Mathematics

https://www.aimspress.com/journal/Math

AIMS Mathematics, 10(5): 11656–11675.
DOI: 10.3934/math.2025528
Received: 16 March 2025
Revised: 06 May 2025
Accepted: 12 May 2025
Published: 21 May 2025

Research article

A derivative-free RMIL conjugate gradient method for constrained nonlinear systems of monotone equations

Xiaowei Fang*

Department of Mathematics, Huzhou University, Huzhou Zhejiang, 313000, China

* **Correspondence:** Email: fangxiaowei@163.com.

Abstract: In this paper, we introduce a derivative-free RMIL conjugate gradient method designed for nonlinear systems of monotone equations with convex constraints. The proposed method represents a refinement of the RMIL method through its integration with projection techniques and a derivative-free line search strategy. When certain reasonable assumptions are satisfied, we can prove the global convergence of the proposed method. Numerical experiments demonstrate that the method is highly effective. Moreover, we employ the method to solve sparse signal restoration problems.

Keywords: nonlinear monotone equations; conjugate gradient method; hyperplane projection technique; signal recovery
Mathematics Subject Classification: 90C56, 90C30

1. Introduction

This paper aims at finding solutions to the following constrained nonlinear monotone equations:

$$F(x) = 0, \quad x \in \Omega, \tag{1.1}$$

in which $F : \mathbb{R}^n \to \mathbb{R}^n$ is monotone and continuously differentiable, and $\Omega \subseteq \mathbb{R}^n$ is a nonempty closed convex set. The property of monotonicity for F is defined as

$$(F(x) - F(y))^T (x - y) \geq 0, \quad \forall x, y \in \mathbb{R}^n. \tag{1.2}$$

Nonlinear monotone equations emerge from different problems and are widely applied in interdisciplinary domains. For example, the automatic control systems [1], the compressive sensing [2–4], and the optimal power flow [5].

There are numerous iterative algorithms for solving (1.1). For instance, the Newton algorithm [6], the quasi-Newton algorithm [7], the Gauss-Newton algorithm [8], and the BFGS algorithm [9]. These algorithms exhibit good convergence properties from suitable initial guesses. Nevertheless, they are unsuitable for handling large-scale nonlinear equations since the computation of the Jacobian matrix or its approximation is required in every iteration.

Recently, some optimization methods that build on the projection and proximal point method presented by Solodov and Svaiter [10] have been extended to solve (1.1). These include conjugate gradient methods originally applied to solve large-scale unconstrained optimization problems due to the low memory and simple structures. For example, Li and Li [11] described a category of derivative-free approaches for systems of monotone equations. Cheng [12] formulated a PRP-type method using the line search strategy, and Awwal *et al*. [13] proposed a DFP-type derivative-free method. Ahookhosh *et al*. [14] developed two new derivative-free approaches for systems of nonlinear monotone equations. According to the RMIL conjugate gradient method presented by Rivaie *et al* [15], Fang [16, 17] put forward a set of adjusted derivative-free gradient-type approaches for solving nonlinear equations. Depending on the hybrid idea and a new adaptive line search strategy, Liu *et al.* in [18] proposed a three-term algorithm to solve the nonlinear monotone equations under convex constraints. According to the spectral secant equation, Zhang *et al.* [19] developed a three-term projection method for the purpose of dealing with nonlinear monotone equations. Moreover, they utilized this method to solve the problem in signal processing. Abdullahi *et al.* [20,21] described a modified self-adaptive algorithm and a three-term projection algorithm for solving systems of monotone nonlinear equations. Furthermore, they employ these algorithms in signal and image recovery problems. Aji *et al.* [22] proposed a novel spectral algorithm using an inertial technique to solve a system of monotone equations and for its applications.

Motivated by the aforementioned works, we give a new conjugate gradient projection algorithm for nonlinear monotone equation (1). The structure of this paper is as follows. Section 2 is dedicated to proposing the new method. In Section 3, we conduct the convergence analysis of the newly proposed method. Section 4 showcases the numerical results obtained from solving monotone nonlinear equations. Subsequently, in Section 5, we apply the proposed method to address the signal recovery problem. Finally, Section 6 presents the conclusions. The symbol $\|\cdot\|$ stands for Euclidean norm, and $\|\cdot\|_1$ stands for l_1 norm.

2. Algorithm

In this part, we initially review the conjugate gradient method that is employed to solve the following unconstrained optimization problems:

$$\min f(x), \tag{2.1}$$

in which f is a smooth function mapping from $\mathbb{R}^n$ to $\mathbb{R}$, and the gradient of $f(x_k)$ is symbolized as g_k. The conjugate gradient method has the following iterative formula:

$$x_{k+1} = x_k + \alpha_k d_k, \tag{2.2}$$

in which the step length α_k is given through some line search techniques, and the search direction d_k is derived from

$$d_k = \begin{cases} -g_k & \text{if } k = 0 \\ -g_k + \beta_k d_{k-1} & \text{if } k \geq 1 \end{cases}. \tag{2.3}$$

In recent times, Rivaie *et al*. [15] proposed the RMIL conjugate gradient method (RMIL, named after its developers Rivaie, Mustafa, Ismail, and Leong), where the β_k is computed by

$$\beta_k = \frac{g_k^T(g_k - g_{k-1})}{\|d_{k-1}\|^2}. \tag{2.4}$$

Results from numerical experiments reveal that the method is highly efficient.

Motivated by the results of the RMIL method, we develop a derivative-free approach aimed at solving nonlinear systems of monotone equations. For the sake of simplicity, we substitute g_k in equations (2.3) and (2.4) with F_k, where $F_k = F(x_k)$. Moreover, we change the coefficient of g_k from -1 to $-\theta_k$ when $k \geq 1$. In this way, it can ensure that the search direction generated by the algorithm satisfies the property of sufficient descent, and it can effectively improve the computational efficiency of the algorithm. Consequently, the search direction d_k is given by

$$d_k = \begin{cases} -F_k & \text{if } k = 0 \\ -\theta_k F_k + \beta_k d_{k-1} & \text{if } k \geq 1 \end{cases}, \tag{2.5}$$

where $\theta_k = \beta_k \frac{F_k^T d_{k-1}}{\|F_k\|^2} + 1, \beta_k = \frac{F_k^T y_{k-1}}{\|d_{k-1}\|^2}$, $F_k = F(x_k)$, $y_{k-1} = F_k - F_{k-1}$. From the definitions of d_k, it is not difficult to see that for $k \in \mathbb{N}$, we have

$$\begin{aligned} F_k^T d_k &= -F_k^T(\beta_k \frac{F_k^T d_{k-1}}{\|F_k\|^2} + 1)F_k + F_k^T \beta_k d_{k-1} \\ &= -\beta_k \frac{F_k^T d_{k-1}}{\|F_k\|^2}\|F_k\|^2 - \|F_k\|^2 + \beta_k F_k^T d_{k-1} \\ &= -\|F_k\|^2. \end{aligned} \tag{2.6}$$

When $k = 0$, we get $F_0^T d_0 = -\|F_0\|^2$.

Next we state the projection operator $P_\Omega[.]$, which is described as follows:

$$P_\Omega[x] = \arg\min\{\|x - y\| \mid y \in \Omega\}, \quad \forall x \in \mathbb{R}^n. \tag{2.7}$$

This operation refers to projecting the vector x onto the closed convex set Ω. By doing so, it is guaranteed that the subsequent iterative point produced by our algorithm remains within the domain Ω. The projection operator is known to possess the non-expansive property, namely

$$\|P_\Omega[x] - P_\Omega[y]\| \leq \|x - y\|, \quad \forall x, y \in \mathbb{R}^n. \tag{2.8}$$

From here on, our attention is directed towards the method for solving the monotone equations (1.1). We adopt the projection procedure presented in [10] and obtain a point

$$z_k = x_k + \alpha_k d_k, \tag{2.9}$$

with the result that

$$F(z_k)^T(x_k - z_k) > 0, \tag{2.10}$$

where α_k is obtained using a suitable line search technique.

Meanwhile, for any x^* satisfying $F(x^*) = 0$, owing to the monotonic property of F, we have

$$F(z_k)^T(x^* - z_k) = -(F(x^*) - F(z_k))^T(x^* - z_k) \leq 0. \tag{2.11}$$

Incorporating the expression (2.10), we have the conclusion that the hyperplane

$$H_k = \{x \in \mathbb{R}^n | F(z_k)^T(x - z_k) = 0\} \tag{2.12}$$

strictly divides the solutions of the equations (1.1) and x_k. Consequently, Solodov and Svaiter [10] calculate the next iterate x_{k+1} through the projection of x_k onto the hyperplane H_k. The x_{k+1} is defined as

$$x_{k+1} = x_k - \frac{F(z_k)^T(x_k - z_k)}{\|F(z_k)\|^2}F(z_k). \tag{2.13}$$

Now, we give the steps of our algorithm.

