$$\begin{aligned} F_1(x) &= x_1 - e^{\cos((x_1+x_2)/(n+1))}, \\ F_i(x) &= x_i - e^{\cos((x_{i-1}+x_i+x_{i+1})/(n+1))}, \quad i = 2,\ldots,n-1, \\ F_n(x) &= x_n - e^{\cos((x_{n-1}+x_n)/(n+1))}. \end{aligned} \tag{56}$$

Problem 17 (see [28]). The elements of $F(x)$ are given by

$$\begin{aligned} F(x) &= Ax + b, \\ b &= (-1,-1,-1,\ldots,-1)^T, \\ A &= \begin{pmatrix} \frac{5}{2} & 1 & & & \\ 1 & \frac{5}{2} & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & 1 \\ & & & 1 & \frac{5}{2} \end{pmatrix}. \end{aligned} \tag{57}$$

Clearly, problem 4 is a linear system of equations, which is used to test Algorithm 1 in this special case.

Problem 18 (see [15]). The elements of $F(x)$ are given by

$$F_i(x) = x_i - \frac{1}{n}x_i^2 + \frac{1}{n}\sum_{i=1}^{n} x_i + i, \quad i = 2,\ldots,n-1. \tag{58}$$

Problem 19 (see [15]). The elements of $F(x)$ are given by

$$\begin{aligned} F_1(x) &= \frac{1}{3}x_1^3 + \frac{1}{2}x_2^2, \\ F_i(x) &= -\frac{1}{2}x_i^2 + \frac{i}{3}x_i^3 + \frac{1}{2}x_{i+1}^2, \quad i = 2,\ldots,n-1, \\ F_n(x) &= -\frac{1}{2}x_n^2 + \frac{n}{3}x_n^3. \end{aligned} \tag{59}$$

Clearly, apart from problem 4, all the others are nonlinear system of equations. Problem 15 is nonsmooth at point $(0,0,\ldots,0)$. The size of all the test problems is variable. If this size is larger than 1000, the problem can be regarded to be large-scale. We solve all the test problems with different sizes by Algorithm 1, especially in comparison with the existing seven similar algorithms, such as those developed very recently in [10, 13–15, 28].

All the algorithms are coded in MATLAB R2010b and run on a personal computer with a 2.2GHZ CPU processor, 8GB memory, and Windows 10 operation system. The relevant parameters of algorithms are specified by

$$\begin{aligned} k_{max} &= 100000, \\ \sigma &= 0.3, \\ s &= 0.5, \\ \rho &= 0.7, \end{aligned} \tag{60}$$

$$\lambda_k = \begin{cases} \dfrac{\lambda_0}{2}, & \text{if } k = 1, \\ \dfrac{\lambda_{k-1} + \lambda_{k-2}}{2}, & \text{if } k > 1, \end{cases} \tag{61}$$

where $\lambda_0 = 1$. The termination condition is $\|F_k\| \le 10^{-4}$.

Numerical performance of all the algorithms is reported in Tables 1, 2, and 3. Table 1 shows the numerical performance of all the eight algorithms with the fixed initial points: $x_0 = (-1,\ldots,-1)^T$ for problems 1, 5, and 6; $x_0 = (1,\ldots,1)^T$ for problems 2 and 4; $x_0 = (10,\ldots,10)^T$ for problem 3. Table 2 demonstrates the numerical performance of all the eight algorithms with initial points randomly generated by Matlab's Code "rand(n,1)". Table 3 shows the numerical performance of our algorithm with dimension of 1000000.

For simplification of statement, we use the following notations:

Dim: the dimension of test problems.

Ni: the number of iterations.

Nf: the number of function evaluations.

MPPRP-M: the developed algorithm with γ_k determined by (19) in this paper.

MPPRP-W: the developed algorithm with γ_k generated by (18) in this paper.

MSDFPB: the modified spectral derivative-free projection-based algorithm in [13].

PRP: the PRP conjugate gradient derivative-free projection-based methods in [28].

MPRP: the modified PRP conjugate gradient derivative-free projection-based methods in [10].

TPRP: the two-term PRP conjugate gradient derivative-free projection-based methods in [10].

DFPB1: the steepest descent derivative-free projection-based methods in [14] with search direction as follows:

$$d_k = \begin{cases} -F_0, & \text{if } k = 0, \\ -F_k + \beta_k d_{k-1} + \theta_k, & \text{if } k > 0, \end{cases} \tag{62}$$

where $\beta_k = \beta_k^{PRP} = F_k^T y_{k-1}/\|F_{k-1}\|^2$, $\theta_k = (F_k^T y_{k-1})\|\omega_{k-1}\|^2/\|F_{k-1}\|^4$, $y_{k-1} = F_k - F_{k-1}$, and $\omega_{k-1} = \alpha_{k-1}d_{k-1}$.

