mathematics

MDPI

Article

A Modified Fletcher–Reeves Conjugate Gradient Method for Monotone Nonlinear Equations with Some Applications

Auwal Bala Abubakar [1,2], Poom Kumam [1,3,4,*], Hassan Mohammad [2], Aliyu Muhammed Awwal [1,5] and Kanokwan Sitthithakerngkiet [6]

1. KMUTTFixed Point Research Laboratory, SCL 802 Fixed Point Laboratory, Science Laboratory Building, Department of Mathematics, Faculty of Science, King Mongkut's University of Technology Thonburi (KMUTT), 126 Pracha-Uthit Road, Bang Mod, Thrung Khru, Bangkok 10140, Thailand
2. Department of Mathematical Sciences, Faculty of Physical Sciences, Bayero University, Kano 700241, Nigeria
3. Center of Excellence in Theoretical and Computational Science (TaCS-CoE), Science Laboratory Building, King Mongkut's University of Technology Thonburi (KMUTT), 126 Pracha-Uthit Road, Bang Mod, Thrung Khru, Bangkok 10140, Thailand
4. Department of Medical Research, China Medical University Hospital, China Medical University, Taichung 40402, Taiwan
5. Department of Mathematics, Faculty of Science, Gombe State University, Gombe 760214, Nigeria
6. Department of Mathematics, Faculty of Applied Science, King Mongkut's University of Technology North Bangkok, 1518 Pracharat 1 Road, Wongsawang, Bangsue, Bangkok 10800, Thailand

* Correspondence: poom.kum@kmutt.ac.th

Received: 24 June 2019; Accepted: 5 August 2019; Published: 15 August 2019

Abstract: One of the fastest growing and efficient methods for solving the unconstrained minimization problem is the conjugate gradient method (CG). Recently, considerable efforts have been made to extend the CG method for solving monotone nonlinear equations. In this research article, we present a modification of the Fletcher–Reeves (FR) conjugate gradient projection method for constrained monotone nonlinear equations. The method possesses sufficient descent property and its global convergence was proved using some appropriate assumptions. Two sets of numerical experiments were carried out to show the good performance of the proposed method compared with some existing ones. The first experiment was for solving monotone constrained nonlinear equations using some benchmark test problem while the second experiment was applying the method in signal and image recovery problems arising from compressive sensing.

Keywords: nonlinear equations; conjugate gradient method; projection method; convex constraints; signal and image processing

MSC: 65K05; 90C52; 90C56; 94A08

1. Introduction

In this paper, we are considering a system of nonlinear monotone equations of the form

$$F(x) = 0, \quad \text{subject to} \quad x \in E, \tag{1}$$

where $E \subseteq R^n$ is closed and convex, $F : R^n \to R^m$, $(m \geq n)$ is continuous and monotone, which means

$$\langle F(x) - F(y), (x - y) \rangle \geq 0, \quad \forall x, y \in R^n.$$

A well-known fact is that under the above assumption, the solution set of (1) is convex unless is empty. It is important to mention that nonlinear monotone equations arise in many practical applications. These and other reasons motivate researchers to develop a large number of class of Iterative methods for solving such systems, for example, see [1–7] among others. In addition, convex constrained equations have application in many scientific fields, some of which are the economic equilibrium problems [8], the chemical equilibrium systems [9], etc. Several algorithms were developed to solve (1), among them, are the trust-region [10] and the Levenberg-Marquardt method [11]. Moreover, the requirement to compute and store the matrix in every iteration makes them ineffective for large-scale nonlinear equations.

Conjugate gradient (CG) methods are efficient for solving large-scale optimization and nonlinear systems because of their low memory requirements. This forms part of the reason several Iterative methods with CG-like directions are proposed in recent years [12,13]. Initially, CG methods and their modified versions are proposed for unconstrained optimization problems [14–19]. Inspired by them, in the last decade, many authors used the CG direction to solve nonlinear monotone equations for both constrained and unconstrained cases. Since in this article, we are interested in solving nonlinear monotone equations with convex constraints, we will only discuss existing methods with such properties.

Many methods for solving nonlinear monotone equations with convex constraints have been presented in the last decade. For examples, Xiao and Zhu [20] presented a CG method, which combines the well-known CG-DESCENT method in [17] and the projection method by Solodov and Svaiter [21]. Liu et al. [22] proposed two CG methods with projection strategy for solving (1). In [23], a modification of the method in [20] was presented by Liu and Li. One of the reasons for the modification was to improve the numerical performance of the method in [20]. Also, Sun and Liu [24] presented derivative-free projection methods for solving nonlinear equations with convex constraints. These methods are the combination of some existing CG methods and the well-known projection method. In addition, a hybrid CG projection method for convex constrained equations was developed in [25]. Ou and Li [26] proposed a combination of a scaled CG method and the projection strategy to solve (1). Furthermore, Ding et al. [27] extended the Dai and Kou (DK) CG method to solve (1) by also combining it with the projection method. Just recently, to popularize the Dai-Yuan (DY) method, Liu and Feng [28] proposed a modified DY method for solving convex constraints monotone equation. The global convergence was also obtained under certain assumptions and finally, some numerical results were reported to show its efficiency.