Algorithm 2.1. :
Step 0 : *Choose* $\sigma > 0, s > 0, \epsilon > 0, 1 > \rho > 0, 2 > \gamma > 0$. *and the starting point* $x_0 \in \mathbb{R}^n$. *Set* $k = 0$.
Step 1 : *If* $\|F(x_k)\| < \epsilon$*, stop. Otherwise, move to step 2.*
Step 2 : *Define the search direction* d_k *through* (2.5).
Step 3 : *Determine the steplength* $\alpha_k = \max\{s\rho^i | i = 0, 1, 2, \cdots\}$ *satisfies*

$$-F(x_k + \alpha_k d_k)^T d_k \geq \sigma\alpha_k\|F(x_k + \alpha_k d_k)\|\|d_k\|^2, \tag{2.14}$$

then set $z_k = x_k + \alpha_k d_k$ *and move to step 4.*
Step 4 : *If* $z_k \in \Omega$ *and* $\|F(z_k)\| < \epsilon$*, set* $x_{k+1} = z_k$*, stop. Else compute*

$$x_{k+1} = P_\Omega\Big[x_k - \gamma\frac{F(z_k)^T(x_k - z_k)}{\|F(z_k)\|^2}F(z_k)\Big] \tag{2.15}$$

and set $k := k + 1$ *, move to step 1.*

3. Convergence analysis

In this section, for the purpose of attaining the global convergence of Algorithm 2.1, the following assumptions are necessary.

Assumption 3.1. *(1) The set of solutions for problem* (1.1) *is nonempty.*
(2) The mapping $F(\cdot)$ *is Lipschitz continuous on* $\mathbb{R}^n$*, that is*

$$\|F(x) - F(y)\| \leq L\|x - y\|, \quad \forall x, y \in \mathbb{R}^n, \tag{3.1}$$

in which L represents a positive constant.

From Assumption 3.1, it can be inferred that

$$\|F(x)\| \leq \kappa \quad \forall x \in \mathbb{R}^n, \tag{3.2}$$

in which κ is defined as a positive constant.

Lemma 3.1. *Assume that Assumption 3.1 holds and the sequences $\{x_k\}, \{\alpha_k\}, \{d_k\}$ are yielded in accordance with Algorithm 2.1. Then, for all $x^* \in \Omega$ that satisfies $F(x^*) = 0$, and given that $\gamma \in (0, 2)$, we obtain*

$$\|x_{k+1} - x^*\|^2 - \|x_k - x^*\|^2 \le -\gamma(2-\gamma)\sigma^2 \|\alpha_k d_k\|^4. \tag{3.3}$$

Furthermore, it holds that

$$\lim_{k\to+\infty} \|\alpha_k d_k\| = 0. \tag{3.4}$$

Proof. Suppose $F(x^*) = 0$ and $x^* \in \Omega$; due to the monotonicity of F, we obtain

$$F(z_k)^T(z_k - x^*) \ge 0. \tag{3.5}$$

Combining with (3.5), we get

$$F(z_k)^T(x_k - x^*) = F(z_k)^T(x_k - z_k) + F(z_k)^T(z_k - x^*) \ge F(z_k)^T(x_k - z_k). \tag{3.6}$$

Thus, from (2.8), (2.10), (2.14), (2.15), and (3.6), we have

$$\begin{aligned}
\|x_{k+1} - x^*\|^2 - \|x_k - x^*\|^2 &= \|P_\Omega[x_k - \gamma \frac{F(z_k)^T(x_k - z_k)}{\|F(z_k)\|^2} F(z_k)] - x^*\|^2 - \|x_k - x^*\|^2 \\
&\le \|x_k - \gamma \frac{F(z_k)^T(x_k - z_k)}{\|F(z_k)\|^2} F(z_k) - x^*\|^2 - \|x_k - x^*\|^2 \\
&= -2\gamma \frac{F(z_k)^T(x_k - z_k)}{\|F(z_k)\|^2} F(z_k)^T(x_k - x^*) + \gamma^2 \frac{(F(z_k)^T(x_k - z_k))^2}{\|F(z_k)\|^2} \\
&\le -2\gamma \frac{F(z_k)^T(x_k - z_k)}{\|F(z_k)\|^2} F(z_k)^T(x_k - z_k) + \gamma^2 \frac{(F(z_k)^T(x_k - z_k))^2}{\|F(z_k)\|^2} \\
&= -\frac{\gamma(2-\gamma)(F(z_k)^T(x_k - z_k))^2}{\|F(z_k)\|^2} \\
&\le -\gamma(2-\gamma)\sigma^2 \|\alpha_k d_k\|^4.
\end{aligned} \tag{3.7}$$

Then, we can deduce that

$$\gamma(2-\gamma)\sigma^2 \sum_{i=0}^{k} \|\alpha_i d_i\|^4 \le \|x_0 - x^*\|^2 - \|x_{k+1} - x^*\|^2 \le \|x_0 - x^*\|^2, \tag{3.8}$$

which together with $\gamma \in (0, 2)$, means that

$$\lim_{k\to+\infty} \|\alpha_k d_k\| = 0. \tag{3.9}$$

□

Lemma 3.2. *Assume that Assumption 3.1 holds and sequences $\{x_k\}, \{d_k\}$ are yielded in accordance with Algorithm 2.1.Then we have*

$$\|d_k\| \le \kappa(1 + 2\gamma L s), \tag{3.10}$$

in which $\kappa > 0, s > 0, L > 0, 2 > \gamma > 0$.

Proof. From (2.8) and step 4 of Algorithm 2.1, we obtain

$$\begin{aligned}
\|x_k - x_{k-1}\| &= \|P_\Omega\Big[x_{k-1} - \gamma\frac{F(z_{k-1})^T(x_{k-1} - z_{k-1})}{\|F(z_{k-1})\|^2}F(z_{k-1})\Big] - x_{k-1}\| \\
&\leq \|x_{k-1} - \gamma\frac{F(z_{k-1})^T(x_{k-1} - z_{k-1})}{\|F(z_{k-1})\|^2}F(z_{k-1}) - x_{k-1}\| \\
&= \|\gamma\frac{F(z_{k-1})^T(x_{k-1} - z_{k-1})}{\|F(z_{k-1})\|^2}F(z_{k-1})\| \\
&\leq \gamma\|x_{k-1} - z_{k-1}\|
\end{aligned} \tag{3.11}$$

For $k \in \mathbb{N}$, using (2.5), (3.1), (3.2), (3.11), and step 3 of Algorithm 2.1, we have

$$\begin{aligned}
\|d_k\| &= \| - (\beta_k\frac{F_k^T d_{k-1}}{\|F_k\|^2} + 1)F_k + \beta_k d_{k-1}\| \\
&\leq \|F_k\| + 2\|\beta_k d_{k-1}\| \\
&\leq \|F_k\| + 2\frac{\|F_k\|\|d_{k-1}\|\|y_{k-1}\|}{\|d_{k-1}\|^2} \\
&= \|F_k\| + 2\frac{\|F_k\|\|F_k - F_{k-1}\|}{\|d_{k-1}\|} \\
&\leq \|F_k\| + 2L\frac{\|F_k\|\|x_k - x_{k-1}\|}{\|d_{k-1}\|} \\
&\leq \|F_k\| + 2\gamma L\frac{\|F_k\|\|x_{k-1} - z_{k-1}\|}{\|d_{k-1}\|} \\
&= \|F_k\| + 2\gamma L\alpha_{k-1}\frac{\|F_k\|\|d_{k-1}\|}{\|d_{k-1}\|} \\
&= \|F_k\|(1 + 2\gamma L\alpha_{k-1}) \\
&\leq \kappa(1 + 2\gamma Ls).
\end{aligned} \tag{3.12}$$

From (2.5) and (3.2), we get

$$\|d_0\| = \| - F_0\| \leq \kappa. \tag{3.13}$$

Combining with (3.12), we yield (3.10). □

Lemma 3.3. *Assume that Assumption 3.1 holds and the sequences $\{x_k\}$, $\{z_k\}$, and $\{\alpha_k\}$ are yielded in accordance with Algorithm 2.1; then we get*

$$\alpha_k \geq \min\left\{s, \frac{\|F_k\|^2}{\rho^{-1}\kappa^2\Big(1 + 2\gamma Ls\Big)^2\Big(L + \sigma\kappa + \sigma\kappa\rho^{-1}Ls(1 + 2\gamma Ls)\Big)}\right\}. \tag{3.14}$$

Proof. In the case where $\alpha_k \neq s$, relying on the line search technique of Algorithm 2.1, it follows that $\rho^{-1}\alpha_k$ fails to satisfy (2.14), namely

$$-F(x_k + \rho^{-1}\alpha_k d_k)^T d_k < \sigma\rho^{-1}\alpha_k\|F(x_k + \rho^{-1}\alpha_k d_k)\|\|d_k\|^2. \tag{3.15}$$

Combining with (3.1), (3.2) and (3.10), we obtain

$$\begin{aligned}\|F(x_k+\rho^{-1}\alpha_k d_k)\| &= \|F(x_k+\rho^{-1}\alpha_k d_k)-F(x_k)\|+\|F(x_k)\|\\ &\le L\rho^{-1}\alpha_k\|d_k\|+\kappa \\ &\le \kappa\Big(1+\rho^{-1}Ls(1+2\gamma Ls)\Big).\end{aligned} \tag{3.16}$$

It follows from (2.6), (3.1), (3.10), (3.15), and (3.16) that we have

$$\begin{aligned}\|F_k\|^2 &= -F_k^T d_k\\ &= [F(x_k+\rho^{-1}\alpha_k d_k)-F(x_k)]^T d_k - [F(x_k+\rho^{-1}\alpha_k d_k)]^T d_k\\ &\le L\rho^{-1}\alpha_k\|d_k\|^2+\sigma\rho^{-1}\alpha_k(L\rho^{-1}\alpha_k\|d_k\|+\kappa)\|d_k\|^2\\ &= \rho^{-1}\alpha_k\|d_k\|^2\Big(L+\sigma(L\rho^{-1}\alpha_k\|d_k\|+\kappa)\Big)\\ &\le \rho^{-1}\alpha_k\kappa^2\Big(1+2\gamma Ls\Big)^2\Big(L+\sigma\kappa+\sigma\kappa\rho^{-1}Ls(1+2\gamma Ls)\Big).\end{aligned} \tag{3.17}$$