Table 1: Numerical performance with fixed initial points.

Problem	Dim	Method	CPU-time	Ni	Nf
P1	10000	MPPRP-M	*0.049106*	*19*	*60*
		MPPRP-W	0.083986	28	132
		MSDFPB	0.207396	49	357
		PRP	0.210513	57	373
		MPRP	0.225433	49	357
		TPRP	0.205193	49	357
		DFPB1	0.208191	52	360
		DFPB2	0.180871	41	333
		MHS	0.201207	49	357
P1	20000	MPPRP-M	*0.097751*	*20*	*63*
		MPPRP-W	0.202214	34	187
		MSDFPB	0.538910	63	522
		PRP	0.601340	72	544
		MPRP	0.578528	63	522
		TPRP	0.567042	63	522
		DFPB1	0.564317	67	534
		DFPB2	0.538403	56	503
		MHS	0.568248	63	522
P1	50000	MPPRP-M	*0.189582*	*20*	*63*
		MPPRP-W	0.776513	47	317
		MSDFPB	2.349394	93	913
		PRP	2.426823	101	926
		MPRP	2.373229	93	913
		TPRP	2.32399	93	913
		DFPB1	2.421556	97	925
		DFPB2	2.364173	86	894
		MHS	2.601235	93	913
P1	100000	MPPRP-M	*0.519087*	*21*	*66*
		MPPRP-W	2.762991	61	478
		MSDFPB	9.002493	127	1384
		PRP	9.108308	135	1402
		MPRP	9.448788	127	1384
		TPRP	8.986777	127	1384
		DFPB1	9.100524	130	1391
		DFPB2	8.819755	119	1360
		MHS	9.419715	127	1384
P2	10000	MPPRP-M	*0.054535*	*19*	*60*
		MPPRP-W	0.088811	28	132
		MSDFPB	0.217590	49	357
		PRP	0.218645	57	373
		MPRP	0.226537	49	357
		TPRP	0.226429	49	357
		DFPB1	0.210470	52	360
		DFPB2	0.204578	41	333
		MHS	0.234442	49	357
P2	20000	MPPRP-M	*0.089917*	*20*	*63*
		MPPRP-W	0.198822	34	187
		MSDFPB	0.563358	63	522
		PRP	0.604772	72	544

Table 1: Continued.

Problem	Dim	Method	CPU-time	Ni	Nf
		MPRP	0.670294	63	522
		TPRP	0.677819	63	522
		DFPB1	0.679165	67	534
		DFPB2	0.607357	56	503
		MHS	0.589872	63	522
P2	50000	MPPRP-M	*0.191132*	*20*	*63*
		MPPRP-W	0.720108	47	317
		MSDFPB	2.248779	93	913
		PRP	2.464597	101	926
		MPRP	2.353173	93	913
		TPRP	2.342777	93	913
		DFPB1	2.363392	97	925
		DFPB2	2.279782	86	894
		MHS	2.384060	93	913
P2	100000	MPPRP-M	*0.516081*	*21*	*66*
		MPPRP-W	2.734834	61	478
		MSDFPB	9.004299	127	1384
		PRP	8.763470	135	1402
		MPRP	8.647855	127	1384
		TPRP	8.626593	127	1384
		DFPB1	8.626593	130	1391
		DFPB2	8.476836	119	1360
		MHS	8.905454	127	1384
P3	5000	MPPRP-M	*0.077881*	*22*	*69*
		MPPRP-W	0.955814	127	1360
		MSDFPB	2.155000	242	3141
		PRP	2.346950	251	3167
		MPRP	2.200878	242	3141
		TPRP	2.204338	242	3141
		DFPB1	2.246450	245	3148
		DFPB2	2.209059	234	3117
		MHS	2.236389	242	3141
P3	10000	MPPRP-M	*0.145481*	*23*	*72*
		MPPRP-W	2.590862	174	2059
		MSDFPB	6.749566	337	4751
		PRP	6.892930	346	4767
		MPRP	6.843273	337	4751
		TPRP	6.885266	337	4751
		DFPB1	7.222119	341	4763
		DFPB2	6.881919	330	4732
		MHS	6.992060	337	4751
P3	15000	MPPRP-M	*0.190225*	*23*	*72*
		MPPRP-W	4.894470	212	2656
		MSDFPB	12.242514	411	6059
		PRP	13.494894	420	6078
		MPRP	13.547676	411	6059
		TPRP	12.255626	411	6059
		DFPB1	12.473808	415	6071

Table 1: Continued.