Inspired by some the above proposals, we present a simple modification of the Fletcher–Reeves (FR) conjugate gradient method [19] considered in [12] to solve nonlinear monotone equations with convex constraints. The modification ensures that the direction is automatically descent, improves its numerical performance and still inherits the nice convergence properties of the method. Under suitable assumptions, we establish the global convergence of the proposed algorithm. Numerical experiments presented show the good performance and competitiveness of the method. In addition, the proposed method has the advantages of the direct methods [29] such as boundary control method by Belishev and Kuryiev [30], the globally convergent method proposed by Beilina and Klibanov [31] and method based on the multidimensional analogs of Gelfand–Levitan–Krein equations [32,33]. The proposed method can be seen as a local method that looks for the closest root. However, there are several global nonlinear solvers that guarantee finding all roots inside a domain and within a very fine double-float accuracy. In some cases a combination of subdivision-based polynomial solver with a decomposition algorithm are employed in order to handle large and complex systems (see for examples [34–36] and references therein).

The remaining part of this article is organized as follows. In Section 2, we mention some preliminaries and present the proposed method. The global convergence of the method is established in Section 3. Finally, Section 4 reports some numerical results to show the performance of the method in solving monotone nonlinear equations with convex constraints, and also apply it to recover a noisy signal and a blurred image.

2. Algorithm

In this section, we define the projection map together with its well-known properties, give some useful assumptions and finally present the proposed algorithm. Throughout this article, $\|\cdot\|$ denotes the Euclidean norm.

Definition 1. *Let $E \subset R^n$ be nonempty closed and convex set. Then for any $x \in R^n$, its projection onto E is defined as*

$$P_E(x) = \arg\min\{\|x-y\| \ : y \in E.\}$$

The following lemma gives some properties of the projection map.

Lemma 1 ([37]). *Suppose $E \subset R^n$ is nonempty, closed and convex set. Then the following statements are true:*

1. $\langle x - P_E(x), P_E(x) - z\rangle \geq 0, \quad \forall x, z \in R^n.$
2. $\|P_E(x) - P_E(y)\| \leq \|x - y\|, \quad \forall x, y \in R^n.$
3. $\|P_E(x) - z\|^2 \leq \|x - z\|^2 - \|x - P_E(x)\|^2, \quad \forall x, z \in R^n.$

Throughout, we suppose the followings

(C_1) The solution set of (1), denoted by E', is nonempty.
(C_2) The mapping F is monotone.
(C_3) The mapping F is Lipschitz continuous, that is there exists a positive constant L such that $\|F(x) - F(y)\| \leq L\|x - y\|, \quad \forall x, y \in R^n.$

Our algorithm is motivated by the work of Papp and Rapajić in [12]. In the paper, they modified the well known Fletcher–Reeves conjugate gradient method to solve unconstrained nonlinear monotone equation. The modification was adding the term $-\theta_k F(x_k)$ to the direction of Fletcher–Reeves. The parameter θ_k was then determined in three different ways and three different directions were proposed, namely, M3TFR1, M3TFR2 and M3TFR3. The direction we are interested in is M3TFR1 and is defined as:

$$d_k = \begin{cases} -F(x_k), & \text{if } k = 0, \\ -F(x_k) + \beta_k^{FR} w_{k-1} + \theta_k F(x_k), & \text{if } k \geq 1, \end{cases} \tag{2}$$

where,

$$\beta_k^{FR} = \frac{\|F(x_k)\|^2}{\|F(x_{k-1})\|^2}, \quad \theta_k = -\frac{F(x_k)^T w_{k-1}}{\|F(x_{k-1})\|^2}, \quad w_{k-1} = z_{k-1} - x_{k-1}, \ z_{k-1} = x_{k-1} + \alpha_{k-1} d_{k-1}.$$

It follows that

$$F(x_k)^T d_k = -\|F(x_k)\|^2.$$

Using same modification proposed in [3], we modify the direction (2) as follows

$$d_k = \begin{cases} -F(x_k), & \text{if } k = 0, \\ -F(x_k) + \frac{\|F(x_k)\|^2 w_{k-1} - F(x_k)^T w_{k-1} F(x_k)}{\max\{\mu\|w_{k-1}\|\|F(x_k)\|, \|F(x_{k-1})\|^2\}}, & \text{if } k \geq 1, \end{cases} \tag{3}$$

where $\mu > 0$ is a positive constant. The difference between the M3TFR1 direction and the direction proposed in this paper is the scaling term appearing in the denominator of Equation (3) i.e., $\max\{\mu\|w_{k-1}\|\|F(x_k)\|, \|F(x_{k-1})\|^2\}$. This modification was shown to have a very good numerical performance in [3] and also helps in obtaining the boundedness of the direction easily.

Remark 1. *Note the the parameter μ is chosen to be strictly positive because if $\mu \leq 0$ then*

$$\max\{\mu\|w_{k-1}\|\|F(x_k)\|, \|F(x_{k-1})\|^2\} = \|F(x_{k-1})\|^2.$$

This means that the direction d_k will always be M3TFR1 given by (2).

3. Convergence Analysis

To prove the global convergence of Algorithm 1, the following results are needed.

Algorithm 1: A modified descent Fletcher–Reeves CG method (MFRM).

Step 0. Select the initial point $x_0 \in R^n$, parameters $\mu > 0$, $\sigma > 0$, $0 < \rho < 1$, $Tol > 0$, and set $k := 0$.
Step 1. If $\|F(x_k)\| \leq Tol$, stop, otherwise go to **Step 2**.
Step 2. Find d_k using (3).
Step 3. Find the step length $\alpha_k = \gamma\rho^{m_k}$ where m_k is the smallest non-negative integer m such that

$$-\langle F(x_k + \alpha_k d_k), d_k\rangle \geq \sigma\alpha_k\|F(x_k + \alpha_k d_k)\|\|d_k\|^2. \tag{4}$$

Step 4. Set $z_k = x_k + \alpha_k d_k$. If $z_k \in E$ and $\|F(z_k)\| \leq Tol$, stop. Else compute

$$x_{k+1} = P_E[x_k - \zeta_k F(z_k)]$$

where

$$\zeta_k = \frac{F(z_k)^T(x_k - z_k)}{\|F(z_k)\|^2}.$$

Step 5. Let $k = k + 1$ and go to Step 1.