The above inequality shows that

$$\alpha_k \ge \frac{\|F_k\|^2}{\rho^{-1}\kappa^2\Big(1+2\gamma Ls\Big)^2\Big(L+\sigma\kappa+\sigma\kappa\rho^{-1}Ls(1+2\gamma Ls)\Big)}. \tag{3.18}$$

This implies (3.14). □

Theorem 3.4. *Assume that Assumption 3.1 holds and the sequences $\{x_k\}$ and $\{F_k\}$ are yielded in accordance with Algorithm 2.1. Then we obtain*

$$\liminf_{k\to\infty}\|F_k\|=0. \tag{3.19}$$

Proof. Assume that (3.19) is false. Then, there is a positive constant k for which

$$\|F_k\|\ge\xi. \tag{3.20}$$

Based on (2.6) and (3.20), we obtain

$$\|d_k\|=\frac{\|F_k\|\|d_k\|}{\|F_k\|}\ge\frac{\|F_k^T d_k\|}{\|F_k\|}\ge\|F_k\|\ge\xi. \tag{3.21}$$

Combining with Lemma 3.3, we obtain

$$\alpha_k\|d_k\|\ge\min\left\{s\xi,\ \frac{\xi^3}{\rho^{-1}\kappa^2\Big(1+2\gamma Ls\Big)^2\Big(L+\sigma\kappa+\sigma\kappa\rho^{-1}Ls(1+2\gamma Ls)\Big)}\right\}>0. \tag{3.22}$$

which contradicts the conclusion (3.4) of Lemma 3.1, then (3.19) holds. □

4. Numerical experiments

In the present section, we conduct a series of numerical experiments in order to assess the performance of Algorithm 2.1. We make a comparison between it and the three-term projection method put forward by Zhang et al. [19], the three-term conjugate method proposed by Gao et al. [23], and the PRP-like method presented by Abubakar et al. [24]. We conduct our tests using Matlab R2018a on a personal computer equipped with 32 GB of RAM and an 11th Gen Intel(R) Core(TM) i7-11700K processor.

We test the algorithms on ten problems that have been widely used in the literature for dimensions 50000 and 200000. All test problems are initialized with the following eight starting points: $x_1 = (10, 10, \cdots, 10)^T$, $x_2 = (-10, -10, \cdots, -10)^T$, $x_3 = (-1, -1, \cdots, -1)^T$, $x_4 = (0.1, 0.1, \cdots, 0.1)^T$, $x_5 = (\frac{1}{2}, \frac{1}{2^2}, \cdots, \frac{1}{2^n})^T$, $x_6 = (1, \frac{1}{2}, \cdots, \frac{1}{n})^T$, $x_7 = (\frac{1}{n}, \frac{2}{n}, \cdots, 1)^T$, $x_8 = (\frac{n-1}{n}, \frac{n-2}{n}, \cdots, 0)^T$, which are derived from [17,19,23]. With respect to every algorithm and each test problem,, the program concludes when any of the three conditions described below is achieved: (1) $\|F(x_k)\| \leq 10^{-6}$,(2) $\|F(z_k)\| \leq 10^{-6}$, (3) The algorithm still fails to converge after 1000 iterations. The parameters of methods introduced by Zhang, Gao and Abubakar are set as described in their respective articles. Regarding Algorithm 2.1, we define $\sigma = 10^{-4}, s = 1, \epsilon = 10^{-5}, \rho = 0.55, \gamma = 1.2$. At present, we are going to list out the test problems, where the mapping F has the following definition: $F(x) = (f_1(x), f_2(x), \cdots, f_n(x))^T$.

Test Problem 1 This problem sourced from [23] is

$$f_i(x) = x_i - sin(|x_i| - 1), i = 1, 2, \cdots, n,$$
$$\Omega = \{x \in \mathbb{R}^n | x_i \geq -1, \sum_{i=1}^{n} x_i \leq n, i = 1, 2, \cdots, n\}.$$

Test Problem 2 This problem sourced from [23] is

$$f_i(x) = e^{x_i} - 1, i = 1, 2, \cdots, n,$$
$$\Omega = \mathbb{R}^n_+.$$

Test Problem 3 This problem sourced from [23] is

$$f_i(x) = 2x_i - sin(x_i), i = 1, 2, \cdots, n,$$
$$\Omega = \mathbb{R}^n_+.$$

Test Problem 4 This problem sourced from [19] is

$$f_i(x) = x_i - 3x_i\Big(\frac{sin(x_i)}{3} - 0.66\Big) + 2, i = 1, 2, \cdots, n,$$
$$\Omega = \{x \in \mathbb{R}^n | x_i \geq -5, i = 1, 2, \cdots, n\}.$$

Test Problem 5 This problem sourced from [19] is

$$f_i(x) = (e^{x_i})^2 + 3sin(x_i)cos(x_i) - 1, i = 1, 2, \cdots, n,$$
$$\Omega = \{x \in \mathbb{R}^n | x_i \geq -5, i = 1, 2, \cdots, n\}.$$

Test Problem 6 This problem sourced from [19] is

$$
\begin{aligned}
f_1(x) &= 4x_1(x_1^2 + x_2^2) - 4, \\
f_i(x) &= 4x_i(x_{i-1}^2 + x_i^2) + 4x_i(x_i^2 + x_{i+1}^2) - 4, i = 2, 3, \cdots, n-1, \\
f_n(x) &= 4x_n(x_{n-1}^2 + x_n^2), \\
\Omega &= \mathbb{R}_+^n.
\end{aligned}
$$

Test Problem 7 This problem sourced from [12] is

$$
\begin{aligned}
f_1(x) &= 2.5x_1 + x_2 - 1, \\
f_i(x) &= x_{i-1} + 2.5x_i + x_{i+1}, i = 2, 3, \cdots, n-1, \\
f_n(x) &= x_{n-1} + 2.5x_n - 1), \\
\Omega &= \mathbb{R}_+^n.
\end{aligned}
$$

Test Problem 8 This problem sourced from [25] is

$$
\begin{aligned}
f_1(x) &= cos(x_1) - 9 + 3x_1 + 8exp(x_2), \\
f_i(x) &= cos(x_i) - 9 + 3x_i + 8exp(x_{i-1}), i = 2, 3, \cdots, n, \\
\Omega &= \mathbb{R}_+^n.
\end{aligned}
$$

Test Problem 9 This problem sourced from [25] is

$$
\begin{aligned}
f_1(x) &= \frac{1}{(x_1 + 1)^2} - exp(x_i), \\
f_i(x) &= \frac{1}{(x_i + 1)^2} + cos(x_{i+1}) - 2exp(x_i), i = 2, 3, \cdots, n-1, \\
f_n(x) &= \frac{1}{(x_n + 1)^2} - exp(x_n), \\
\Omega &= \mathbb{R}_+^n.
\end{aligned}
$$

Test Problem 10 This problem sourced from [25] is

$$
\begin{aligned}
f_i(x) &= 2x_i - sin|x_i|, i = 1, 2, \cdots, n, \\
\Omega &= \mathbb{R}_+^n.
\end{aligned}
$$

In Tables 1, 2, and 3, we present the detailed outcomes using the format of "Niter/ Nfun/Time". Here, "Niter" represents the number of iterations, "Nfun" denotes the number of function value calculations, and "Time" stands for the CPU time counted in seconds. "Dim" indicates the dimension of test problems, and "Sta" represents the starting points. Through a comprehensive analysis of Tables 1, 2 and 3, it can be clearly observed that, for the given problems, Algorithm 2.1 registered the fewest values for Niter and Nfun in a large number of instances.

Table 1. The results of four algorithms for Test Problems 1, 2, 3, and 4.