Problem	Dim	Method	CPU-time	Ni	Nf
		DFPB2	12.734751	404	6040
		MHS	12.425026	411	6059
P3	20000	MPPRP-M	*0.239615*	*23*	*72*
		MPPRP-W	7.505229	242	3126
		MSDFPB	18.305460	472	7144
		PRP	19.195911	481	7169
		MPRP	19.482789	472	7144
		TPRP	19.011112	472	7144
		DFPB1	19.165800	475	7151
		DFPB2	19.428512	464	7120
		MHS	18.928398	472	7144
P4	10000	MPPRP-M	*0.061549*	*16*	*102*
		MPPRP-W	0.113153	19	131
		MSDFPB	0.289696	68	555
		PRP	0.187397	39	391
		MPRP	0.233670	48	462
		TPRP	0.358863	89	658
		DFPB1	0.657537	185	1142
		DFPB2	0.213256	40	422
		MHS	0.414446	62	803
P4	20000	MPPRP-M	*0.125760*	*17*	*108*
		MPPRP-W	0.256772	31	203
		MSDFPB	0.703061	79	730
		PRP	0.558050	52	565
		MPRP	0.663444	59	635
		TPRP	0.822848	100	833
		DFPB1	1.442003	203	1352
		DFPB2	0.565771	51	596
		MHS	0.674281	60	724
P4	50000	MPPRP-M	*0.295365*	*17*	*108*
		MPPRP-W	0.961757	38	273
		MSDFPB	2.972641	102	1109
		PRP	2.655601	73	943
		MPRP	2.738925	81	1007
		TPRP	3.209823	122	1207
		DFPB1	4.783178	234	1771
		DFPB2	2.644189	75	981
		MHS	6.106110	158	2717
P4	100000	MPPRP-M	*0.768978*	*17*	*108*
		MPPRP-W	3.113887	45	357
		MSDFPB	10.575505	129	1568
		PRP	9.164701	101	1406
		MPRP	9.756561	108	1465
		TPRP	11.151034	146	1651
		DFPB1	16.812871	275	2301
		DFPB2	9.292510	99	1423
		MHS	21.792651	179	3305
P5	100	MPPRP-M	*0.012253*	*18*	*77*
		MPPRP-W	0.018540	62	555
		MSDFPB	0.049702	129	1522
		PRP	0.033290	144	1558

Table 1: Continued.

Problem	Dim	Method	CPU-time	Ni	Nf
		MPRP	0.033017	139	1560
		TPRP	0.034758	128	1518
		DFPB1	0.026313	150	1601
		DFPB2	0.029466	131	1538
		MHS	0.040165	133	1539
P5	500	MPPRP-M	*0.016447*	*22*	*93*
		MPPRP-W	0.231399	544	9066
		MSDFPB	0.772128	1313	25232
		PRP	0.727796	1332	25274
		MPRP	0.585951	1372	25394
		TPRP	0.577368	1315	25238
		DFPB1	0.578794	1337	25315
		DFPB2	0.589438	1317	25247
		MHS	0.595316	1315	25241
P5	1000	MPPRP-M	*0.017438*	*23*	*97*
		MPPRP-W	1.072332	1505	29740
		MSDFPB	3.218997	3680	81797
		PRP	2.889154	3696	81825
		MPRP	2.963704	3735	81946
		TPRP	2.948045	3682	81802
		DFPB1	3.040253	3704	81880
		DFPB2	3.045603	3683	81803
		MHS	3.220028	3678	81791
P5	2000	MPPRP-M	*0.021613*	*25*	*106*
		MPPRP-W	5.944706	4206	95732
		MSDFPB	17.227153	10341	260347
		PRP	16.073363	10355	260379
		MPRP	16.960973	10394	260490
		TPRP	16.826907	10343	260352
		DFPB1	16.257104	10364	260424
		DFPB2	16.739385	10345	260356
		MHS	17.503252	10339	260342
P6	100	MPPRP-M	0.217224	1182	3717
		MPPRP-W	*0.193153*	*1148*	*3605*
		MSDFPB	0.237793	1186	3706
		PRP	0.220389	1207	3766
		MPRP	0.224211	1186	3658
		TPRP	0.248104	1195	3722
		DFPB1	0.236804	1201	3750
		DFPB2	0.235470	1170	3620
		MHS	0.236250	1198	3720
P6	500	MPPRP-M	0.944828	1348	5050
		MPPRP-W	0.794555	1278	4482
		MSDFPB	0.797832	1185	3968
		PRP	0.911934	1322	4267
		MPRP	0.845336	1258	3983
		TPRP	0.912343	1326	4335
		DFPB1	0.903852	1321	4300
		DFPB2	1.020228	1299	4143
		MHS	0.918611	1307	4147
P6	1000	MPPRP-M	2.337405	1495	6667
		MPPRP-W	1.680275	1381	4951
		MSDFPB	1.536904	1331	4320

Table 1: Continued.

