An Accelerated Conjugate Gradient Algorithm for Solving Nonlinear Monotone Equations and Image Restoration Problems

Haishan Feng and Tingting Li

Business, College of Mathematics and Information Science, Guangxi University, Nanning, Guangxi, China

Correspondence should be addressed to Tingting Li; ltt@st.gxu.edu.cn

Received 6 July 2020; Accepted 25 August 2020; Published 5 October 2020

Copyright © 2020 Haishan Feng and Tingting Li. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Combining the three-term conjugate gradient method of Yuan and Zhang and the acceleration step length of Andrei with the hyperplane projection method of Solodov and Svaiter, we propose an accelerated conjugate gradient algorithm for solving nonlinear monotone equations in this paper. The presented algorithm has the following properties: (i) All search directions generated by the algorithm satisfy the sufficient descent and trust region properties independent of the linesearch technique. (ii) A derivative-free search technique is proposed along the direction to obtain the steplength \( \alpha_k \). (iii) If \( \phi_k = -\alpha_k (h_k - h(\omega_j))^T d_k > 0 \), then an acceleration scheme is used to modify the step length in a multiplicative manner and create a point. (iv) If the point satisfies the given condition, then it is the next point; otherwise, the hyperplane projection technique is used to obtain the next point. (v) The global convergence of the proposed algorithm is established under some suitable conditions. Numerical comparisons with other conjugate gradient algorithms show that the accelerated computing scheme is more competitive. In addition, the presented algorithm can also be applied to image restoration.

1. Introduction

In this paper, the following nonlinear equation is considered:

\[
h(x) = 0, \text{ subject to } x \in \mathbb{R}^n,
\]

where \( h: \mathbb{R}^n \rightarrow \mathbb{R}^n \) is continuous and monotone, and \( h(x) \) satisfies

\[
(h(x) - h(y))^T (x - y) \geq 0, \forall x, y \in \mathbb{R}^n.
\]

It is not difficult to show that the solution set of monotone equation (1), unless empty, is convex. This problem has many significant applications in applied mathematics, economics, and engineering. For example, the economic equilibrium problem [1] can be transformed into problem (1). Generally, an iteration formula generates the next iteration point by

\[
x_{k+1} = x_k + \alpha_k d_k,
\]

where \( \alpha_k \) is the step length and \( d_k \) is a search direction, which are two important factors for solving nonlinear equations.

Some derivative-free line search techniques [2–5] were proposed to search for step length \( \alpha_k \). Li and Li [6] presented a derivative-free line search to find \( \alpha_k = \max_i \{ \rho s^T i: i = 0, 1, \ldots, 4 \} \) such that

\[
-h(x_k + \alpha_k d_k)^T d_k \geq \lambda \| h(x_k + \alpha_k d_k) \| d_k^2.
\]

where \( \rho, \lambda > 0 \) and \( s \in (0, 1) \). \( \| \cdot \| \) represents the Euclidean norm. The line search technique (4) is different from other existing derivative-free line search techniques because it does not use a merit function. If \( d_k \) satisfies \( h(x_k)^T d_k < 0 \), the inequality (4) holds for all sufficiently small \( \alpha_k > 0 \). As a result, \( \alpha_k \) can be obtained by some backtracking processes.

For \( d_k \), it is well known that the Newton methods [7, 8], quasi-Newton methods [9–13], spectral gradient methods [14, 15], and conjugate gradient methods [16, 17] can deal with large-scale nonlinear equations. For solving large-scale optimization problems, the conjugate gradient methods are quite effective since they only calculate and store the gradient value of the objective function. Many scholars have applied conjugate gradient theory to solve nonlinear
monotone equations and have achieved good results [17–19]. Classical conjugate gradient methods include HS method [20], FR method [21], PRP method [22, 23], LS method [24], CD method [25], and DY method [26]. In particular, the PRP method, as one of the most effective methods, generates a small step near a minimum point; then, the subsequence generated for the search direction $d_k$ will automatically approach the negative gradient direction, which avoids continuously generating a small step. However, the global convergence of the PRP method is not established under inexact line search techniques for general functions. Many scholars have performed continuous research and have reached satisfactory conclusions. Zhang [27] proposed the MPRP method, where $d_k$ is designed as follows:

$$d_k = \begin{cases} -h_k + \rho_k^{\text{PRP}} y_k, & \text{if } k \geq 1, \\
-h_k, & \text{if } k = 0, \end{cases}$$

(5)

where $h(x_k) = h_k, h(x_{k-1}) = h_{k-1}, y_k = h_k - h_{k-1}, \rho_k^{\text{PRP}} = h_k^T y_k / \|h_k\|^2, \text{and } \theta_k = h_k^T d_{k-1} / \|h_{k-1}\|^2$. It is easy to obtain from (5) that

$$d_k^T h_k = -\|h_k\|^2.$$  

(6)