Problem	Dim	Sta	Zhang Niter/Nfun/Time	Gao Niter/Nfun/Time	Abubakar Niter/Nfun/Time	Algorithm 2.1 Niter/Nfun/Time
TP1	50000	x_1	18 / 53 / 0.043	12 / 34 / 0.031	10 / 39 / 0.025	18 / 53 / 0.047
		x_2	18 / 54 / 0.038	11 / 33 / 0.021	10 / 39 / 0.020	18 / 54 / 0.040
		x_3	17 / 52 / 0.035	10 / 30 / 0.024	9 / 38 / 0.022	17 / 52 / 0.033
		x_4	14 / 42 / 0.030	11 / 33 / 0.025	10 / 40 / 0.024	14 / 42 / 0.031
		x_5	17 / 51 / 0.034	11 / 33 / 0.021	36 / 179 / 0.099	17 / 51 / 0.035
		x_6	17 / 51 / 0.036	11 / 33 / 0.021	39 / 194 / 0.105	18 / 54 / 0.033
		x_7	21 / 58 / 0.043	14 / 39 / 0.030	48 / 228 / 0.129	22 / 61 / 0.042
		x_8	21 / 58 / 0.041	14 / 39 / 0.027	48 / 228 / 0.140	22 / 61 / 0.045
	200000	x_1	19 / 56 / 0.191	13 / 37 / 0.137	11 / 43 / 0.123	19 / 56 / 0.171
		x_2	19 / 57 / 0.195	12 / 36 / 0.143	11 / 43 / 0.122	19 / 57 / 0.193
		x_3	18 / 55 / 0.189	11 / 33 / 0.117	10 / 42 / 0.123	18 / 55 / 0.166
		x_4	15 / 45 / 0.161	12 / 36 / 0.136	10 / 40 / 0.126	15 / 45 / 0.142
		x_5	17 / 51 / 0.186	11 / 33 / 0.111	39 / 192 / 0.516	18 / 54 / 0.203
		x_6	18 / 54 / 0.180	11 / 33 / 0.114	39 / 194 / 0.527	18 / 54 / 0.173
		x_7	21 / 58 / 0.199	14 / 39 / 0.141	50 / 238 / 0.670	21 / 60 / 0.192
		x_8	21 / 58 / 0.222	14 / 39 / 0.142	50 / 238 / 0.663	21 / 60 / 0.197
TP2	50000	x_1	1 / 15 / 0.009	15 / 59 / 0.046	1 / 37 / 0.017	1 / 15 / 0.007
		x_2	1 / 2 / 0.002	1 / 2 / 0.002	1 / 2 / 0.002	1 / 2 / 0.002
		x_3	1 / 2 / 0.002	1 / 2 / 0.002	1 / 2 / 0.001	1 / 2 / 0.001
		x_4	16 / 33 / 0.021	11 / 23 / 0.018	1 / 3 / 0.001	16 / 33 / 0.020
		x_5	13 / 27 / 0.012	10 / 22 / 0.010	31 / 64 / 0.028	13 / 27 / 0.013
		x_6	15 / 31 / 0.021	11 / 25 / 0.015	33 / 71 / 0.041	14 / 29 / 0.016
		x_7	19 / 39 / 0.026	14 / 31 / 0.020	55 / 116 / 0.082	19 / 39 / 0.022
		x_8	19 / 39 / 0.030	14 / 31 / 0.021	55 / 116 / 0.075	19 / 39 / 0.024
	200000	x_1	1 / 15 / 0.045	16 / 61 / 0.220	1 / 37 / 0.085	1 / 15 / 0.045
		x_2	1 / 2 / 0.009	1 / 2 / 0.007	1 / 2 / 0.009	1 / 2 / 0.007
		x_3	1 / 2 / 0.007	1 / 2 / 0.006	1 / 2 / 0.006	1 / 2 / 0.006
		x_4	17 / 35 / 0.131	12 / 25 / 0.113	1 / 3 / 0.007	17 / 34 / 0.124
		x_5	13 / 27 / 0.095	10 / 22 / 0.069	31 / 64 / 0.176	13 / 27 / 0.075
		x_6	15 / 31 / 0.112	11 / 25 / 0.096	33 / 71 / 0.238	14 / 29 / 0.109
		x_7	20 / 41 / 0.167	15 / 33 / 0.121	57 / 120 / 0.421	20 / 41 / 0.150
		x_8	20 / 41 / 0.166	15 / 33 / 0.120	57 / 120 / 0.413	20 / 41 / 0.144
TP3	50000	x_1	2 / 6 / 0.005	14 / 31 / 0.022	1 / 6 / 0.007	2 / 6 / 0.007
		x_2	1 / 4 / 0.005	1 / 3 / 0.003	7 / 18 / 0.012	1 / 4 / 0.004
		x_3	1 / 3 / 0.002	1 / 3 / 0.002	7 / 15 / 0.010	1 / 3 / 0.002
		x_4	16 / 33 / 0.022	12 / 25 / 0.017	6 / 13 / 0.007	16 / 32 / 0.017
		x_5	13 / 27 / 0.015	9 / 19 / 0.009	20 / 41 / 0.018	13 / 27 / 0.011
		x_6	14 / 29 / 0.016	10 / 21 / 0.011	37 / 75 / 0.035	14 / 29 / 0.013
		x_7	18 / 37 / 0.024	13 / 27 / 0.015	47 / 95 / 0.059	18 / 37 / 0.025
		x_8	18 / 37 / 0.024	13 / 27 / 0.014	47 / 95 / 0.059	18 / 37 / 0.021
	200000	x_1	2 / 6 / 0.030	15 / 33 / 0.148	1 / 6 / 0.029	2 / 6 / 0.031
		x_2	1 / 4 / 0.019	1 / 3 / 0.013	7 / 18 / 0.065	1 / 4 / 0.019
		x_3	1 / 3 / 0.008	1 / 3 / 0.010	7 / 15 / 0.047	1 / 3 / 0.010
		x_4	17 / 35 / 0.131	12 / 25 / 0.099	6 / 13 / 0.036	15 / 30 / 0.091
		x_5	13 / 27 / 0.080	9 / 19 / 0.057	20 / 41 / 0.119	13 / 27 / 0.068
		x_6	14 / 29 / 0.101	10 / 21 / 0.068	37 / 75 / 0.221	14 / 29 / 0.088
		x_7	18 / 37 / 0.140	14 / 29 / 0.111	49 / 99 / 0.300	18 / 37 / 0.109
		x_8	18 / 37 / 0.126	14 / 29 / 0.115	49 / 99 / 0.330	18 / 37 / 0.117
TP4	50000	x_1	14 / 56 / 0.044	27 / 131 / 0.107	6 / 48 / 0.032	14 / 56 / 0.046
		x_2	14 / 56 / 0.045	27 / 132 / 0.082	6 / 44 / 0.031	14 / 56 / 0.041
		x_3	13 / 53 / 0.038	27 / 134 / 0.093	5 / 41 / 0.021	13 / 53 / 0.035
		x_4	14 / 57 / 0.040	21 / 103 / 0.069	5 / 40 / 0.023	14 / 57 / 0.034
		x_5	23 / 93 / 0.062	29 / 144 / 0.100	45 / 360 / 0.197	19 / 77 / 0.050
		x_6	23 / 93 / 0.055	29 / 144 / 0.105	52 / 416 / 0.233	20 / 81 / 0.049
		x_7	30 / 120 / 0.090	30 / 148 / 0.093	70 / 556 / 0.317	23 / 92 / 0.056
		x_8	30 / 120 / 0.077	30 / 148 / 0.099	70 / 556 / 0.336	23 / 92 / 0.067
	200000	x_1	15 / 60 / 0.198	28 / 136 / 0.472	6 / 48 / 0.156	15 / 60 / 0.196
		x_2	15 / 60 / 0.195	28 / 137 / 0.406	6 / 44 / 0.132	15 / 60 / 0.199
		x_3	13 / 53 / 0.164	28 / 139 / 0.416	5 / 41 / 0.098	13 / 53 / 0.150
		x_4	14 / 57 / 0.183	22 / 108 / 0.314	5 / 40 / 0.100	14 / 57 / 0.156
		x_5	23 / 93 / 0.279	30 / 149 / 0.463	45 / 360 / 0.838	20 / 81 / 0.218
		x_6	23 / 93 / 0.264	30 / 149 / 0.473	53 / 422 / 0.986	20 / 81 / 0.220
		x_7	31 / 124 / 0.368	31 / 153 / 0.506	72 / 572 / 1.386	23 / 92 / 0.262
		x_8	31 / 124 / 0.374	31 / 153 / 0.469	72 / 572 / 1.395	23 / 92 / 0.257

Table 2. The results of four algorithms for Test Problems 5, 6, 7, and 8.