Problem	Dim	Method	CPU-time	Ni	Nf
		PRP	1.851282	1389	4553
		MPRP	1.647356	1277	4040
		TPRP	1.686418	1273	4109
		DFPB1	1.792886	1360	4409
		DFPB2	1.812678	1343	4330
		MHS	1.775351	1341	4283
P6	2000	MPPRP-M	6.738143	1737	11016
		MPPRP-W	3.257538	1440	4936
		MSDFPB	3.065903	1381	4524
		PRP	3.574894	1390	4522
		MPRP	3.381889	1316	4247
		TPRP	3.565741	1378	4510
		DFPB1	3.609619	1377	4511
		DFPB2	3.647662	984	3232
		MHS	3.608315	1362	4397

DFPB2: the steepest descent derivative-free projection-based methods in [14] with θ_k in (62) being replaced by

$$\theta_k = \frac{F_k^T \omega_{k-1}}{\|F_{k-1}\|^2} + \frac{\left(F_k^T y_{k-1}\right)\|y_{k-1}\|^2}{\|F_{k-1}\|^4}. \tag{63}$$

MHS: the MHS-PRP conjugate gradient derivative-free projection-based methods in [15].

From the results in Tables 1, 2, and 3, it follows that our algorithm (MPPRP) outperforms the other seven algorithms, no matter how to choose the initial points (see the italicized results). Especially, it seems to more efficiently solve large-scale test problems. Actually, Table 3 shows that MPPRP-M can solve the first five Problems with dimension of 1000000 in less time, compared with the other algorithms.

In order to further measure the efficiency difference of all the eight algorithms, we calculate the average number of iteration, the average consumed CPU time, and their standard deviations, respectively. In Table 4, A-Ni and Std-Ni stand for the average number of iteration and its standard deviation, respectively. A-CT and Std-CT represent the average consumed CPU time and its standard deviation, respectively. The average number of function evaluation and its standard deviation are denoted by A-Nf and Std-Nf, respectively. Clearly, Std-Ni, Std-Nf, and Std-CT can show robustness of all the algorithms.

The results in Table 4 demonstrate that both of MPPRP-M and MPPRP-W outperform the other seven algorithms.

In the end of this section, we use our algorithm to solve a singular system of equations. The next test problem is a modified version of problem 1.

Problem 20. The elements of $F(x)$ are given by

$$F_i(x) = x_i - \sin(x_i), \quad i = 1, 2, \dots, n-1. \tag{64}$$

The initial point is fixed as $x_0 = (1, \dots, 1)^T$.

Note that problem 7 is singular since zero is its solution and the Jacobian matrix $F'(0)$ is singular. We implement Algorithm 1 to solve problem 7 with different dimensions to test whether it can find the singular solution or not.

The results in Table 5 indicate that Algorithm 1 is also efficient to solve the singular system of equations.

5. Preliminary Application in Compressed Sensing

In this section, we will apply our algorithm to solve an engineering problem originated from compressed sensing of sparse signals.

Let $A \in R^{m\times n}$ $(m \ll n)$ be a linear operator, let $\overline{x} \in R^n$ be a sparse or a nearly sparse original signal, and let $b \in R^m$ be an observed value which satisfies the following linear equations:

$$b = A\overline{x}, \tag{65}$$

The original signal $\overline{x}$ is desirable to be reconstructed from the linear equations $Ax = b$. Unfortunately, it is often that this linear equation is underdetermined or ill-conditioned in practice, and has infinitely many solutions. From the fundamental principles of compressed sensing, it is a reasonable approach to seek the sparsest one among all the solutions, which contains the least nonzero components.

In [2], it was shown that the compressed sensing of sparse signals can be formulated the following nonlinear system of equations:

$$\min\{z, (H+D)z + c\} = 0, \tag{66}$$

Table 2: Efficiency with random initial points.