The above equation indicates that $d_k$ is a descent direction of $h$ at $x_k$. If the exact line search is used, then we have $\theta_k = 0$. Consequently, formula (5) is inferred to be the standard PRP method. Under some mild conditions, the MPRP method is globally convergent under the Armijo-type line search, but global convergence cannot be established under the weak Wolfe–Powell line search. Andrei [35] presented an acceleration scheme that modifies the step length $\alpha_k$ in a multiplicative manner to improve the reduction of the function values along the iterations. The step length is defined as follows:

$$\tilde{\alpha}_k = \xi_k \alpha_k,$$  

(10)

where $\xi_k = -\varphi_k / \varphi_k, \varphi_k = \alpha_k h_k^T d_k, \varphi_k = -\alpha_k (h_k - h(w_k))^T d_k$. If $\varphi_k > 0$, let $\alpha_k = \tilde{\alpha}_k$. A numerical comparison with some conjugate gradient algorithms shows that the computational scheme is effective.

Inspired by the above discussions, we proposed an accelerated conjugate gradient algorithm that combines the TTPRP method, the acceleration step length, and the hyperplane projection method. The main contributions of the algorithm are as follows:

An accelerated conjugate gradient algorithm is introduced for solving nonlinear monotone equations

All search directions of the algorithm satisfy the sufficient descent condition

All search directions of the algorithm belong to a trust region

The global convergence of the presented algorithm is proved

The numerical results show that the proposed algorithm is more effective for nonlinear monotone equations

The algorithm can be applied to restore an original image from an image damaged by impulse noise

This paper is organized as follows: in the next section, we discuss the ATTPRP algorithm and global convergence analysis. In Section 3, we report the preliminary numerical experiments to show that the algorithm is efficient for nonlinear monotone equations and applicable to image restoration problems. In Section 4, the conclusion regarding the proposed algorithm is given.
2. Accelerated Algorithm and Convergence Analysis

In this section, we will propose an accelerated algorithm and prove its global convergence. The steps of the given algorithm are as follows.

2.1. Accelerated Three-Term PRP Conjugate Gradient (ATTPRP) Algorithm

Step 0: choose any \( x_0 \in \mathbb{R}^n \) as the initial point and constants \( s \in (0, 1), \rho > 0, \lambda > 0, \kappa > 0, \) and \( \varepsilon \in (0, 1) \), let \( k^0 = 0 \).

Step 1: stop if \( \|h_k\| \leq \varepsilon \). Otherwise, compute \( d_k \) by using formula (5).

Step 2: choose \( \alpha_k \) satisfying the inequality (4).

Step 3: if \( \phi_k > 0 \), then \( \alpha_k = \overline{\alpha}_k \).

Step 4: let the next iterative value be \( x_k + \alpha_k d_k \).

Step 5: if \( \|h_{k+1}\| \leq \varepsilon \), stop and set \( x_{k+1} = x_k \). Otherwise, determine \( x_{k+1} \) using formula (9).

Step 6: let \( k^* = k + 1 \). Go to Step 1.

The following lemma shows that the search direction \( d_k \) designed by using formula (7) has not only the sufficient descent property but also the trust region property independent of the line search.

**Lemma 1.** \( d_k \) is defined by using formula (7); then, we obtain

\[
d_k^T h_k = -\|h_k\|^2, \tag{11}
\]

\[
\|h_k\| \leq \|d_k\| \leq \left(1 + \frac{2}{\kappa}\right)\|h_k\|. \tag{12}
\]

**Proof.** If \( k \geq 0 \), formulas (7) and (12) are obviously true. If \( k \geq 1 \), we obtain from formula (7) that

\[
h_k^T d_k = h_k^T \left[-h_k + \frac{h_k^T y_k d_{k-1} - h_k^T d_{k-1} y_k}{\max\{\|d_{k-1}\| \|y_k\|, \|h_{k-1}\|^2\}}\right]
\]

\[
= -\|h_k\|^2 + \frac{h_k^T y_k d_{k-1}^T h_k - h_k^T d_{k-1} y_k h_k^T}{\max\{\|d_{k-1}\| \|y_k\|, \|h_{k-1}\|^2\}} = -\|h_k\|^2. \tag{13}
\]

In addition, by formula (7), we get \( \|d_k\| \geq \|h_k\| \) and

\[
\|d_k\| = \left\| -h_k + \frac{h_k^T y_k d_{k-1} - h_k^T d_{k-1} y_k}{\max\{\|d_{k-1}\| \|y_k\|, \|h_{k-1}\|^2\}} \right\|
\]

\[
\leq \|h_k\| + \frac{\|h_k\| \|y_k\| \|d_{k-1}\| + \|h_k\| \|d_{k-1}\| \|y_k\|}{\max\{\|d_{k-1}\| \|y_k\|, \|h_{k-1}\|^2\}} \tag{14}
\]

\[
\leq \|h_k\| + \frac{\|h_k\| \|y_k\| \|d_{k-1}\|}{\|d_{k-1}\| \|y_k\|} = \left(1 + \frac{2}{\kappa}\right)\|h_k\|,
\]

where \( \max\{\|d_{k-1}\| \|y_k\|, \|h_{k-1}\|^2\} \geq \kappa \|d_{k-1}\| \|y_k\| \). Then, the proof is completed.