Problem	Dim	Sta	Zhang Niter/Nfun/Time	Gao Niter/Nfun/Time	Abubakar Niter/Nfun/Time	Algorithm 2.1 Niter/Nfun/Time
TP5	50000	x_1	9 / 45 / 0.197	27 / 157 / 0.758	9 / 106 / 0.478	9 / 56 / 0.215
		x_2	9 / 24 / 0.040	21 / 85 / 0.111	9 / 47 / 0.066	9 / 24 / 0.040
		x_3	5 / 20 / 0.022	14 / 67 / 0.079	6 / 51 / 0.045	5 / 20 / 0.025
		x_4	4 / 17 / 0.021	13 / 64 / 0.067	5 / 46 / 0.045	4 / 17 / 0.020
		x_5	33 / 142 / 0.098	12 / 59 / 0.041	48 / 419 / 0.261	11 / 45 / 0.028
		x_6	34 / 147 / 0.135	14 / 72 / 0.068	48 / 436 / 0.362	13 / 52 / 0.058
		x_7	38 / 167 / 0.190	17 / 87 / 0.095	fail / fail / fail	15 / 62 / 0.073
		x_8	37 / 164 / 0.168	17 / 87 / 0.106	fail / fail / fail	15 / 62 / 0.063
	200000	x_1	9 / 45 / 0.802	29 / 165 / 3.370	9 / 106 / 1.911	9 / 56 / 0.875
		x_2	9 / 24 / 0.179	21 / 85 / 0.456	9 / 47 / 0.256	9 / 24 / 0.174
		x_3	5 / 20 / 0.091	14 / 67 / 0.313	6 / 51 / 0.176	5 / 20 / 0.089
		x_4	4 / 17 / 0.070	13 / 64 / 0.279	5 / 46 / 0.151	4 / 17 / 0.067
		x_5	33 / 142 / 0.462	12 / 59 / 0.203	48 / 419 / 1.064	11 / 45 / 0.131
		x_6	32 / 139 / 0.521	14 / 72 / 0.265	48 / 436 / 1.342	13 / 52 / 0.196
		x_7	34 / 150 / 0.656	17 / 87 / 0.397	fail / fail / fail	15 / 62 / 0.266
		x_8	43 / 174 / 0.859	17 / 87 / 0.451	fail / fail / fail	15 / 62 / 0.250
TP6	50000	x_1	41 / 269 / 0.102	32 / 287 / 0.103	69 / 1169 / 0.378	37 / 239 / 0.085
		x_2	140 / 1211 / 0.420	320 / 4034 / 1.358	fail / fail / fail	144 / 1415 / 0.479
		x_3	45 / 283 / 0.104	61 / 471 / 0.165	54 / 809 / 0.258	31 / 188 / 0.069
		x_4	51 / 313 / 0.117	388 / 2730 / 1.005	177 / 2499 / 0.813	30 / 180 / 0.063
		x_5	34 / 208 / 0.075	27 / 208 / 0.076	50 / 746 / 0.245	31 / 188 / 0.069
		x_6	44 / 268 / 0.097	27 / 210 / 0.075	55 / 820 / 0.264	31 / 189 / 0.067
		x_7	49 / 308 / 0.113	495 / 3482 / 1.295	68 / 1070 / 0.344	39 / 244 / 0.086
		x_8	53 / 336 / 0.122	31 / 244 / 0.088	75 / 1193 / 0.374	34 / 216 / 0.077
	200000	x_1	40 / 265 / 0.572	47 / 448 / 0.916	67 / 1123 / 2.037	36 / 235 / 0.477
		x_2	180 / 1562 / 3.159	fail / fail / fail	50 / 903 / 1.714	221 / 2338 / 4.424
		x_3	42 / 262 / 0.588	59 / 453 / 0.967	53 / 794 / 1.399	32 / 194 / 0.403
		x_4	47 / 292 / 0.635	404 / 2835 / 6.352	99 / 1401 / 2.509	32 / 193 / 0.397
		x_5	53 / 328 / 0.715	30 / 232 / 0.504	52 / 779 / 1.373	31 / 189 / 0.398
		x_6	42 / 263 / 0.581	28 / 218 / 0.469	55 / 824 / 1.472	35 / 213 / 0.448
		x_7	66 / 414 / 0.907	399 / 2816 / 6.304	75 / 1202 / 2.128	38 / 239 / 0.483
		x_8	59 / 378 / 0.828	32 / 252 / 0.543	76 / 1211 / 2.237	37 / 234 / 0.508
TP7	50000	x_1	62 / 217 / 0.078	112 / 487 / 0.172	46 / 372 / 0.114	66 / 239 / 0.085
		x_2	76 / 257 / 0.092	101 / 432 / 0.150	59 / 481 / 0.141	71 / 255 / 0.091
		x_3	69 / 237 / 0.089	139 / 563 / 0.202	47 / 383 / 0.115	64 / 231 / 0.081
		x_4	69 / 233 / 0.087	136 / 548 / 0.203	46 / 369 / 0.119	61 / 219 / 0.086
		x_5	64 / 217 / 0.082	100 / 432 / 0.156	44 / 356 / 0.103	60 / 216 / 0.073
		x_6	62 / 212 / 0.078	94 / 394 / 0.143	46 / 375 / 0.111	58 / 209 / 0.074
		x_7	63 / 216 / 0.078	93 / 406 / 0.148	47 / 384 / 0.116	62 / 224 / 0.081
		x_8	63 / 216 / 0.078	93 / 406 / 0.144	47 / 382 / 0.111	62 / 224 / 0.081
	200000	x_1	61 / 212 / 0.520	114 / 501 / 1.400	42 / 344 / 0.617	69 / 248 / 0.565
		x_2	76 / 262 / 0.668	101 / 444 / 1.087	60 / 490 / 0.908	70 / 252 / 0.557
		x_3	70 / 243 / 0.618	102 / 445 / 1.098	53 / 427 / 0.835	65 / 234 / 0.559
		x_4	71 / 241 / 0.672	97 / 421 / 1.077	46 / 370 / 0.692	65 / 233 / 0.511
		x_5	68 / 233 / 0.619	92 / 399 / 0.966	46 / 368 / 0.664	62 / 223 / 0.486
		x_6	56 / 194 / 0.508	127 / 513 / 1.289	44 / 355 / 0.644	59 / 213 / 0.472
		x_7	63 / 219 / 0.551	95 / 417 / 1.014	45 / 363 / 0.666	66 / 237 / 0.559
		x_8	63 / 219 / 0.530	95 / 417 / 1.010	45 / 363 / 0.669	66 / 237 / 0.538
TP8	50000	x_1	21 / 140 / 0.149	28 / 215 / 0.232	1 / 46 / 0.077	21 / 140 / 0.149
		x_2	1 / 5 / 0.008	1 / 4 / 0.007	2 / 21 / 0.019	1 / 5 / 0.008
		x_3	1 / 6 / 0.006	1 / 5 / 0.006	2 / 24 / 0.023	1 / 6 / 0.007
		x_4	18 / 109 / 0.091	25 / 173 / 0.163	1 / 13 / 0.011	18 / 109 / 0.097
		x_5	373 / 2929 / 1.437	26 / 90 / 0.053	fail / fail / fail	12 / 58 / 0.033
		x_6	146 / 1338 / 1.183	8 / 36 / 0.035	fail / fail / fail	8 / 34 / 0.033
		x_7	284 / 1640 / 1.546	48 / 222 / 0.150	65 / 936 / 0.759	56 / 299 / 0.273
		x_8	300 / 1714 / 1.616	32 / 156 / 0.112	32 / 984 / 3.637	45 / 243 / 0.224
	200000	x_1	22 / 146 / 0.601	29 / 222 / 1.012	1 / 46 / 0.308	22 / 146 / 0.596
		x_2	1 / 5 / 0.033	1 / 4 / 0.028	2 / 21 / 0.089	1 / 5 / 0.032
		x_3	1 / 6 / 0.037	1 / 5 / 0.026	2 / 24 / 0.099	1 / 6 / 0.029
		x_4	19 / 115 / 0.385	25 / 173 / 0.650	1 / 13 / 0.047	19 / 115 / 0.373
		x_5	373 / 2929 / 6.855	26 / 90 / 0.313	fail / fail / fail	12 / 58 / 0.166
		x_6	fail / fail / fail	8 / 36 / 0.152	fail / fail / fail	8 / 34 / 0.155
		x_7	318 / 1853 / 8.556	67 / 298 / 1.056	65 / 1010 / 4.233	62 / 334 / 1.378
		x_8	305 / 2005 / 9.320	43 / 188 / 0.723	34 / 584 / 5.046	49 / 261 / 1.107

Table 3. The results of four algorithms for Test Problems 9 and 10.

Problem	Dim	Sta	Zhang Niter/Nfun/Time	Gao Niter/Nfun/Time	Abubakar Niter/Nfun/Time	Algorithm 2.1 Niter/Nfun/Time
TP9	50000	x_1	1 / 2 / 0.004	1 / 2 / 0.004	1 / 2 / 0.004	1 / 2 / 0.004
		x_2	1 / 2 / 0.004	1 / 2 / 0.005	1 / 2 / 0.004	1 / 2 / 0.005
		x_3	1 / 2 / 0.008	1 / 2 / 0.007	1 / 2 / 0.006	1 / 2 / 0.007
		x_4	3 / 4 / 0.016	4 / 8 / 0.027	4 / 5 / 0.019	3 / 4 / 0.013
		x_5	3 / 4 / 0.005	3 / 4 / 0.005	3 / 4 / 0.005	3 / 4 / 0.004
		x_6	4 / 5 / 0.015	3 / 4 / 0.011	3 / 4 / 0.014	4 / 5 / 0.015
		x_7	2 / 3 / 0.012	2 / 3 / 0.009	2 / 3 / 0.006	2 / 3 / 0.011
		x_8	2 / 3 / 0.014	2 / 3 / 0.006	2 / 3 / 0.005	2 / 3 / 0.013
	200000	x_1	1 / 2 / 0.017	1 / 2 / 0.017	1 / 2 / 0.017	1 / 2 / 0.017
		x_2	1 / 2 / 0.019	1 / 2 / 0.018	1 / 2 / 0.017	1 / 2 / 0.018
		x_3	1 / 2 / 0.025	1 / 2 / 0.024	1 / 2 / 0.024	1 / 2 / 0.028
		x_4	3 / 4 / 0.072	4 / 8 / 0.087	4 / 5 / 0.069	3 / 4 / 0.039
		x_5	3 / 4 / 0.022	3 / 4 / 0.023	3 / 4 / 0.023	3 / 4 / 0.021
		x_6	4 / 5 / 0.062	3 / 4 / 0.053	3 / 4 / 0.049	4 / 5 / 0.063
		x_7	2 / 3 / 0.053	2 / 3 / 0.028	2 / 3 / 0.029	2 / 3 / 0.044
		x_8	2 / 3 / 0.050	2 / 3 / 0.034	2 / 3 / 0.028	2 / 3 / 0.051
TP10	50000	x_1	2 / 6 / 0.006	14 / 31 / 0.022	1 / 6 / 0.005	2 / 6 / 0.007
		x_2	1 / 4 / 0.004	1 / 3 / 0.002	2 / 8 / 0.007	1 / 4 / 0.003
		x_3	16 / 34 / 0.020	1 / 4 / 0.003	7 / 19 / 0.009	16 / 34 / 0.022
		x_4	16 / 33 / 0.017	12 / 25 / 0.016	6 / 13 / 0.008	16 / 32 / 0.022
		x_5	13 / 27 / 0.013	9 / 19 / 0.010	20 / 42 / 0.019	13 / 27 / 0.010
		x_6	14 / 29 / 0.016	10 / 21 / 0.010	24 / 49 / 0.023	14 / 29 / 0.012
		x_7	18 / 37 / 0.024	13 / 27 / 0.014	29 / 60 / 0.031	18 / 37 / 0.019
		x_8	18 / 37 / 0.020	13 / 27 / 0.015	29 / 60 / 0.036	18 / 37 / 0.020
	200000	x_1	2 / 6 / 0.024	15 / 33 / 0.129	1 / 6 / 0.017	2 / 6 / 0.025
		x_2	1 / 4 / 0.013	1 / 3 / 0.011	2 / 8 / 0.030	1 / 4 / 0.015
		x_3	17 / 36 / 0.127	1 / 4 / 0.013	7 / 19 / 0.061	13 / 27 / 0.097
		x_4	17 / 35 / 0.128	12 / 25 / 0.098	6 / 13 / 0.038	15 / 30 / 0.093
		x_5	13 / 27 / 0.087	9 / 19 / 0.067	20 / 42 / 0.117	13 / 27 / 0.074
		x_6	14 / 29 / 0.103	10 / 21 / 0.074	24 / 49 / 0.170	14 / 29 / 0.093
		x_7	18 / 37 / 0.141	14 / 29 / 0.103	31 / 64 / 0.191	18 / 37 / 0.112
		x_8	18 / 37 / 0.130	14 / 29 / 0.107	31 / 64 / 0.210	18 / 37 / 0.118