Problem	Dim	Method	CPU-time	Ni	Nf
P1	100000	MPPRP-M	*0.489058*	*20*	*63*
		MPPRP-W	2.005818	42	268
		MSDFPB	4.565133	79	717
		PRP	4.757480	87	825
		MPRP	4.561151	79	796
		TPRP	4.631983	79	796
		DFPB1	4.763311	82	806
		DFPB2	4.610199	74	776
		MHS	4.879031	79	796
P2	100000	MPPRP-M	*0.483632*	*20*	*63*
		MPPRP-W	2.251160	42	268
		MSDFPB	4.375745	79	717
		PRP	4.561437	87	825
		MPRP	4.830143	79	796
		TPRP	4.560123	79	796
		DFPB1	4.427558	82	806
		DFPB2	4.550071	74	776
		MHS	4.428999	79	796
P3	100000	MPPRP-M	*1.187410*	*20*	*63*
		MPPRP-W	5.215137	46	306
		MSDFPB	10.136405	79	710
		PRP	10.808303	87	816
		MPRP	10.124779	79	789
		TPRP	10.068535	79	789
		DFPB1	10.372113	82	799
		DFPB2	9.458892	71	757
		MHS	10.137711	79	789
P4	100000	MPPRP-M	9.152258	253	*1306*
		MPPRP-W	11.371204	260	1423
		MSDFPB	13.374780	293	1837
		PRP	8.352463	170	1352
		MPRP	9.223706	184	1474
		TPRP	14.279675	302	2184
		DFPB1	15.329979	336	2386
		DFPB2	11.244725	231	1770
		MHS	17.738408	216	2902
P5	2000	MPPRP-M	*0.019793*	*25*	*106*
		MPPRP-W	7.148874	4207	95776
		MSDFPB	16.293374	10362	260980
		PRP	16.689685	10375	271376
		MPRP	16.112906	10415	271538
		TPRP	16.006655	10364	271349
		DFPB1	16.272868	10385	271442
		DFPB2	17.018004	10366	271355
		MHS	19.200453	10360	271335
P6	2000	MPPRP-M	9.722610	3415	15767
		MPPRP-W	*8.031480*	*3293*	12678
		MSDFPB	8.254000	3395	12524
		PRP	9.187302	3298	15178
		MPRP	9.667492	3366	15503
		TPRP	9.727760	3363	15708
		DFPB1	9.565437	3431	15990
		DFPB2	9.316936	3329	15590
		MHS	8.893926	3393	14502

Table 3: Efficiency for 1000000-dimension problems.

Problem	Dim	Method	CPU-time	Ni	Nf
P1	1000000	MPPRP-M	4.831098	22	69
P2	1000000	MPPRP-M	5.150450	22	69
P3	1000000	MPPRP-M	13.954404	26	81
P4	1000000	MPPRP-M	8.408822	19	120
P5	1000000	MPPRP-M	9.018623	36	154

Table 4: Average numerical efficiency and robustness of algorithms.

Method	(A-CT, Std-CT)	(A-Ni, Std-Ni)	(A-Nf, Std-CT)
MPPRP-M	(*1.1657*, *2.5775*)	(*330.7*, *763.88*)	(*1513.5*, *3652.3*)
MPPRP-W	(2.5725, 2.9258)	(689.4, 1199.35)	(9205.9, 24233.7)
MSDFPB	(5.4010, 5.6632)	(1244.56, 2638.61)	(23143.6, 66309.9)
PRP	(5.3178, 5.5599)	(1247.73, 2641.15)	(23583.0, 67602.4)
MPRP	(5.3717, 5.6475)	(1243.36, 2654.63)	(23574.3, 67649.2)
TPRP	(5.5309, 5.7401)	(1249.63, 2636.94)	(23633.4, 67579.6)
DFPB1	(5.8718, 6.0331)	(1275.9, 2636.85)	(23751.2, 67568.5)
DFPB2	(5.3586, 5.6541)	(1223.93, 2642.11)	(23543.4, 67612.0)
MHS	(6.2549, 6.7712)	(1248.8, 2637.78)	(23719.2, 67539.8)

where

$$\begin{aligned} z &= \begin{bmatrix} u \\ v \end{bmatrix}, \\ y &= A^T b, \\ c &= \tau e_{2n} + \begin{bmatrix} -y \\ y \end{bmatrix}, \\ H &= \begin{bmatrix} A^T A & -A^T A \\ -A^T A & A^T A \end{bmatrix}, \\ D &= \begin{bmatrix} 0 & \delta E_n \\ 0 & 0 \end{bmatrix}. \end{aligned} \tag{67}$$

where $u \in R^n$, $v \in R^n$, and $u_i = (x_i)_+$, $v_i = (-x_i)_+$ for all $i = 1, 2, \ldots, n$ with $(\cdot)_+ = \max\{0, \cdot\}$, and e_{2n} is an $2n$-dimensional vector with all elements being one. Clearly, (66) is nonsmooth.