The following assumption need to be established in order to study some properties of the ATTPRP algorithm:

**Assumption 1.**

(i) The solution set of the problem (1) is nonempty.

(ii) The function \( h(x) \) is Lipschitz continuous on \( \mathbb{R}^n \); that is, there exists a positive constant \( K \) satisfying

\[
\|h(x) - h(y)\| \leq K \|x - y\|, \quad \forall x, y \in \mathbb{R}^n. \tag{15}
\]

**Remark 1.** Assumption 1(ii) implies that \( \|h_k\| \) is bounded; then, there exists a constant \( \varphi \) such that

\[
\|h_k\| \leq \varphi. \tag{16}
\]

In the following paper, if not specifically stated, we always assume that the conditions in Assumption 1 hold.

**Lemma 2.** Let \( \{x_k\} \) and \( \{w_k\} \) be generated by the ATTPRP algorithm. The step length \( \alpha_k \) generated by the ATTPRP algorithm satisfies

\[
\alpha_k \geq \min\left\{ \rho_0, \frac{s}{\lambda K + \lambda h(w'_k) \|d_k\|^2} \right\}, \tag{17}
\]

where \( w'_k = x_k + \alpha'_k d_k \) and \( \alpha'_k = \alpha_k s^{-1} \).

**Proof.** By line search (4), assuming \( \alpha_k \neq \rho \), let \( \alpha'_k = \alpha_k s^{-1} \), by the definition \( \alpha'_k \), does not satisfy the line search (4). That is,

\[
-h(w'_k)^T d_k < \lambda\alpha'_k \|h(w'_k)\| \|d_k\|^2. \tag{18}
\]

Since \( h(x) \) is Lipschitz continuous and by using formula (11), we have

\[
\|h_k\|^2 = -h_k^T d_k = (h(w'_k) - h_k - h(w'_k))^T d_k
\]

\[
= (h(w'_k) - h_k)^T d_k - h(w'_k)^T d_k
\]

\[
\leq K \|w'_k - x_k\| \|d_k\| + \lambda\alpha'_k \|h(w'_k)\| \|d_k\|^2
\]

\[
= K \alpha'^2 \|d_k\|^2 + \lambda\alpha'_k \|h(w'_k)\| \|d_k\|^2
\]

\[
= \alpha'^2 (K + \lambda \|h(w'_k)\|) \|d_k\|^2
\]

\[
= \alpha_k s^{-1} (K + \lambda \|h(w'_k)\|) \|d_k\|^2,
\]

namely,

\[
\alpha_k \geq \frac{s}{\lambda K + \lambda \|h(w'_k)\| \|d_k\|^2}. \tag{20}
\]

This leads to the ideal inequality (17). The proof is completed.

The following lemma is similar to Lemma 1 in the study of Solodov and Svaiter [34], which also holds for the
ATTTPRP algorithm. Therefore, we only state it as follows but omit its proof.

\[\text{Lemma 3. Let the sequence } \{x_k\} \text{ be generated by using the ATTPRP algorithm. Suppose that } x \text{ is a solution of problem (1) with } h(x) = 0. \text{ We obtain}\]
\[\|x_{k+1} - x\|^2 \leq \|x_k - x\|^2 - \|x_{k+1} - x_k\|^2. \] (21)

In particular, the sequence \(|x_k|\) is bounded, and
\[\sum_{k=0}^{\infty} \|x_{k+1} - x_k\|^2 < \infty. \] (22)

\[\text{Remark 2. The above lemma reveals that the distance from the iterative points to the solution set of the problem (1) decreases along iterations. Otherwise, for any } k, \text{ it is followed from formulas (9) and (4) that}\]
\[\|x_{k+1} - x_k\| = \frac{|h(w_k)^T (x_k - w_k)|}{\|h(w_k)\|} = \frac{-\alpha_k h(w_k)^T d_k}{\|h(w_k)\|^2} \geq \lambda \alpha_k^2 \|d_k\|^2. \] (23)

Particularly, we obtain
\[\lim_{k \to \infty} \alpha_k \|d_k\| = 0. \] (24)

In the following part, the global convergence and the strong global convergence properties of the ATTPRP algorithm will be proven.

\[\text{Theorem 1. Let } \{x_k\} \text{ be generated by using the ATTPRP algorithm. Then, we have}\]
\[\lim_{k \to \infty} \inf \|h_k\| = 0. \] (25)

\[\text{Proof. We will prove this theorem by contradiction. Suppose that the equation (25) does not hold, there exists a constant } \eta > 0 \text{ such that } \|h_k\| \geq \eta \text{ holds for all } k. \text{ From formula (12), we have}\]
\[\|d_k\| \geq \|h_k\| \geq \eta, \forall k \geq 0. \] (26)