Lemma 2. *Let d_k be defined by Equation (3), then*

$$d_k^T F(x_k) = -\|F(x_k)\|^2 \tag{5}$$

and

$$\|F(x_k)\| \leq \|d_k\| \leq \left(1 + \frac{2}{\mu}\right)\|F(x_k)\|. \tag{6}$$

Proof. By Equation (3), suppose $k = 0$,

$$d_k^T F(x_k) = -F(x_k)^T F(x_k) = -\|F(x_k)\|^2.$$

Now suppose $k > 0$,

$$\begin{aligned}
d_k^T F(x_k) &= -F(x_k)^T F(x_k) + \frac{(\|F(x_k)\|^2 w_{k-1})^T F(x_k) - (F(x_k)^T w_{k-1} F(x_k))^T F(x_k)}{\max\{\mu\|w_{k-1}\|\|F(x_k)\|, \|F(x_{k-1})\|^2\}} \\
&= -\|F(x_k)\|^2 + \frac{\|F(x_k)\|^2 w_{k-1}^T F(x_k) - F(x_k)^T (w_{k-1}^T F(x_k)) F(x_k)}{\max\{\mu\|w_{k-1}\|\|F(x_k)\|, \|F(x_{k-1})\|^2\}} \\
&= -\|F(x_k)\|^2 + \frac{\|F(x_k)\|^2 w_{k-1}^T F(x_k) - \|F(x_k)\|^2 w_{k-1}^T F(x_k)}{\max\{\mu\|w_{k-1}\|\|F(x_k)\|, \|F(x_{k-1})\|^2\}} \\
&= -\|F(x_k)\|^2.
\end{aligned} \tag{7}$$

Using Cauchy–Schwartz inequality, we get

$$\|F(x_k)\| \leq \|d_k\|. \tag{8}$$

Furthermore, since $\max\{\mu\|w_{k-1}\|\|F(x_k)\|, \|F(x_{k-1})\|^2\} \geq \mu\|w_{k-1}\|\|F(x_k)\|$, then,

$$\begin{aligned}
\|d_k\| &= \left\| -F(x_k) + \frac{\|F(x_k)\|^2 w_{k-1} - (F(x_k)^T w_{k-1})F(x_k)}{\max\{\mu\|w_{k-1}\|\|F(x_k)\|, \|F(x_{k-1})\|^2\}} \right\| \\
&\leq \| - F(x_k)\| + \frac{\|\|F(x_k)\|^2 w_{k-1} - (F(x_k)^T w_{k-1})F(x_k)\|}{\max\{\mu\|w_{k-1}\|\|F(x_k)\|, \|F(x_{k-1})\|^2\}} \\
&\leq \|F(x_k)\| + \frac{\|F(x_k)\|^2\|w_{k-1}\|}{\mu\|w_{k-1}\|\|F(x_k)\|} + \frac{\|F(x_k)^T w_{k-1} F(x_k)\|}{\mu\|w_{k-1}\|\|F(x_k)\|} \\
&\leq \|F(x_k)\| + \frac{\|F(x_k)\|^2\|w_{k-1}\|}{\mu\|w_{k-1}\|\|F(x_k)\|} + \frac{\|F(x_k)\|^2\|w_{k-1}\|}{\mu\|w_{k-1}\|\|F(x_k)\|} \\
&= \|F(x_k)\| + \frac{2\|F(x_k)\|}{\mu} \\
&= \left(1 + \frac{2}{\mu}\right)\|F(x_k)\|.
\end{aligned} \tag{9}$$

Combining (8) and (9), we get the desired result. □

Lemma 3. *Suppose that assumptions (C_1)–(C_3) hold and the sequences $\{x_k\}$ and $\{z_k\}$ are generated by Algorithm 1. Then we have*

$$\alpha_k \geq \rho \min\left\{1, \frac{\|F(x_k)\|^2}{(L+\sigma)\|F(x_k + \frac{\alpha_k}{\rho}d_k)\|\|d_k\|^2}\right\}$$

Proof. Suppose $\alpha_k \neq \rho$, then $\frac{\alpha_k}{\rho}$ does not satisfy Equation (4), that is

$$-F\left(x_k + \frac{\alpha_k}{\rho}d_k\right) < \sigma\frac{\alpha_k}{\rho}\|F(x_k + \frac{\alpha_k}{\rho}d_k)\|\|d_k\|^2.$$

This combined with (7) and the fact that F is Lipschitz continuous yields

$$\begin{aligned}
\|F(x_k)\|^2 &= -F(x_k)^T d_k \\
&= \left(F(x_k + \frac{\alpha_k}{\rho}d_k) - F(x_k)\right)^T d_k - F^T\left(x_k + \frac{\alpha_k}{\rho}d_k\right)d_k \\
&\leq L\frac{\alpha_k}{\rho}\|F(x_k + \frac{\alpha_k}{\rho}d_k)\|\|d_k\|^2 + \sigma\frac{\alpha_k}{\rho}\|F(x_k + \frac{\alpha_k}{\rho}d_k)\|\|d_k\|^2 \\
&= \frac{L+\sigma}{\rho}\alpha_k\|F(x_k + \frac{\alpha_k}{\rho}d_k)\|\|d_k\|^2.
\end{aligned} \tag{10}$$