In an effort to comprehensively compare all algorithms, we utilize the performance profiles that were proposed by Dolan and Moré [26]. The performance profiles concerning the number of iterations are shown in Figure 1. The performance profiles for the number of function evaluations are displayed in Figure 2, while Figure 3 reveals the performance profiles of the CPU time. It is readily observable that the Algorithm 2.1 introduced in this paper outperforms the algorithms presented by Zhang, Gao, and Abubakar in terms of efficiency.

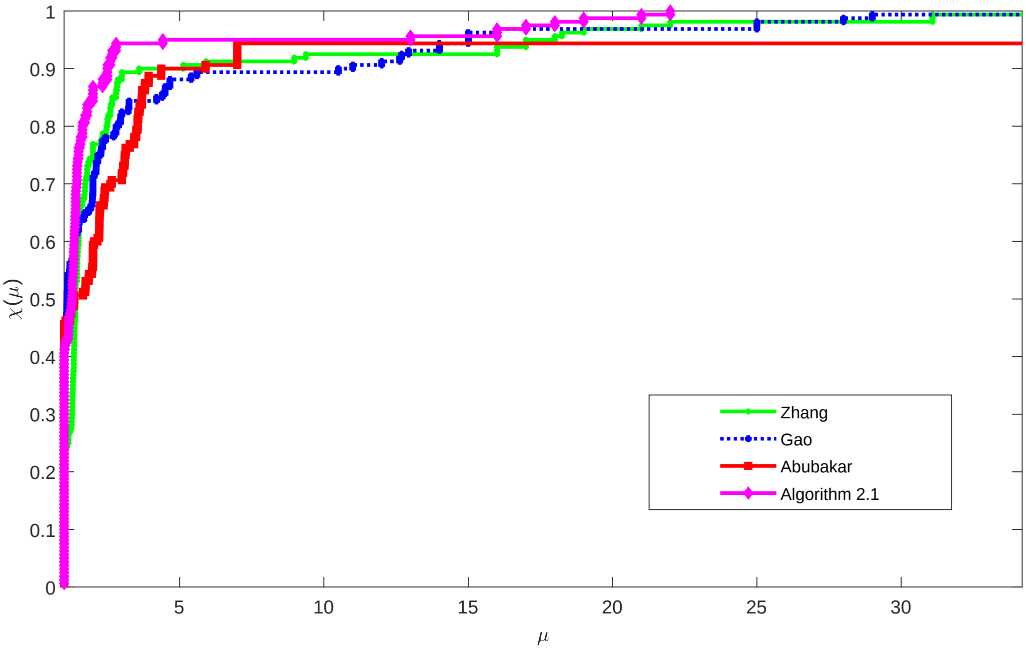

Figure 1. Performance profiles for the number of iterations.

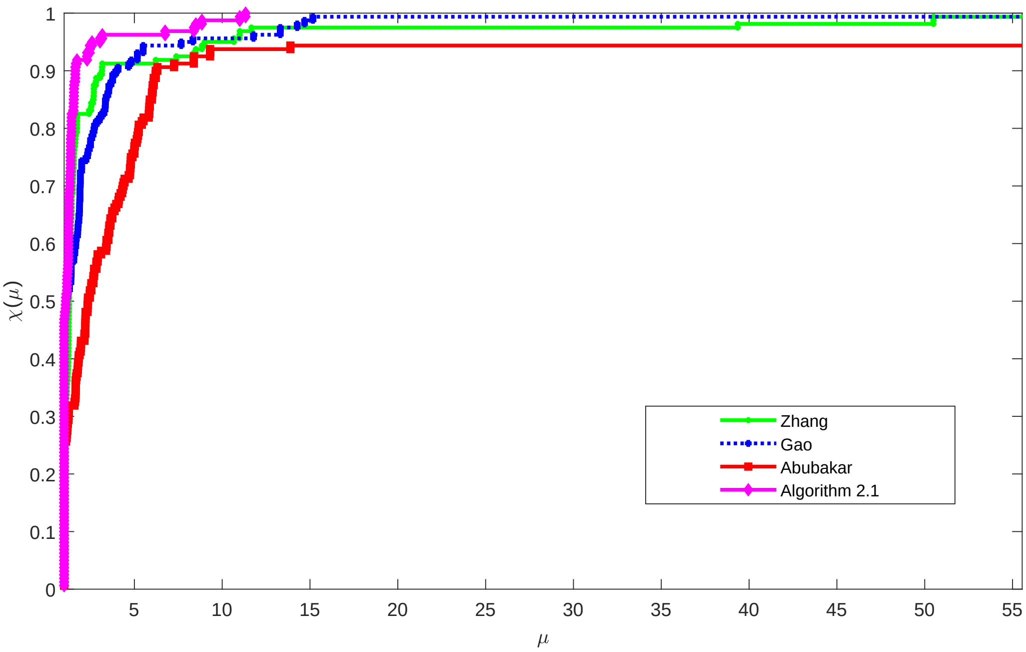

Figure 2. Performance profiles for the number of function evaluations.

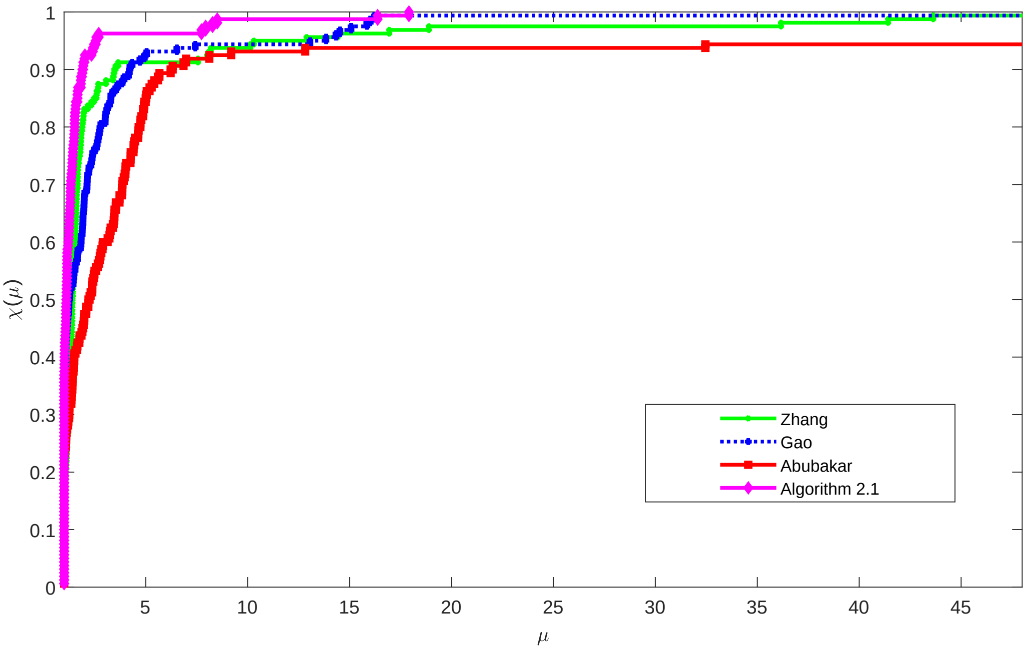

Figure 3. Performance profiles for the CPU time.

5. Applications in sparse signal restoration

In this section, we make use of Algorithm 2.1 to tackle sparse signal restoration problems and compare its performance with that of Zhang [19], Gao [23], and Abubakar [24]. We begin by recalling the compressed sensing model. This model is utilized to retrieve sparse solutions or signals from underdetermined linear systems represented as $Ax = b$, where $b \in \mathbb{R}^m$ represents the data obtained from observations and $A \in \mathbb{R}^{m\times n}(m \ll n)$ denotes a linear mapping. Specifically, we can achieve sparse signal restoration through the solution of the following l_1-norm regularized optimization problem

$$\min_{x\in\mathbb{R}^n} \quad \frac{1}{2}\|Ax - b\|_2^2 + \tau\|x\|_1, \tag{5.1}$$

where τ acts as a constant that reconciles data fitting and sparsity. The vector $x \in \mathbb{R}^n$ can be decomposed as $x = u-v, u \geq 0, v \geq 0$, where $u \in \mathbb{R}^n$, $v \in \mathbb{R}^n$, and $u_i = \max\{x_i, 0\}$, $v_i = \max\{-x_i, 0\}$ for $i = 1, 2, \cdots, n$. Therefore, we can replace problem (5.1) with the next programming formulation:

$$\min_{u,v\geq 0} \frac{1}{2}\|A(u-v) - b\|_2^2 + \tau e_n^T u + \tau e_n^T v, \tag{5.2}$$

in which $e_n = (1, 1, \cdots, 1)^T \in \mathbb{R}^n$.