According to Lemma 3 and equation (24), the sequences \(|x_k|\) and \(|w_k|\) are bounded. By formulas (7) and (16), for all \(k \geq 1\), we obtain
\[\|d_k\| \leq \|h_k\| + \frac{h_k^T y_k d_{k-1} - h_k^T d_k y_k}{\max \left\{ \kappa \|d_{k-1}\| \|y_k\|, \|h_{k-1}\|^2 \right\}} \]
\[\leq \|h_k\| + \frac{\|h_k\| \|y_k\| \|d_{k-1}\| + \|h_k\| \|d_{k-1}\| \|y_k\|}{\max \left\{ \kappa \|d_{k-1}\| \|y_k\|, \|h_{k-1}\|^2 \right\}} \]
\[\leq \|h_k\| + \|h_k\| \|y_k\| \|d_{k-1}\| + \|h_k\| \|d_{k-1}\| \|y_k\| \]
\[\|h_{k-1}\|^2 \]
\[\leq \phi + 2 \phi \eta^{-2} K \|x_k - x_{k-1}\| \|d_{k-1}\|. \] (27)

where \(\max \{\kappa \|d_{k-1}\| \|y_k\|, \|h_{k-1}\|^2\} \geq \|h_{k-1}\|^2\). Since \(|x_{k+1} - x_k|\) converges to zero, the last inequality shows that \(|d_k|\) is bounded. By formulas (12) and (16), we obtain
\[\|d_k\| \leq M, \] (28)
where \(M = 1 + 2 \kappa \phi\). Thus, from the formulas (16) and (17), we obtain that
\[\alpha_k \|d_k\| \geq \rho \frac{s}{\phi} \left( \frac{h_k^2}{\|h_k\|^2} \right) \|d_k\| \]
\[\geq \min \left\{ \rho \eta, \frac{2 \eta^2}{\phi (1 + \lambda \phi) M} \right\} > 0. \] (29)

This contradicts with formula (24). Consequently, the proof is completed.

The following theorem indicates the strong global convergence of the ATTPRP algorithm, which is similar to Theorem 1 in [6]. We also give a specific proof for convenience of understanding.

\[\text{Theorem 2. Let } \{x_k\} \text{ be generated by using the ATTPRP algorithm. Then, the whole sequence } \{x_k\} \text{ converges to a solution of the problem (1).}\]

\[\text{Proof. Theorem 1 shows that there exists a subsequence of } \{x_k\} \text{ converging to a solution } x \text{ of the problem (1). On the other side, it follows from Lemma 3 that the sequence } \{|x_{k+1} - x|\} \text{ converges. Therefore, the whole sequence } \{x_k\} \text{ converges to } x. \]

\[\text{3. Numerical Experiments}\]

In this section, the numerical experiments will be divided into two parts for illustration. The first subsection involves normal nonlinear equations, and the second subsection describes image restoration problems. All tests in this section are coded in MATLAB R2017a, run on a PC with Intel (R) Core (TM) i5-4460 3.20GHz, 8.00GB of SDRAM memory, and Windows 7 operating system.

\[\text{3.1. Normal Nonlinear Equations. In this subsection, we perform some numerical experiments to show the effectiveness of the ATTPRP algorithm. Some test problems and their relevant initial points are listed as follows:}\]
\[h(x) = (f_1(x), f_2(x), \ldots, f_n(x))^T. \] (30)

\[\text{Function 1. Exponential Function 1:}\]
\[f_1(x) = e^{x_{i-1}} - 1, \quad f_2(x) = i (e^{x_{i-1}} - x_i), \quad i = 2, 3, \ldots, n. \] (31)

Initial guess: \(x_0 = (n/n - 1, n/n - 1, \ldots, n/n - 1)^T. \)
Function 2. Exponential Function 2:
\[ f_1(x) = e^{x_1} - 1, \]
\[ f_i(x) = \frac{i(e^{x_i} + x_{i-1} - 1)}{10}, \quad i = 2, 3, \ldots, n. \]
Initial guess: \( x_0 = (1/n^2, 1/n^2, \ldots, 1/n^2) \).

Function 3. Singular function:
\[ f_1(x) = \frac{x_1^3 + x_2^2}{2}, \]
\[ f_i(x) = -\frac{x_i^2}{2} + \frac{i}{3}x_i^3 + \frac{x_{i-1}^2}{2}, \quad i = 2, 3, \ldots, n - 1, \]
\[ f_n(x) = -\frac{x_n^2}{2} + \frac{n}{3}x_n^3. \]
Initial guess: \( x_0 = (1, 1, \ldots, 1) \).

Function 4. Logarithmic function:
\[ f_i(x) = \ln(x_i + 1) - \frac{x_i}{n}, \quad i = 1, 2, \ldots, n. \]
Initial guess: \( x_0 = (1, 1, \ldots, 1) \).

Function 5. Broyden tridiagonal function:
\[ f_1(x) = (3 - 0.5x_1)x_1 - 2x_2 + 1, \]
\[ f_i(x) = (3 - 0.5x_i)x_i - x_{i-1} - 2x_{i+1} + 1, \quad i = 2, 3, \ldots, n - 1, \]
\[ f_n(x) = (3 - 0.5x_n)x_n - x_{n-1} + 1. \]
Initial guess: \( x_0 = (-1, -1, \ldots, -1) \).