The above equation implies

$$\alpha_k \geq \rho \min \frac{\|F(x_k)\|^2}{(L+\sigma)\|F(x_k + \frac{\alpha_k}{\rho}d_k)\|\|d_k\|^2},$$

which completes the proof. □

Lemma 4. *Suppose that assumptions (C_1)–(C_3) holds, then the sequences $\{x_k\}$ and $\{z_k\}$ generated by Algorithm 1 are bounded. Moreover, we have*

$$\lim_{k\to\infty} \|x_k - z_k\| = 0 \tag{11}$$

and

$$\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0. \tag{12}$$

Proof. We will start by showing that the sequences $\{x_k\}$ and $\{z_k\}$ are bounded. Suppose $\bar{x} \in E'$, then by monotonicity of F, we get

$$\langle F(z_k), x_k - \bar{x}\rangle \geq \langle F(z_k), x_k - z_k\rangle. \tag{13}$$

Also by definition of z_k and the line search (4), we have

$$\langle F(z_k), x_k - z_k\rangle \geq \sigma\alpha_k^2\|F(z_k)\|\|d_k\|^2 \geq 0. \tag{14}$$

So, we have

$$\begin{aligned}
\|x_{k+1} - \bar{x}\|^2 = \|P_E[x_k - \zeta_k F(z_k)] - \bar{x}\|^2 &\leq \|x_k - \zeta_k F(z_k) - \bar{x}\|^2 \\
&= \|x_k - \bar{x}\|^2 - 2\zeta\langle F(z_k), x_k - \bar{x}\rangle + \|\zeta F(z_k)\|^2 \\
&\leq \|x_k - \bar{x}\|^2 - 2\zeta_k\langle F(z_k), x_k - z_k\rangle + \|\zeta F(z_k)\|^2 \\
&= \|x_k - \bar{x}\|^2 - \left(\frac{\langle F(z_k), x_k - z_k\rangle}{\|F(z_k)\|}\right)^2 \\
&\leq \|x_k - \bar{x}\|^2.
\end{aligned} \tag{15}$$

Thus the sequence $\{\|x_k - \bar{x}\|\}$ is non increasing and convergent, and hence $\{x_k\}$ is bounded. Furthermore, from Equation (15), we have

$$\|x_{k+1} - \bar{x}\|^2 \leq \|x_k - \bar{x}\|^2, \tag{16}$$

and we can deduce recursively that

$$\|x_k - \bar{x}\|^2 \leq \|x_0 - \bar{x}\|^2, \quad \forall k \geq 0.$$

Then from assumption (C_3), we obtain

$$\|F(x_k)\| = \|F(x_k) - F(\bar{x})\| \leq L\|x_k - \bar{x}\| \leq L\|x_0 - \bar{x}\|.$$

If we let $L\|x_0 - \bar{x}\| = \kappa$, then the sequence $\{F(x_k)\}$ is bounded, that is,

$$\|F(x_k)\| \leq \kappa, \quad \forall k \geq 0. \tag{17}$$

By the definition of z_k, Equation (14), monotonicity of F and the Cauchy–Schwatz inequality, we get

$$\sigma\|x_k - z_k\| = \frac{\sigma\|\alpha_k d_k\|^2}{\|x_k - z_k\|} \leq \frac{\langle F(z_k), x_k - z_k\rangle}{\|x_k - z_k\|} \leq \frac{\langle F(x_k), x_k - z_k\rangle}{\|x_k - z_k\|} \leq \|F(x_k)\|. \tag{18}$$

The boundedness of the sequence $\{x_k\}$ together with Equations (17) and (18), implies the sequence $\{z_k\}$ is bounded.

Now, as $\{z_k\}$ is bounded, then for any $\bar{x} \in E^{'}$, the sequence $\{z_k - \bar{x}\}$ is also bounded, that is, there exists a positive constant $\nu > 0$ such that

$$\|z_k - \bar{x}\| \leq \nu.$$

This together with assumption (C_3), this yields

$$\|F(z_k)\| = \|F(z_k) - F(\bar{x})\| \leq L\|z_k - \bar{x}\| \leq L\nu.$$

Therefore, using Equation (15), we have

$$\frac{\sigma^2}{(L\nu)^2}\|x_k - z_k\|^4 \leq \|x_k - \bar{x}\|^2 - \|x_{k+1} - \bar{x}\|^2,$$

which implies

$$\frac{\sigma^2}{(L\nu)^2}\sum_{k=0}^{\infty}\|x_k - z_k\|^4 \leq \sum_{k=0}^{\infty}(\|x_k - \bar{x}\|^2 - \|x_{k+1} - \bar{x}\|^2) \leq \|x_0 - \bar{x}\| < \infty. \tag{19}$$

Equation (19) implies

$$\lim_{k\to\infty}\|x_k - z_k\| = 0.$$

However, using statement 2 of Lemma 1, the definition of ζ_k and the Cauchy-Schwartz inequality, we have

$$\begin{aligned} \|x_{k+1} - x_k\| &= \|P_E[x_k - \zeta_k F(z_k)] - x_k\| \\ &\leq \|x_k - \zeta_k F(z_k) - x_k\| \\ &= \|\zeta_k F(z_k)\| \\ &= \|x_k - z_k\|, \end{aligned} \tag{20}$$

which yields

$$\lim_{k\to\infty}\|x_{k+1} - x_k\| = 0.$$

□

Remark 2. *By Equation (11) and definition of z_k, then*

$$\lim_{k\to\infty}\alpha_k\|d_k\| = 0. \tag{21}$$

Theorem 1. *Suppose that assumption (C_1)–(C_3) holds and let the sequence $\{x_k\}$ be generated by Algorithm 1, then*