Further, we obtain its standard form

$$\min_{z\geq 0} \frac{1}{2} z^T Hz + c^T z, \tag{5.3}$$

in which

$$z = \begin{bmatrix} u \\ v \end{bmatrix},\ c = \begin{bmatrix} \tau e_n - A^T b \\ \tau e_n + A^T b \end{bmatrix},\ H = \begin{bmatrix} A^T A & -A^T A \\ -A^T A & A^T A \end{bmatrix}. \tag{5.4}$$

Evidently, matrix H has semi-positive definiteness; thus, Eq (5.3) is a convex quadratic problem. Based on first-order optimality conditions for (5.3), z qualifies as a minimizer if and only if it satisfies the subsequent equations:

$$F(z) = \min\{z, Hz + c\} = 0, z \geq 0, \tag{5.5}$$

in which $F(z)$ is monotone and continuous. Henceforth, our proposed method may be employed for resolving (5.5).

During these experiments, the primary objective is to reconstruct a one-dimensional sparse signal with a length of n from $m(m \ll n)$ observations. The performance of signal restoration is evaluated using the mean squared error (MSE), defined as

$$\text{MSE} = \frac{1}{n}\|x - x^*\|^2, \tag{5.6}$$

in which x^* represents the recovered signal and x indicates the initial signal. Let $f(x) = \frac{1}{2}\|Ax - b\|_2^2 + \tau\|x\|_1$ serves as the auxiliary function. The iterative sequence commences with the initial measurement signal $x_0 = A^T b$ and comes to an end when

$$\frac{\|f(x_k) - f(x_{k-1})\|}{\|f(x_{k-1})\|} < 10^{-5}. \tag{5.7}$$

During our experiments, we define r as the quantity of nonzero entries in x. For a specified set of values of m, n, and r, we generate the subsequent test data using MATLAB:

$$\begin{aligned} x &= zeros(n, 1); \\ q &= randperm(n); \\ x(q(1:r)) &= randn(r, 1); \\ b &= orth(randn(m, n)')' * x + 0.001 * randn(m, 1); \\ x_0 &= A' * b. \end{aligned} \tag{5.8}$$

The numerical outcomes presented in Figure 4 contain the original sparse signal, the measurement data, and the reconstructed signals yielded by four algorithms. Clearly, all four algorithms reconstruct the original signal from the measurements with near perfection. However, Algorithm 2.1 outperforms the other algorithms in terms of MSE, the number of iterations, and CPU time. Figure 5 depicts four graphs to visually assess the performance of the four algorithms. These graphs display the convergence characteristics of the algorithms, depicting the changes in the values of the merit function and the MSE as the iterations progress and as the CPU time is consumed. Table 4 displays the numerical outcomes obtained from the experiment on ten diverse noise samples. Evidently, in a majority of cases, Algorithm 2.1 outperforms other algorithms. It achieves a lower MSE, demands a smaller number of iterations, and consumes less time on the CPU.

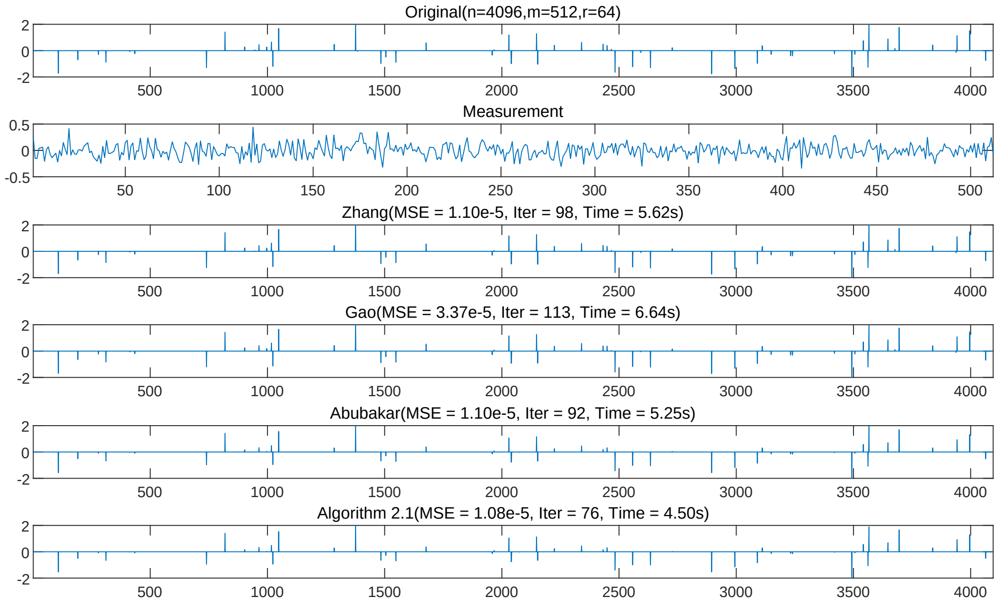

Figure 4. From top to bottom: the original signal, the measurement, and the reconstructed signals by four algorithms.

Table 4. The results of four algorithms in ten experiments on the signal recovery problem.

	Zhang Niter/Time/MSE	Gao Niter/Time/MSE	Abubakar Niter/Time/MSE	Algorithm 2.1 Niter/Time/MSE
	87 / 4.47 / 2.07E-05	116 / 6.20 / 2.39E-05	85 / 4.80 / 1.26E-05	70 / 3.73 / 1.93E-05
	144 / 7.28 / 1.21E-05	150 / 7.81 / 4.06E-05	103 / 5.75 / 1.41E-05	89 / 4.78 / 1.20E-05
	120 / 6.18 / 1.26E-05	132 / 6.92 / 5.45E-05	112 / 6.21 / 1.61E-05	66 / 3.51 / 1.34E-05
	135 / 6.85 / 1.77E-05	124 / 6.47 / 9.44E-05	139 / 7.63 / 1.88E-05	95 / 4.99 / 1.05E-05
	146 / 7.40 / 1.72E-05	136 / 7.14 / 7.16E-05	115 / 6.32 / 2.32E-05	91 / 4.74 / 1.62E-05
	115 / 5.90 / 1.17E-05	127 / 6.62 / 4.50E-05	89 / 4.94 / 1.66E-05	70 / 3.66 / 1.75E-05
	153 / 7.77 / 1.25E-05	161 / 8.31 / 3.99E-05	157 / 8.51 / 1.49E-05	105 / 5.44 / 1.16E-05
	104 / 5.38 / 1.21E-05	113 / 6.10 / 4.62E-05	103 / 5.73 / 1.30E-05	82 / 4.42 / 1.15E-05
	132 / 6.71 / 1.32E-05	132 / 7.01 / 4.64E-05	95 / 5.34 / 1.34E-05	85 / 4.48 / 1.30E-05
	164 / 8.26 / 1.69E-05	160 / 8.22 / 7.25E-05	113 / 6.23 / 2.17E-05	108 / 5.58 / 1.74E-05
Average	130 / 6.62 / 1.47E-05	135 / 7.08 / 5.35E-05	111 / 6.15 / 1.64E-05	86 / 4.53 / 1.42E-05

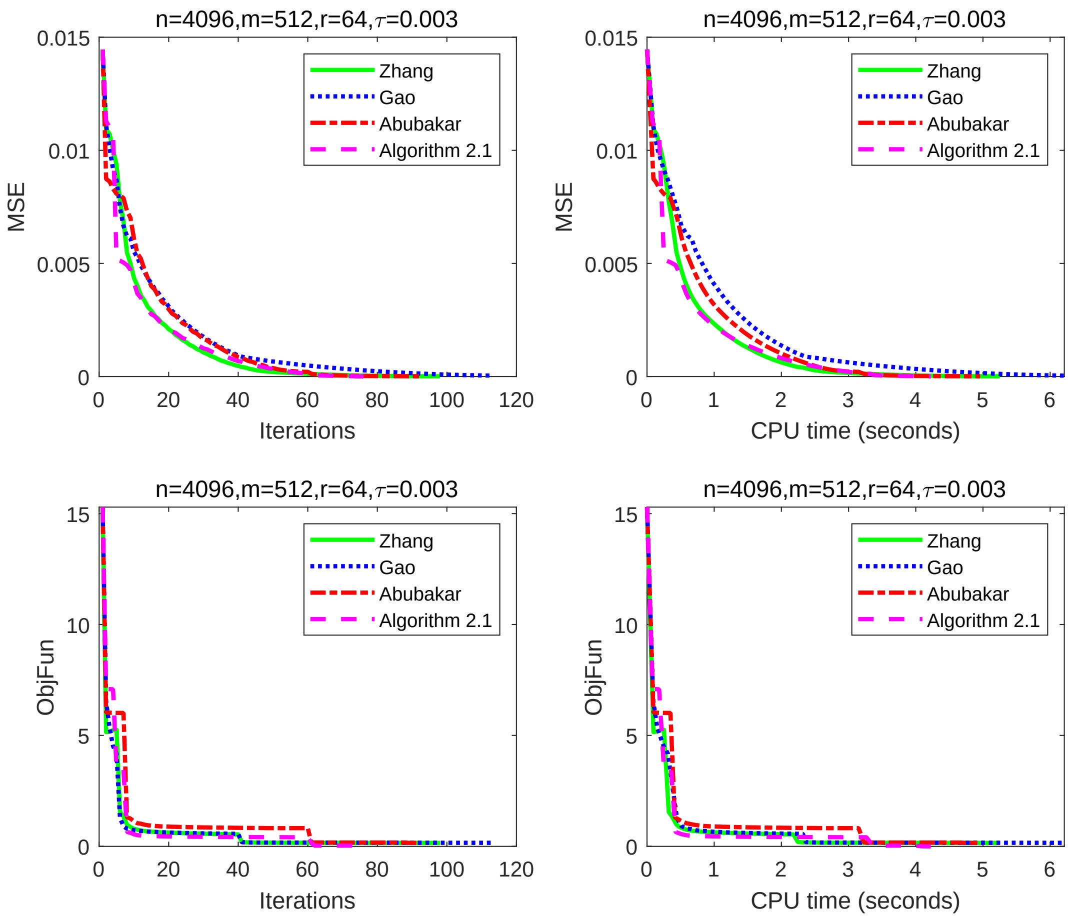

Figure 5. The upper graphs show MSE against iterations and CPU time, and the lower graphs show objective function value against iterations and CPU time.