Function 6. Trigexp function:
\[ f_1(x) = 3x_1^3 + 2x_2 - 5 + \sin(x_1 - x_2)\sin(x_1 + x_2), \]
\[ f_i(x) = -x_{i-1}e^{(x_i-x_1)} + x_i(4 + 3x_i^2) + 2x_{i+1} + \sin(x_i - x_{i+1})\sin(x_i + x_{i+1}) - 8, \quad i = 2, 3, \ldots, n - 1, \]
\[ f_n(x) = -x_{n-1}e^{(x_n-x_1)} + 4x_n - 3. \]
Initial guess: \( x_0 = (0, 0, \ldots, 0) \).

Function 7. Strictly convex Function 1:
\[ h(x) = \sum f_i(\sum_{i=1}^{n} e^{x_i} - x_i). \]
\[ f_i(x) = e^{x_i} - 1, \quad i = 1, 2, \ldots, n. \]
Initial guess: \( x_0 = (1/n, 2/n, \ldots, 1) \).

Function 8. Variable dimensioned function:
\[ f_i(x) = x_i - 1, \quad i = 1, 2, \ldots, n - 1, \]
\[ f_{n-1}(x) = \sum_{j=1}^{n-2} f_j(x_j - 1), \]
\[ f_n(x) = \left( \sum_{j=1}^{n-2} f_j(x_j - 1) \right)^2. \]
Initial guess: \( x_0 = (1 - 1/n, 1 - 2/n, \ldots, 0) \).

Function 9. Tridiagonal system:
\[ f_1(x) = 4(x_1 - x_2^2), \]
\[ f_i(x) = 8x_i(x_i^2 - x_{i-1}) - 2(1 - x_i), \quad i = 2, \ldots, n - 1, \]
\[ f_n(x) = 8x_n(x_n^2 - x_{n-1}) - 2(1 - x_n). \]
Initial guess: \( x_0 = (12, 12, \ldots, 1) \).

Function 10. Five-diagonal system:
\[ f_1(x) = 4(x_1 - x_2^2) + x_2 - x_3, \]
\[ f_2(x) = 8x_2(x_2^2 - x_1) - 2(1 - x_2) + 4(x_2 - x_3^2) + x_3 - x_4^2, \]
\[ f_i(x) = 8x_i(x_i^2 - x_{i-1}) - 2(1 - x_i) + 4(x_i - x_{i+1}^2) + x_{i+1}^2 - x_{i-2}, \quad i = 3, \ldots, n - 2, \]
\[ f_{n-1}(x) = 8x_{n-1}(x_{n-1}^2 - x_{n-2}) - 2(1 - x_{n-1}), \]
\[ f_n(x) = 8x_n(x_n^2 - x_{n-1}) - 2(1 - x_n) + x_{n-1}^2 - x_{n-2}. \]
Initial guess: \( x_0 = (-2, -2, \ldots, -2) \).

Function 11. Extended Freudenstein and Roth function (\( n \) is even):
For \( i = 1, 2, \ldots, n/2 \):
\[ f_{2i-1}(x) = x_{2i-1} + ((5 - x_{2i})x_{2i} - 2)x_{2i} - 13, \]
\[ f_{2i}(x) = x_{2i-1} + ((x_{2i} + 1)x_{2i} - 14)x_{2i} - 29. \]
Initial guess: \( x_0 = (6, 3, 6, 3, \ldots, 6, 3) \).

Function 12. Brent problem:
\[ f_1(x) = 3x_1(x_2 - 2x_1) + x_2^2/4, \]
\[ f_i(x) = 3x_i(x_{i+1} - 2x_i + x_{i+1}) + (x_{i+1} - x_{i-1})^2/4, \quad i = 2, \ldots, n - 1, \]
\[ f_n(x) = 3x_n(20 - 2x_n + x_{n-1}) + (20 - x_{n-1})^2/4. \]
Initial guess: \( x_0 = (0, 0, \ldots, 0, 20, 20) \).

To test the numerical performances of the ATTPRP algorithm, we also perform the experiments with the LS algorithm and the TTTPRP algorithm. The columns of Tables 1–3 have the following meanings:
| No. | Dim | NI/NF | ATTPRP algorithm | CPU  | GN  |
|-----|-----|-------|------------------|------|-----|
| 1   | 3000| 123/124 | 0.608404 | 9.97E-06 |
| 2   | 3000| 28/314  | 0.686404 | 9.50E-06 |
| 3   | 3000| 10/276  | 0.9636825 | 1.00E-06 |
| 4   | 3000| 63/558  | 1.029607 | 5.24E-07 |
| 5   | 9000| 167/1694 | 29.250188 | 8.82E-06 |
| 6   | 9000| 91/1223  | 1.918812 | 8.09E-06 |
| 7   | 9000| 70/615  | 8.800005 | 8.52E-06 |
| 8   | 3000| 1/2     | 0.000000 | 0.00E+00 |
| 9   | 9000| 0.0468  | 0.00E+00 |
| 10  | 3000| 1/2     | 0.000000 | 0.00E+00 |
| 11  | 3000| 1/2     | 0.000000 | 0.00E+00 |
| 12  | 9000| 0.0156  | 0.00E+00 |