$$\liminf_{k\to\infty}\|F(x_k)\| = 0. \tag{22}$$

Proof. Assume that Equation (22) is not true, then there exists a constant $\epsilon > 0$ such that

$$\|F(x_k)\| \geq \epsilon, \quad \forall k \geq 0. \tag{23}$$

Combining (8) and (23), we have

$$\|d_k\| \geq \|F(x_k)\| \geq \epsilon, \quad \forall k \geq 0.$$

As $w_k = x_k + \alpha_k d_k$ and $\lim_{k\to\infty} \|x_k - z_k\| = 0$, we get $\lim_{k\to\infty} \alpha_k \|d_k\| = 0$ and

$$\lim_{k\to\infty} \alpha_k = 0. \tag{24}$$

On the other side, if $M = \left(1 + \frac{2}{\mu}\right)\kappa$, Lemma 3 and Equation (9) implies $\alpha_k \|d_k\| \geq \rho \frac{\epsilon^2}{(L+\sigma)MLv}$, which contradicts with (24). Therefore, (22) must hold. $\square$

4. Numerical Experiments

To test the performance of the proposed method, we compare it with accelerated conjugate gradient descent (ACGD) and projected Dai-Yuan (PDY) methods in [27,28], respectively. In addition, MFRM method is applied to solve signal and image recovery problems arising in compressive sensing. All codes were written in MATLAB R2018b and run on a PC with intel COREi5 processor with 4GB of RAM and CPU 2.3GHZ. All runs were stopped whenever $\|F(x_k)\| < 10^{-5}$. The parameters chosen for each method are as follows:

MFRM method: $\gamma = 1, \rho = 0.9, \mu = 0.01, \sigma = 0.0001$.
ACGD method: all parameters are chosen as in [27].
PDY method: all parameters are chosen as in [28].

We tested eight problems with dimensions of $n = 1000, 5000, 10{,}000, 50{,}000, 100{,}000$ and 6 initial points: $x_1 = (0.1, 0.1, \cdots, 1)^T$, $x_2 = (0.2, 0.2, \cdots, 0.2)^T$, $x_3 = (0.5, 0.5, \cdots, 0.5)^T$, $x_4 = (1.2, 1.2, \cdots, 1.2)^T$, $x_5 = (1.5, 1.5, \cdots, 1.5)^T$, $x_6 = (2, 2, \cdots, 2)^T$. In Tables 1–8, the number of Iterations (Iter), number of function evaluations (Fval), CPU time in seconds (time) and the norm at the approximate solution (NORM) were reported. The symbol '$-$' is used when the number of Iterations exceeds 1000 and/or the number of function evaluations exceeds 2000.

The test problems are listed below, where the function F is taken as $F(x) = (f_1(x), f_2(x), \dots, f_n(x))^T$.

Problem 1 [38] Exponential Function.

$$\begin{aligned} &f_1(x) = e^{x_1} - 1, \\ &f_i(x) = e^{x_i} + x_i - 1, \text{ for } i = 2, 3, ..., n, \\ &\text{and } E = \mathbb{R}^n_+. \end{aligned}$$

Problem 2 [38] Modified Logarithmic Function.

$$\begin{aligned} &f_i(x) = \ln(x_i + 1) - \frac{x_i}{n}, \text{ for } i = 2, 3, ..., n, \\ &\text{and } E = \{x \in \mathbb{R}^n : \sum_{i=1}^n x_i \leq n, x_i > -1, i = 1, 2, \dots, n\}. \end{aligned}$$

Problem 3 [6] Nonsmooth Function.

$$\begin{aligned} &f_i(x) = 2x_i - \sin|x_i|, \ i = 1, 2, 3, ..., n, \\ &\text{and } E = \{x \in \mathbb{R}^n : \sum_{i=1}^n x_i \leq n, x_i \geq 0, i = 1, 2, \dots, n\}. \end{aligned}$$

It is clear that problem 3 is nonsmooth at $x = 0$.

Problem 4 [38] Strictly Convex Function I.

$$f_i(x) = e^{x_i} - 1, \text{ for } i = 1, 2, ..., n,$$
$$\text{and } E = \mathbb{R}^n_+.$$

Problem 5 [38] Strictly Convex Function II.

$$f_i(x) = \frac{i}{n} e^{x_i} - 1, \text{ for } i = 1, 2, ..., n,$$
$$\text{and } E = \mathbb{R}^n_+.$$

Problem 6 [39] Tridiagonal Exponential Function

$$\begin{aligned} f_1(x) &= x_1 - e^{\cos(h(x_1 + x_2))}, \\ f_i(x) &= x_i - e^{\cos(h(x_{i-1} + x_i + x_{i+1}))}, \text{ for } i = 2, ..., n-1, \\ f_n(x) &= x_n - e^{\cos(h(x_{n-1} + x_n))}, \\ h &= \frac{1}{n+1} \text{ and } E = \mathbb{R}^n_+. \end{aligned}$$

Problem 7 [40] Nonsmooth Function

$$f_i(x) = x_i - \sin|x_i - 1|, \ i = 1, 2, 3, ..., n.$$
$$\text{and } E = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i \leq n, x_i \geq -1, i = 1, 2, \ldots, n\}.$$

Problem 8 [27] Penalty 1

$$\begin{aligned} t_i &= \sum_{i=1}^{n} x_i^2, \ \ c = 10^{-5} \\ f_i(x) &= 2c(x_i - 1) + 4(t_i - 0.25)x_i, \ i = 1, 2, 3, ..., n. \\ &\text{and } E = \mathbb{R}^n_+. \end{aligned}$$

To show in detail the efficiency and robustness of all methods, we employ the performance profile developed in [41], which is a helpful process of standardizing the comparison of methods. Suppose that we have n_s solvers and n_l problems and we are interested in using either number of Iterations, CPU time or number of function evaluations as our measure of performance; so we let $k_{l,s}$ to be the number of iterations, CPU time or number of function evaluations required to solve problem by solver s. To compare the performance on problem l by a solver s with the best performance by any other solver on this problem, we use the performance ratio $r_{l,s}$ defined as

$$r_{l,s} = \frac{k_{l,s}}{\min\{k_{l,s} : s \in S\}},$$

where S is the set of solvers.