Implement Algorithm 1 to solve the compressed sensing problem of sparse signals with different compressed ratios. In this experiment, we consider a typical compressive sensing scenario, where the goal is to reconstruct an n-length sparse signal from m observations. We test the three algorithms (two popular algorithms: CGD in [34] and SGCS in [35], and MPPRP-M) under three CS ratios (CS-R): $m/n = 0.125, 0.25, 0.5$ corresponding to different numbers of measurements with $n = 2048$, respectively. The original sparse signal contains 32(64) randomly nonzero elements. The measurement b is disturbed by noise, i.e., $b = A\overline{x} + \omega$, where ω is the Gaussian noise distributed as $N(0, \sigma^2 I)$ with $\sigma^2 = 10^{-4}$. A is the Gaussian matrix generated by commend randn(m, n) in Matlab. The quality of restoration is measured by the mean of squared error (MSE) to the original signal $\overline{x}$; that is,

$$\mathrm{MSE} = \frac{1}{n} \left\| \overline{x} - x^* \right\|^2, \tag{68}$$

where x^* is the restored signal. The iterative process starts at the measurement image, i.e., $x_0 = A^T b$, and terminates when the relative change between successive iterates falls below 10^{-5}. It says that

$$\frac{\left\| f_k - f_{k-1} \right\|}{\left\| f_{k-1} \right\|} < 10^{-5}, \tag{69}$$

where $f_k = \tau \|x_k\|_1 + (1/2)\|A x_k - b\|_2^2$ and $\tau = 0.005\|A^T b\|_\infty$ is chosen as suggested in [36]. The other parameters in this experiment are same as those in the numerical experiments conducted in Section 4. Numerical efficiency is shown in Table 6.

Clearly, from the results in Table 6, it follows that, for any CS ratio, MPPRP-M can recover the sparse signals more efficiently without reduction of recovery quality (see the italicized results in Table 6). If the sparsity level 32 is replaced by 64 in the 2048-length original signal, the corresponding results are also presented in Table 6, denoted by $(\cdot)$.

6. Conclusions

In this paper, we have presented a modified spectral PRP conjugate gradient derivative-free projection-based method for solving the large-scale nonlinear monotonic equations, where the search direction is proved to be sufficiently descent for any line search rule, and the step length is chosen by a line search which can overcome the difficulty of choosing an appropriate weight.

Table 5: Numerical performance in singular case.

Problem	Dim	Method	CPU-time	Ni	Nf
P7	500	MPPRP-M	0.907980	*6684*	*20055*
		MPPRP-W	0.823281	6684	20055
		MSDFPB	1.063594	6684	20055
		PRP	0.801892	6695	20088
		MPRP	0.856992	6684	20055
		TPRP	0.892188	6684	20055
		DFPB1	0.906388	6690	20073
		DFPB2	0.894474	6684	20055
		MHS	1.045179	6684	20055
P7	1000	MPPRP-M	*1.742640*	*8424*	*25275*
		MPPRP-W	1.787419	8424	25275
		MSDFPB	2.140070	8424	25275
		PRP	1.773320	8435	25308
		MPRP	1.878275	8424	25275
		TPRP	1.856212	8424	25275
		DFPB1	1.855210	8430	25293
		DFPB2	1.869769	8424	25275
		MHS	2.085814	8424	25275
P7	2000	MPPRP-M	*3.712167*	*10616*	*31851*
		MPPRP-W	4.182644	10616	31851
		MSDFPB	4.622078	10617	31855
		PRP	4.465109	10628	31888
		MPRP	4.716474	10617	31855
		TPRP	5.772379	10617	31855
		DFPB1	5.357838	10622	31870
		DFPB2	4.663553	10616	31852
		MHS	5.737888	10617	31855
P7	5000	MPPRP-M	*20.734981*	*14412*	*43239*
		MPPRP-W	21.112604	14413	43243
		MSDFPB	23.963617	14414	43252
		PRP	22.231907	14425	43283
		MPRP	23.381419	14414	43252
		TPRP	23.871918	14414	43252
		DFPB1	23.222197	14420	43269
		DFPB2	23.300951	14414	43252
		MHS	25.432421	14414	43252

Table 6: Recovering sparse signals under different CS ratios.