**Table 1: Test results of the ATTPRP algorithm.**

| No. | Dim | NI/NF | CPU  | GN  |
|-----|-----|-------|------|-----|
| 1   | 3000| 129/130 | 0.436803 | 9.97E-06 |
| 2   | 9000| 89/90  | 0.873606 | 9.96E-06 |
| 3   | 3000| 59/60  | 1.232408 | 9.96E-06 |
| 4   | 9000| 41/42  | 1.435209 | 9.73E-06 |
| 5   | 3000| 46/1055 | 1.435209 | 9.99E-06 |
| 6   | 9000| 10/267 | 0.951606 | 6.76E-06 |
| 7   | 3000| 9/278  | 2.667617 | 9.83E-06 |
| 8   | 9000| 8/279  | 4.004026 | 9.00E-06 |
| 9   | 3000| 17401/18908 | 109.481502 | 9.98E-06 |
| 10  | 9000| 19999/34409 | 1542.678289 | 2.56E-05 |
| 11  | 3000| 70/662  | 1.123207 | 3.49E-06 |
| 12  | 9000| 113/136 | 6.099639 | 2.14E-06 |
| 13  | 3000| 196/2809 | 28.765584 | 6.47E-07 |
| 14  | 9000| 334/5473 | 101.884253 | 3.11E-06 |
| 15  | 3000| 54/464  | 0.514803 | 3.38E-06 |
| 16  | 9000| 72/711  | 2.152841 | 3.75E-06 |
| 17  | 3000| 100/1225 | 10.99807 | 7.38E-06 |
| 18  | 9000| 160/2256 | 31.403001 | 5.61E-06 |
| 19  | 3000| 84/1417 | 2.246415 | 6.94E-06 |
| 20  | 9000| 127/2473 | 10.311666 | 9.41E-06 |
| 21  | 3000| 211/4669 | 45.349491 | 8.60E-06 |
| 22  | 9000| 346/8524 | 141.773709 | 8.53E-06 |
| 23  | 3000| 45/366  | 0.546003 | 2.67E-06 |
| 24  | 9000| 68/729  | 2.808018 | 1.25E-06 |
| 25  | 3000| 117/1558 | 14.617294 | 8.34E-07 |
| 26  | 9000| 195/3039 | 48.219909 | 7.08E-07 |
| 27  | 3000| 1/2     | 0.0624  | 0.00E+00 |
| 28  | 9000| 0.00001 | 0.00E+00 |
| 29  | 3000| 1/2     | 0.0624  | 0.00E+00 |
| 30  | 9000| 0.00001 | 0.00E+00 |
| 31  | 3000| 6521/123863 | 139.277693 | 9.81E-06 |
| 32  | 9000| 6827/141140 | 425.679929 | 9.97E-06 |
| 33  | 3000| 771/182289 | 1559.916399 | 9.99E-06 |
| 34  | 9000| 9107/251132 | 3460.398582 | 9.85E-06 |
| 35  | 3000| 4904/65319 | 77.750898 | 1.00E-05 |
| 36  | 9000| 5271/71854 | 229.820673 | 9.73E-06 |
| 37  | 3000| 5280/75948 | 689.150018 | 9.97E-06 |
| 38  | 9000| 5665/88105 | 1269.302137 | 9.99E-06 |
| 39  | 3000| 300/4529  | 4.898453 | 9.88E-06 |
| 40  | 9000| 387/6163 | 17.846514 | 9.61E-06 |
| 41  | 3000| 472/8397  | 72.836867 | 9.83E-06 |
| 42  | 9000| 620/12490 | 168.652681 | 9.27E-06 |
| 43  | 3000| 193/198  | 0.124801 | 9.99E-06 |
| 44  | 9000| 193/198  | 0.826805 | 9.99E-06 |
| 45  | 3000| 193/198  | 2.586917 | 9.99E-06 |

NO: the serial number of the problem  
Dim: the variable x dimensions  
NI: the number of iterations  
NF: the number of iterations of the function value  
CPU: the calculation time in seconds  
GN: the final function norm evaluations when the program is stopped

Initialization: the parameters are chosen as \(\rho = 1\),  
\(s = 0.69\), \(\lambda = 0.925\), \(k = 0.0001\), and \(\epsilon = 10^{-5}\)

Stop rule: when the condition \(\|h(x)\| \leq 10^{-5}\) or NI \(\geq 20000\) is satisfied, we stop the process