The overall performance of the solver is obtained using the (cumulative) distribution function for the performance ratio P. So if we let

$$P(t) = \frac{1}{n_l} size\{l \in L : r_{l,s} \leq t\},$$

then $P(t)$ is the probability for solver $s \in S$ that a performance ratio $r_{l,s}$ is within a factor $t \in R$ of the best possible ratio. If the set of problems L is large enough, then the solvers with the large probability $P(t)$ are considered as the best.

Table 1. Numerical results for modified Fletcher–Reeves (MFRM), accelerated conjugate gradient descent (ACGD) and projected Dai-Yuan (PDY) for problem 1 with given initial points and dimensions.

		MFRM				ACGD				PDY			
Dimension	**Initial Point**	**Iter**	**Fval**	**Time**	**Norm**	**Iter**	**Fval**	**Time**	**Norm**	**Iter**	**Fval**	**Time**	**Norm**
1000	x_1	23	98	0.42639	9.01×10^{-6}	8	34	0.21556	9.26×10^{-6}	12	49	0.19349	9.18×10^{-6}
	x_2	7	35	0.019885	8.82×10^{-6}	9	39	0.086582	3.01×10^{-6}	13	53	0.07318	6.35×10^{-6}
	x_3	8	40	0.011238	9.74×10^{-6}	9	38	0.034359	4.02×10^{-6}	14	57	0.01405	5.59×10^{-6}
	x_4	15	70	0.066659	6.01×10^{-6}	16	67	0.017188	9.22×10^{-6}	15	61	0.01421	4.07×10^{-6}
	x_5	5	31	0.16103	0	18	75	0.11646	4.46×10^{-6}	14	57	0.08690	9.91×10^{-6}
	x_6	31	134	0.03232	7.65×10^{-6}	25	104	0.042967	6.74×10^{-6}	40	162	0.04060	9.70×10^{-6}
5000	x_1	8	38	0.053865	5.63×10^{-6}	9	38	0.023729	3.89×10^{-6}	13	53	0.02775	6.87×10^{-6}
	x_2	8	40	0.036653	2.59×10^{-6}	9	38	0.021951	6.65×10^{-6}	14	57	0.02974	4.62×10^{-6}
	x_3	8	40	0.030089	6.41×10^{-6}	9	39	0.019317	8.01×10^{-6}	15	61	0.04353	4.18×10^{-6}
	x_4	16	74	0.081741	4.71×10^{-6}	17	71	0.05235	8.12×10^{-6}	15	61	0.03288	9.08×10^{-6}
	x_5	5	31	0.030748	0	18	75	0.038894	8.14×10^{-6}	15	61	0.03556	7.30×10^{-6}
	x_6	31	134	0.087531	8.1×10^{-6}	26	108	0.053473	7.96×10^{-6}	39	158	0.10419	9.86×10^{-6}
10,000	x_1	5	26	0.03829	3.7×10^{-6}	9	39	0.044961	5.5×10^{-6}	13	53	0.05544	9.70×10^{-6}
	x_2	8	40	0.055099	3.64×10^{-6}	9	39	0.0358	9.39×10^{-6}	14	57	0.06201	6.53×10^{-6}
	x_3	8	40	0.049974	5.44×10^{-6}	10	43	0.04176	2.12×10^{-6}	15	61	0.08704	5.90×10^{-6}
	x_4	16	74	0.125	6.61×10^{-6}	18	75	0.066316	4.58×10^{-6}	16	65	0.07797	4.28×10^{-6}
	x_5	5	31	0.048751	0	18	75	0.11807	7.86×10^{-6}	39	158	0.20751	7.97×10^{-6}
	x_6	28	122	0.13649	7.18×10^{-6}	27	112	0.10593	6.22×10^{-6}	87	351	0.36678	9.93×10^{-6}
50,000	x_1	5	26	0.1584	3.58×10^{-6}	10	43	0.15918	2.33×10^{-6}	14	57	0.23129	7.12×10^{-6}
	x_2	8	40	0.18044	8.1×10^{-6}	10	43	0.16252	3.97×10^{-6}	15	61	0.23975	4.91×10^{-6}
	x_3	8	40	0.186	4.54×10^{-6}	10	43	0.15707	4.67×10^{-6}	16	65	0.24735	4.37×10^{-6}
	x_4	17	78	0.31567	5.47×10^{-6}	19	79	0.27474	4.1×10^{-6}	38	154	0.55277	7.54×10^{-6}
	x_5	5	31	0.18586	0	18	75	0.27118	5.06×10^{-6}	177	712	2.29950	9.44×10^{-6}
	x_6	20	90	0.39237	6.44×10^{-6}	28	116	0.35197	7.69×10^{-6}	361	1449	4.63780	9.74×10^{-6}
100,000	x_1	5	26	0.26116	4.59×10^{-6}	10	42	0.28038	3.29×10^{-6}	15	61	0.50090	3.39×10^{-6}
	x_2	9	43	0.35288	1.59×10^{-6}	10	42	0.28999	5.62×10^{-6}	15	61	0.45876	6.94×10^{-6}
	x_3	8	40	0.35809	4.96×10^{-6}	10	42	0.29255	6.59×10^{-6}	16	65	0.51380	6.18×10^{-6}
	x_4	17	78	0.59347	7.73×10^{-6}	19	79	0.51261	5.79×10^{-6}	175	704	4.48920	9.47×10^{-6}
	x_5	32	138	0.98463	7.09×10^{-6}	18	75	0.46086	4.05×10^{-6}	176	708	4.49410	9.91×10^{-6}
	x_6	17	78	0.57701	9.31×10^{-6}	29	120	0.71678	6.05×10^{-6}	360	1445	9.10170	9.99×10^{-6}