6. Conclusions

This paper presents a derivative-free conjugate gradient method for the purpose of solving large-scale nonlinear systems of monotone equations that are bounded by convex constraints. The proposed approach is a novel expansion of the RMIL conjugate gradient method. It incorporates projection techniques, which play a crucial role in ensuring that the iterates remain within the feasible region defined by the convex constraints. This not only helps in maintaining the validity of the solution process but also improves the efficiency of the algorithm by avoiding unnecessary computations outside the constraint set. Additionally, the method features a line search strategy that operates without using derivatives. Under certain reasonable assumptions, we prove the global convergence property of the method. To further demonstrate the practical performance of our proposed method, we carry out extensive numerical experiments. The experimental results illustrate that the method we proposed has great promise. It outperforms several existing methods in terms of computational efficiency and accuracy when solving large-scale nonlinear systems of monotone equations. Furthermore, our numerical experiments on sparse signal restoration problems further validate the effectiveness and practicality of the method.

Acknowledgments

This research was supported by the Zhejiang Provincial Natural Science Foundation of China under Grant No. LY23A010007 and the Huzhou Natural Science Foundation under Grant No. 2023YZ29.

Conflict of interest

The author declares to have no competing interests.

Use of Generative-AI tools declaration

The author declares they have not used Artificial Intelligence (AI) tools in the creation of this article.

References

1. S. Prajna, P. A. Parrilo, A. Rantzer, Nonlinear control synthesis by convex optimization, *IEEE Trans. Autom. Control.,* **49** (2004), 310–314. https://doi.org/10.1109/TAC.2003.823000
2. Y. P. Hu, Y. J. Wang, An efficient projected gradient method for convex constrained monotone equations with applications in compressive sensing, *J. Appl. Math. Phys.,* **8** (2020), 983–998. https://doi.org/10.4236/jamp.2020.86077
3. J. K. Liu, L. Du, A gradient projection method for the sparse signal reconstruction in compressive sensing, *Appl. Anal.,* **97** (2018), 2122–2131. https://doi.org/10.1080/00036811.2017.1359556
4. Y. H. Xiao, Q. Y. Wang, Q. J. Hu, Non-smooth equations based method for l1-norm problems with applications to compressive sensing, *Nonlinear Anal. Theory Methods Appl.,* **74** (2011), 3570–3577. https://doi.org/10.1016/j.na.2011.02.040
5. B. Ghaddar, J. Marecek, M. Mevissen, Optimal power flow as a polynomial optimization problem, *IEEE Trans. Power Syst.,* **31** (2016), 539–546. https://doi.org/10.1109/tpwrs.2015.2390037
6. G. Zhou, K. C. Toh, Superliner convergence of a Newton-type algorithm for monotone equations, *J. Optim. Theory Appl.,* **125** (2005), 205–221. https://doi.org/10.1007/s10957-004-1721-7
7. X. W. Fang, Q. Ni, M. L. Zeng, A modified quasi-Newton method for nonlinear equations, *J. Comput. Appl. Math.,* **328** (2018), 44–58. https://doi.org/10.1016/j.cam.2017.06.024
8. D. H. Li, M. Fukushima, A globally and superlinearly convergent Gauss-Newton based BFGS method for symmetric equations, *SIAM. J. Numer. Anal.,* **37** (1999), 152–172. https://doi.org/10.1137/s0036142998335704
9. G. L. Yuan, Z. X. Wei, X. W. Lu, A BFGS trust-region method for nonlinear equations, *Computing,* **92** (2011), 317–333. https://doi.org/10.1007/s00607-011-0146-z
10. M. V. Solodov, B. F. Svaiter, *A globally convergent inexact Newton method for systems of monotone equations*, In: M. Fukushima, L. Qi, Reformulation: nonsmooth, piecewise smooth, semismooth and smoothing methods, Springer, **22** (1998), 355–369. https://doi.org/10.1007/978-1-4757-6388-1_18

11. Q. Li, D. H. Li, A class of derivative-free methods for large-scale nonlinear monotone equations, *IMA J. Numer. Anal.,* **31** (2011), 1625–1635. https://doi.org/10.1093/imanum/drq015

12. W. Y. Cheng, A PRP type method for systems of monotone equations, *Math. Comput. Model.,* **50** (2009), 15–20. https://doi.org/10.1016/j.mcm.2009.04.007

13. A. M. Awwal, P. Kumam, K. Sitthithakerngkiet, A. M. Bakoji, A. S. Halilu, I. M. Sulaiman, et al., Derivative-free method based on DFP updating formula for solving convex constrained nonlinear monotone equations and application, *AIMS Math.,* **6** (2021), 8792-8814. https://doi.org/10.3934/math.2021510

14. M. Ahookhosh, K. Amini, S. Bahrami, Two derivative-free projection approaches for systems of large-scale nonlinear monotone equations, *Numer. Algorithms,* **64** (2013), 21–42. https://doi.org/10.1007/s11075-012-9653-z

15. M. Rivaie, M. Mamat, L. W. June, I. Mohd, A new class of nonlinear conjugate gradient coefficients with global convergence properties, *App. Math. Comput.,* **218** (2012), 11323–11332. https://doi.org/10.1016/j.amc.2012.05.030

16. X. W. Fang, Q. Ni, A new derivative-free conjugate gradient method for large-scale nonlinear systems of equations, *B. Aust. Math. Soc.,* **95** (2017), 500–511. https://doi.org/10.1017/s0004972717000168

17. X. W. Fang, A class of new derivative-free gradient type methods for large-scale nonlinear systems of monotone equations, *J. Inequal. Appl.,* **2020** (2020), 1–13. https://doi.org/10.1186/s13660-020-02361-5

18. P. Liu, H. Shao, Y. Wang, X. Wu, A three-term CGPM-based algorithm without Lipschitz continuity for constrained nonlinear monotone equations with applications, *Appl. Numer. Math.,* **175** (2022), 98–107. https://doi.org/10.1016/j.apnum.2022.02.001

19. N. Zhang, J. K. Liu, B. Tang, A three-term projection method based on spectral secant equation for nonlinear monotone equations, *Jpn. J. Ind. Appl. Math.,* **41** (2024), 617–635. https://doi.org/10.1007/s13160-023-00624-4

20. M. Abdullahi, A. B. Abubakar, S. B. Salihu, Global convergence via modified self-adaptive approach for solving constrained monotone nonlinear equations with application to signal recovery problems, *Rairo-oper. Res.,* **57** (2023), 2561–2584. https://doi.org/10.1051/ro/2023099

21. M. Abdullahi, A. B. Abubakar, K. Muangchoo, Modified three-term derivative-free projection method for solving nonlinear monotone equations with application, *Numer. Algorithms,* **95** (2024), 1459-1474. https://doi.org/10.1007/s11075-023-01616-8

22. S. Aji, A. M. Awwal, A. B. Muhammadu, C. Khunpanuk, N. Pakkaranang, B. Panyanak, A new spectral method with inertial technique for solving system of nonlinear monotone equations and applications, *AIMS Math.,* **8** (2023), 4442-4466. https://doi.org/10.3934/math.2023221

23. P. Gao, T. Wang, X. Liu, Y. Wu, An efficient three-term conjugate gradient-based algorithm involving spectral quotient for solving convex constrained monotone nonlinear equations with applications, *Comput. Appl. Math.,* **41** (2022), 89. https://doi.org/10.1007/s40314-022-01796-4

24. A. B. Abubakar, P. Kumam, H. Mohammad, A. H. Ibrahim, PRP-like algorithm for monotone operator equations, *Japan J. Indust. Appl. Math.,* **38** (2021), 805–822. https://doi.org/10.1007/s13160-021-00462-2

25. J. Sabiu, S. Sirisubtawee, An inertial Dai-Liao conjugate method for convex constrained monotone equations that avoids the direction of maximum magnification, *J. Appl. Math. Comput.,* **70** (2024), 4319–4351. https://doi.org/10.1007/s12190-024-02123-2

26. E. D. Dolan, J. J. Moré, Benchmarking optimization software with performance profiles, *Math. Program.,* **91** (2001), 201–213. https://doi.org/10.1007/s101070100263