From Tables 1–3, it is obvious that the three methods can successfully solve most of the test problems with NI<20000. However, for Function 3 with 9000 and 90000 variables, the
TTPRP and LS algorithms cannot handle the function, but the proposed algorithm can do with $NI < 20000$. To more directly show the methods’ performance, Dolan and Moré [36] proposed a drawing tool that can obtain the performance profiles of methods. Therefore, using the drawing tool, we obtain Figures 1–3, which are related to the NI, NF, and CPU in Tables 1–3. In Figure 1, the ATTPRP algorithm solves all test problems at approximately $\tau = 1.18$, while the LS algorithm solves $80\%$ of the test problems at approximately $\tau = 1.388$, and the TTPRP algorithm solves $92\%$ at approximately $\tau = 1.9$. Thus, we can obtain the result that the ATTPRP algorithm performs slightly better than the other two algorithms. When $\tau = 1.45$ in Figure 2, the presented algorithm solves all test problems, the TTPRP algorithm solves $87\%$ of all test problems, and the LS algorithm only
Figure 3: Performance profiles of the methods (CPU).

Figure 4: Restoration of the Cameraman, Barbara, and Man images by using the ATTPRP algorithm and TTPRP algorithm. From left to right: a noisy image with 30% salt-and-pepper noise and the restorations obtained with the ATTPRP algorithm and the TTPRP algorithm by minimizing $z$. 
solves approximately 70% of the test problems. Thus, it is not difficult to see that the ATTPRP algorithm is more competitive than the other two methods. In Figure 3, the curve of the ATTPRP algorithm is above those of the TTPRP and LS algorithms, which indicates that the proposed algorithm is more robust than the other two algorithms in terms of the CPU. In summary, the enhancement of the presented method is noticeable.

3.2. Image Restoration Problems. The purpose of this subsection is to recover the original image from an image damaged by impulse noise. It has important practical significance in optimization fields. The selection of parameters is similar to that in the above subsection. The stop condition is $\|h_{k+1} - h_k\|/\|h_k\| < 10^{-3}$ or $\|x_{k+1} - x_k\|/\|x_k\| < 10^{-3}$. For the experiments, Cameraman (256 × 256), Barbara (512 × 512), and Man (1024 × 1024) are chosen as the test images. We also perform experiments to compare the ATTPRP algorithm with the TTPRP algorithm, where the step length $\alpha_k$ is generated by Step 2 and Step 3 in the ATTPRP algorithm. More detailed performance results are shown in Figures 4–6. It is not difficult to see that both the ATTPRP and TTPRP algorithms are successful in the image restoration of the three images. The expenditure of the CPU time is listed in Table 4 to compare the ATTPRP algorithm with the TTPRP algorithm.

From Figures 1–4, we can obviously note that both algorithms can perfectly restore a noisy image with 30%, 50%, and 70% salt-and-pepper noise. In addition, the results in Table 4 show that the ATTPRP algorithm and the TTPRP algorithm are both successful in restoring these images with an approximate CPU time. The presented algorithm is slightly competitive with the TTPRP algorithm for 30% noise problems, 50% noise problems, and 70% noise problems.
4. Conclusions

In this paper, an accelerated conjugate gradient algorithm that combines the TTPRP method, the acceleration step length, and the hyperplane projection technique is proposed. All search directions $d_k$ generated by using the algorithm automatically have sufficient descent and trust region properties. The global convergence property of the proposed algorithm is established under suitable conditions. The numerical results show that the proposed algorithm is effective. The image restoration problems also demonstrate that the proposed algorithm is successful.

For future research, we have some ideas as follows: (i) If the acceleration system is introduced into the quasi-Newton method, does it have some good properties? (ii) Can the acceleration system be introduced into the trust region method to solve unconstrained optimization problems and nonlinear equations? (iii) Can the proposed algorithm be applied to machine learning?

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.
Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 11661009), the High Level Innovation Teams and Excellent Scholars Program in Guangxi Institutions of Higher Education (Grant No. (2019) 52), and the Guangxi Natural Science Key Fund (No. 2017GXNSFDA198046).

References

[1] S. P. Dirkse and M. C. Ferris, "MCPLIB: a collection of nonlinear mixed complementarity problems," Optimization Methods and Software, vol. 5, no. 4, pp. 319–345, 1995.

[2] A. Griewank. "The "global" convergence of Broyden-like methods with suitable line search," The Journal of the Australian Mathematical Society. Series B. Applied Mathematics, vol. 28, no. 1, pp. 75–92, 1986.

[3] L. Grippo and M. Sciandrone, "Nonmonotone derivative-free methods for nonlinear equations," Computational Optimization and Applications, vol. 37, no. 3, pp. 297–328, 2007.

[4] D. Li and M. Fukushima, "A derivative-free line search and DFP method for symmetric equations with global and superlinear convergence," Numerical Functional Analysis and Optimization, vol. 20, no. 1-2, pp. 59–77, 1999.

[5] 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.

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

[7] S. Kiefer, M. Luttenberger, and J. Esparza, "On the convergence of Newton’s method for monotone systems of polynomial equations," in Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, CA, USA, June 2007.