CS-R	CGD			SGCS			MPPRP-M		
	Iter	Time	MSE	Iter	Time	MSE	Iter	Time	MSE
0.125	1047	4.97s	7.96e-006	752	3.44s	4.40e-006	*607*	*2.58s*	*4.18e-006*
	(985)	(4.86s)	(5.11e-003)	(1018)	(4.19s)	(4.29e-003)	*886*	*(3.63s)*	*(4.16e-003)*
0.25	263	2.05s	2.87e-006	241	1.92s	2.75e-006	*178*	*1.38s*	*2.64e-006*
	(425)	(3.34s)	(7.12e-006)	(359)	(2.84s)	(5.70e-006)	*(274)*	*(2.20s)*	*(5.54e-006)*
0.5	97	1.33s	2.46e-006	135	1.75s	1.59e-006	*85*	*1.16s*	*1.54e-006*
	(95)	(1.33s)	(1.14e-005)	(146)	(2.03s)	(5.52e-006)	*(93)*	*(1.30s)*	*(5.37e-006)*

Under mild assumptions, global convergence theory has been established for the developed algorithm. Since our algorithm does not involve computing the Jacobian matrix or its approximation, both information storage and computational cost of the algorithm are lower. That is to say, our algorithm is helpful to solution of large-scale problems. In addition, it has been shown that our algorithm is also applicable to solve a nonsmooth system of equations or a singular one.

Numerical tests have demonstrated that our algorithm outperforms the others by costing less number of function evaluations, less number of iterations, or less CPU time to find a solution with the same tolerance, especially in comparison with some similar algorithms available in the literature. Efficiency of our algorithm has also been shown by reconstructing sparse signals in compressed sensing.

In future research, due to its satisfactory numerical efficiency, the proposed method in this paper can be extended into solving more large-scale nonlinear system of equations from many fields of sciences and engineering.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

We declare that all the authors have no any conflicts of interest about submission and publication of this paper.

Authors' Contributions

Zhong Wan conceived and designed the research plan and wrote the paper. Jie Guo performed the mathematical modelling and numerical analysis and wrote the paper.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (Grant no. 71671190) and Opening Foundation from State Key Laboratory of Developmental Biology of Freshwater Fish, Hunan Normal University.

References

[1] S. Huang and Z. Wan, "A new nonmonotone spectral residual method for nonsmooth nonlinear equations," *Journal of Computational and Applied Mathematics*, vol. 313, pp. 82–101, 2017.

[2] Z. Wan, J. Guo, J. Liu, and W. Liu, "A modified spectral conjugate gradient projection method for signal recovery," *Signal, Image and Video Processing*, vol. 12, no. 8, pp. 1455–1462, 2018.

[3] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Siam, New York, NY, USA, 1970.

[4] J. E. Dennis and B. R. Schnabel, "Numerical methods for nonlinear equations and unconstrained optimization," *Classics in Applied Mathematics*, p. 16, 1983.

[5] W. Zhou and D. Li, "Limited memory BFGS method for nonlinear monotone equations," *Journal of Computational Mathematics*, vol. 25, no. 1, pp. 89–96, 2007.

[6] J. M. Martinez, "A family of quasi-Newton methods for nonlinear equations with direct secant updates of matrix factorizations," *SIAM Journal on Numerical Analysis*, vol. 27, no. 4, pp. 1034–1049, 1990.

[7] G. Fasano, F. Lampariello, and M. Sciandrone, "A truncated nonmonotone Gauss-Newton method for large-scale nonlinear least-squares problems," *Computational Optimization and Applications*, vol. 34, no. 3, pp. 343–358, 2006.

[8] D. Li and M. Fukushima, "A globally and superlinearly convergent Gauss-Newton-based BFGS method for symmetric nonlinear equations," *SIAM Journal on Numerical Analysis*, vol. 37, no. 1, pp. 152–172, 1999.

[9] Z. Papp and S. Rapajić, "FR type methods for systems of large-scale nonlinear monotone equations," *Applied Mathematics and Computation*, vol. 269, pp. 816–823, 2015.

[10] Q. Li and D.-H. Li, "A class of derivative-free methods for large-scale nonlinear monotone equations," *IMA Journal of Numerical Analysis (IMAJNA)*, vol. 31, no. 4, pp. 1625–1635, 2011.

[11] L. Zhang, W. Zhou, and D. H. Li, "A descent modified Polak-Ribiere-Polyak conjugate gradient method and its global convergence," *IMA Journal of Numerical Analysis (IMAJNA)*, vol. 26, no. 4, pp. 629–640, 2006.

[12] W. La Cruz, J. M. Martnez, and M. Raydan, "Spectral residual method without gradient information for solving large-scale nonlinear systems of equations," *Mathematics of Computation*, vol. 75, no. 255, pp. 1429–1448, 2006.

[13] Z. Wan, W. Liu, and C. Wang, "A modified spectral conjugate gradient projection method for solving nonlinear monotone symmetric equations," *Pacific Journal of Optimization. An International Journal*, vol. 12, no. 3, pp. 603–622, 2016.

[14] M. Ahookhosh, K. Amini, and S. Bahrami, "Two derivative-free projection approaches for systems of large-scale nonlinear monotone equations," *Numerical Algorithms*, vol. 64, no. 1, pp. 21–42, 2013.