Table 2. Numerical results for MFRM, ACGD and PDY for problem 2 with given initial points and dimensions.

		MFRM				ACGD				PDY			
Dimension	**Initial Point**	**Iter**	**Fval**	**Time**	**Norm**	**Iter**	**Fval**	**Time**	**Norm**	**Iter**	**Fval**	**Time**	**Norm**
1000	x_1	3	8	0.007092	5.17×10^{-7}	3	8	0.036061	5.17×10^{-7}	10	39	0.01053	6.96×10^{-6}
	x_2	3	8	0.012401	6.04×10^{-6}	3	8	0.006143	6.04×10^{-6}	11	43	0.00937	9.23×10^{-6}
	x_3	4	11	0.003993	4.37×10^{-7}	4	11	0.006476	4.37×10^{-7}	13	51	0.01111	6.26×10^{-6}
	x_4	5	14	0.010363	1.52×10^{-7}	5	14	0.005968	1.52×10^{-7}	14	55	0.02154	9.46×10^{-6}
	x_5	5	14	0.007234	1.1×10^{-6}	5	14	0.02349	1.1×10^{-6}	15	59	0.01850	4.60×10^{-6}
	x_6	6	17	0.006496	1.74×10^{-8}	6	17	0.00677	1.74×10^{-8}	15	59	0.01938	7.71×10^{-6}
5000	x_1	3	8	0.011561	1.75×10^{-7}	3	8	0.009794	1.75×10^{-7}	11	43	0.03528	4.86×10^{-6}
	x_2	3	8	0.010452	3.13×10^{-6}	3	8	0.009591	3.13×10^{-6}	12	47	0.04032	6.89×10^{-6}
	x_3	4	11	0.01516	1.42×10^{-7}	4	11	0.013767	1.42×10^{-7}	14	55	0.04889	4.61×10^{-6}
	x_4	5	14	0.019733	3.94×10^{-8}	5	14	0.014274	3.94×10^{-8}	15	59	0.04826	6.96×10^{-6}
	x_5	5	14	0.018462	4.05×10^{-7}	5	14	0.011728	4.05×10^{-7}	16	63	0.05969	3.37×10^{-6}
	x_6	6	17	0.028536	2.36×10^{-9}	6	17	0.016345	2.36×10^{-9}	16	63	0.06253	5.64×10^{-6}
10,000	x_1	3	8	0.019053	1.21×10^{-7}	3	8	0.0135	1.21×10^{-7}	11	43	0.06732	6.85×10^{-6}
	x_2	3	8	0.01791	2.79×10^{-6}	3	8	0.015807	2.79×10^{-6}	12	47	0.12232	9.72×10^{-6}
	x_3	4	11	0.033042	9.73×10^{-8}	4	11	0.020752	9.73×10^{-8}	14	55	0.08288	6.51×10^{-6}
	x_4	5	14	0.031576	2.56×10^{-8}	5	14	0.04483	2.56×10^{-8}	15	59	0.08413	9.82×10^{-6}
	x_5	5	14	0.032747	2.93×10^{-7}	5	14	0.026975	2.93×10^{-7}	16	63	0.09589	4.75×10^{-6}
	x_6	6	17	0.036002	1.24×10^{-9}	6	17	0.032445	1.24×10^{-9}	16	64	0.11499	8.55×10^{-6}
50,000	x_1	3	8	0.0737	6.32×10^{-8}	7	26	0.16925	2.94×10^{-6}	12	47	0.27826	5.23×10^{-6}
	x_2	3	8	0.06964	3.37×10^{-6}	9	34	0.18801	2.78×10^{-6}	13	51	0.29642	7.11×10^{-6}
	x_3	4	11	0.093027	4.87×10^{-8}	7	25	0.15375	9.11×10^{-6}	15	59	0.35602	4.82×10^{-6}
	x_4	5	14	0.11219	1.11×10^{-8}	7	24	0.15382	9.18×10^{-6}	35	141	0.69470	6.69×10^{-6}
	x_5	5	14	0.1173	1.84×10^{-7}	9	32	0.18164	6.71×10^{-6}	35	141	0.68488	9.12×10^{-6}
	x_6	6	17	0.13794	4.01×10^{-10}	6	19	0.11216	5.2×10^{-6}	35	141	0.70973	9.91×10^{-6}
100,000	x_1	3	8	0.13021	5.4×10^{-8}	7	26	0.2609	4.14×10^{-6}	12	47	0.44541	7.39×10^{-6}
	x_2	3	8	0.13267	4.27×10^{-6}	9	34	0.32666	3.93×10^{-6}	14	55	0.53299	3.39×10^{-6}
	x_3	4	11	0.17338	4.05×10^{-8}	8	29	0.3113	3.33×10^{-6}	15	60	0.58603	8.71×10^{-6}
	x_4	5	14	0.20036	8.15×10^{-9}	8	28	0.2997	3.34×10^{-6}	72	290	2.70630	8.31×10^{-6}
	x_5	5	14	0.25274	1.8×10^{-7}	9	32	0.32098	9.46×10^{-6}	72	290	2.72220	8.68×10^{-6}
	x_6	6	17	0.24952	2.71×10^{-10}	6	19	0.21972	7.01×10^{-6}	72	290	2.75850	8.96×10^{-6}

Table 3. Numerical results for MFRM, ACGD and PDY for problem 3 with given initial points and dimensions.