[8] G. N. Silva, "Local convergence of Newton’s method for solving generalized equations with monotone operator," Applicable Analysis, vol. 97, no. 7, pp. 1094–1105, 2018.

[9] G. Yuan, Z. Wei, and S. Lu, "Limited memory BFGS method with backtracking for symmetric nonlinear equations," Mathematical and Computer Modelling, vol. 54, no. 1-2, pp. 367–377, 2011.

[10] B. Zhang and Z. Zhu, "A modified quasi-Newton diagonal update algorithm for total variation denoising problems and nonlinear monotone equations with applications in compressive sensing," Numerical Linear Algebra with Applications, vol. 22, no. 3, pp. 500–522, 2015.

[11] W.-J. Zhou and D.-H. Li, "A globally convergent BFGS method for nonlinear monotone equations without any merit functions," Mathematics of Computation, vol. 77, no. 264, pp. 2231–2240, 2008.

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

[13] G. Zhou and K. C. Toh, "Superlinear convergence of a Newton-type algorithm for monotone equations," Journal of Optimization Theory and Applications, vol. 125, no. 1, pp. 205–221, 2005.

[14] Y. Qiu, C. Ying, and L. Lei, "Multivariate spectral conjugate gradient projection method for nonlinear monotone equations," in Proceedings of the 2013 Fourth International Conference on Emerging Intelligent Data and Web Technologies (EIDWT), Xi’an, China, September 2013.

[15] 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.

[16] Y. Hu and Z. Wei, "Wei-yao-liu conjugate gradient projection algorithm for nonlinear monotone equations with convex constraints," International Journal of Computer Mathematics, vol. 92, no. 11, pp. 1–12, 2015.

[17] X. Y. Wang, S. J. Li, and X. P. Kou, "A self-adaptive three-term conjugate gradient method for monotone nonlinear equations with convex constraints," Calcolo, vol. 53, no. 2, pp. 133–145, 2016.

[18] S.-Y. Liu, Y.-Y. Huang, and H.-W. Jiao, "Sufficient descent conjugate gradient methods for solving convex constrained nonlinear monotone equations," Abstract and Applied Analysis, vol. 2014, no. 1, pp. 1–12, 2014.

[19] 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, p. 113, 2018.

[20] M. R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," Journal of Research of the National Bureau of Standards, vol. 49, no. 6, pp. 409–436, 1952.

[21] R. Fletcher and C. Reeves, "Function minimization by conjugate gradients," The Computer Journal, vol. 7, no. 2, pp. 149–154, 1964.

[22] E. Polak and G. Ribière, "Note sur la convergence de méthodes de directions conjuguées," Revue française d’informatique et de recherche opérationnelle. Série rouge, vol. 3, no. 16, pp. 35–43, 1969.

[23] B. T. Polyak, "The conjugate gradient method in extremal problems," USSR Computational Mathematics and Mathematical Physics, vol. 9, no. 4, pp. 94–112, 1969.

[24] Y. Liu and C. Storey, "Efficient generalized conjugate gradient algorithms, part 1: theory," Journal of Optimization Theory and Applications, vol. 69, no. 1, pp. 129–137, 1991.

[25] R. Fletcher, Practical Method of Optimization Vol. I: Unconstrained Optimization, John Wiley and Sons, New York, NY, USA, 1987.

[26] Y. H. Dai and Y. Yuan, "A nonlinear conjugate gradient method with a strong global convergence property," SIAM Journal on Optimization, vol. 10, no. 1, pp. 177–182, 1999.

[27] L. Zhang, W. Zhou, and D.-H. Li, "A descent modified polak-ribière-polyak conjugate gradient method and its global convergence," IMA Journal of Numerical Analysis, vol. 26, no. 4, pp. 629–640, 2006.

[28] 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.

[29] 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.

[30] M. Koorapetse and P. Kaelo, "Globally convergent three-term conjugate gradient methods with suitable line search," Computational Optimization and Applications, vol. 5, no. 4, pp. 319–345, 1995.

[31] 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.
[32] J. Liu, S. Li, and S. Li, “Multivariate spectral DY-type projection method for convex constrained nonlinear monotone equations,” Journal of Industrial & Management Optimization, vol. 13, no. 1, pp. 283–295, 2017.

[33] A. A. Goldstein, “Convex programming in hilbert space,” Bulletin of the American Mathematical Society, vol. 70, no. 5, pp. 709–711, 1964.

[34] M. Solodov and B. Svaiter, “A globally convergent inexact Newton method for systems of monotone equations,” in Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods, pp. 355–369, Kluwer Academic Publishers, New York, NY, USA, 1998.

[35] N. Andrei, “Another conjugate gradient algorithm with guaranteed descent and conjugacy conditions for large-scale unconstrained optimization,” Journal of Optimization Theory and Applications, vol. 159, no. 1, pp. 159–182, 2013.

[36] E. D. Dolan and J. J. Moré, “Benchmarking optimization software with performance profiles,” Mathematical Programming, vol. 91, no. 2, pp. 201–213, 2002.