[15] Q.-R. Yan, X.-Z. Peng, and D.-H. Li, "A globally convergent derivative-free method for solving large-scale nonlinear monotone equations," *Journal of Computational and Applied Mathematics*, vol. 234, no. 3, pp. 649–657, 2010.

[16] K. Amini, A. Kamandi, and S. Bahrami, "A double-projection-based algorithm for large-scale nonlinear systems of monotone equations," *Numerical Algorithms*, vol. 68, no. 2, pp. 213–228, 2015.

[17] Y. Li, Z. Wan, and J. Liu, "Bi-level programming approach to optimal strategy for vendor-managed inventory problems under random demand," *The ANZIAM Journal*, vol. 59, no. 2, pp. 247–270, 2017.

[18] T. Li and Z. Wan, "New adaptive Barzilar-Borwein step size and its application in solving large scale optimization problems," *The ANZIAM Journal*, vol. 61, no. 1, pp. 76–98, 2019.

[19] X. J. Tong and L. Qi, "On the convergence of a trust-region method for solving constrained nonlinear equations with degenerate solutions," *Journal of Optimization Theory and Applications*, vol. 123, no. 1, pp. 187–211, 2004.

[20] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly of Applied Mathematics*, vol. 2, pp. 164–168, 1944.

[21] D. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *SIAM Journal on Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[22] C. Kanzow, N. Yamashita, and M. Fukushima, “Levenberg-Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints,” *Journal of Computational and Applied Mathematics*, vol. 173, no. 2, pp. 321–343, 2005.

[23] J. K. Liu and S. J. Li, “A projection method for convex constrained monotone nonlinear equations with applications,” *Computers & Mathematics with Applications*, vol. 70, no. 10, pp. 2442–2453, 2015.

[24] G. Yuan and M. Zhang, “A three-terms Polak-Ribière-Polyak conjugate gradient algorithm for large-scale nonlinear equations,” *Journal of Computational and Applied Mathematics*, vol. 286, pp. 186–195, 2015.

[25] G. Yuan, Z. Meng, and Y. Li, “A modified Hestenes and Stiefel conjugate gradient algorithm for large-scale nonsmooth minimizations and nonlinear equations,” *Journal of Optimization Theory and Applications*, vol. 168, no. 1, pp. 129–152, 2016.

[26] G. Yuan and W. Hu, “A conjugate gradient algorithm for large-scale unconstrained optimization problems and nonlinear equations,” *Journal of Inequalities and Applications*, vol. 2018, no. 1, article 113, 2018.

[27] L. Zhang and W. Zhou, “Spectral gradient projection method for solving nonlinear monotone equations,” *Journal of Computational and Applied Mathematics*, vol. 196, no. 2, pp. 478–484, 2006.

[28] W. Cheng, “A PRP type method for systems of monotone equations,” *Mathematical and Computer Modelling*, vol. 50, no. 1-2, pp. 15–20, 2009.

[29] M. V. Solodov and B. F. Svaiter, “A globally convergent inexact Newton method for systems of monotone equations,” in *Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods*, pp. 355–369, Springer, Boston, MA, USA, 1998.

[30] Z. Wan, Z. Yang, and Y. Wang, “New spectral PRP conjugate gradient method for unconstrained optimization,” *Applied Mathematics Letters*, vol. 24, no. 1, pp. 16–22, 2011.

[31] Y. Ou and J. Li, “A new derivative-free SCG-type projection method for nonlinear monotone equations with convex constraints,” *Applied Mathematics and Computation*, vol. 56, no. 1-2, pp. 195–216, 2018.

[32] W. Zhou and D. Li, “On the Q-linear convergence rate of a class of methods for monotone nonlinear equations,” *Pacific Journal of Optimization*, vol. 14, pp. 723–737, 2018.

[33] B. T. Polyak, *Introduction to Optimization*, Optimization Software, New York, NY, USA, 1987.

[34] Y. Xiao and H. Zhu, “A conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing,” *Journal of Mathematical Analysis and Applications*, vol. 405, no. 1, pp. 310–319, 2013.

[35] Y. Xiao, Q. Wang, and Q. Hu, “Non-smooth equations based method for l_1 problems with applications to compressed sensing,” *Nonlinear Analysis. Theory, Methods and Applications*, vol. 74, no. 11, pp. 3570–3577, 2011.

[36] S. Kim, K. Koh, M. Lustig et al., “A method for large-scale 1-regularized least squares problems with applications in signal processing and statistics,” Technical Report, Dept. of Electrical Engineering, Stanford University, Stanford, Calif, USA, 2007.