		MFRM				ACGD				PDY			
Dimension	Initial Point	Iter	Fval	Time	Norm	Iter	Fval	Time	Norm	Iter	Fval	Time	Norm
1000	x_1	6	24	0.024062	3.11×10^{-6}	6	40	0.02951	4.44×10^{-6}	12	48	0.01255	4.45×10^{-6}
	x_2	6	24	0.005345	5.94×10^{-6}	6	40	0.0077681	8.75×10^{-6}	12	48	0.01311	9.02×10^{-6}
	x_3	6	24	0.006109	9.94×10^{-6}	6	44	0.0067049	5.09×10^{-6}	13	52	0.01486	8.34×10^{-6}
	x_4	8	33	0.006127	3.1×10^{-6}	8	44	0.007142	5.04×10^{-6}	14	56	0.01698	8.04×10^{-6}
	x_5	11	46	0.010427	2.71×10^{-6}	11	40	0.010411	3.12×10^{-6}	14	56	0.01551	9.72×10^{-6}
	x_6	16	68	0.010682	8.38×10^{-6}	16	77	0.014759	5.98×10^{-6}	14	56	0.01534	9.42×10^{-6}
5000	x_1	6	24	0.020455	6.96×10^{-6}	6	40	0.020368	9.93×10^{-6}	12	48	0.03660	9.94×10^{-6}
	x_2	7	28	0.021552	1.33×10^{-6}	7	44	0.029622	5.09×10^{-6}	13	52	0.03616	6.85×10^{-6}
	x_3	7	28	0.023056	2.22×10^{-6}	7	48	0.030044	2.96×10^{-6}	14	56	0.04594	6.14×10^{-6}
	x_4	8	33	0.022984	6.92×10^{-6}	8	48	0.022777	2.93×10^{-6}	15	60	0.04342	6.01×10^{-6}
	x_5	11	46	0.031466	6.06×10^{-6}	11	40	0.019226	6.97×10^{-6}	15	60	0.04296	7.25×10^{-6}
	x_6	17	72	0.049308	7.67×10^{-6}	17	81	0.036095	6.05×10^{-6}	32	129	0.10081	8.85×10^{-6}
10,000	x_1	6	24	0.03064	9.85×10^{-6}	6	44	0.03997	3.65×10^{-6}	13	52	0.06192	4.77×10^{-6}
	x_2	7	28	0.035806	1.88×10^{-6}	7	44	0.037221	7.19×10^{-6}	13	52	0.06442	9.68×10^{-6}
	x_3	7	28	0.035795	3.14×10^{-6}	7	48	0.053226	4.18×10^{-6}	14	56	0.09499	8.69×10^{-6}
	x_4	8	33	0.041017	9.79×10^{-6}	8	48	0.057984	4.15×10^{-6}	15	60	0.07696	8.5×10^{-6}
	x_5	11	46	0.06448	8.58×10^{-6}	11	40	0.047413	9.85×10^{-6}	33	133	0.18625	6.45×10^{-6}
	x_6	18	76	0.09651	4.44×10^{-6}	18	81	0.085238	8.56×10^{-6}	33	133	0.15548	7.51×10^{-6}
50,000	x_1	7	28	0.14323	2.2×10^{-6}	7	44	0.17175	8.17×10^{-6}	14	56	0.23642	3.51×10^{-6}
	x_2	7	28	0.13625	4.2×10^{-6}	7	48	0.18484	4.18×10^{-6}	14	56	0.24813	7.12×10^{-6}
	x_3	7	28	0.13246	7.03×10^{-6}	7	48	0.1827	9.36×10^{-6}	15	60	0.27049	6.53×10^{-6}
	x_4	9	37	0.18261	4.16×10^{-6}	9	48	0.18993	9.27×10^{-6}	34	137	0.54545	7.13×10^{-6}
	x_5	12	50	0.21743	5.2×10^{-6}	12	44	0.17043	5.73×10^{-6}	68	274	1.02330	9.99×10^{-6}
	x_6	18	76	0.34645	9.93×10^{-6}	18	85	0.32938	8.66×10^{-6}	69	278	1.03810	8.05×10^{-6}
100,000	x_1	7	28	0.27078	3.11×10^{-6}	7	48	0.36144	3×10^{-6}	14	56	0.45475	4.96×10^{-6}
	x_2	7	28	0.26974	5.94×10^{-6}	7	48	0.37515	5.91×10^{-6}	15	60	0.49018	3.39×10^{-6}
	x_3	7	28	0.25475	9.94×10^{-6}	7	52	0.39071	3.44×10^{-6}	15	60	0.49016	9.24×10^{-6}
	x_4	9	37	0.3089	5.88×10^{-6}	9	52	0.35961	3.41×10^{-6}	139	559	4.03110	9.01×10^{-6}
	x_5	12	50	0.41839	7.35×10^{-6}	12	44	0.33105	8.1×10^{-6}	70	282	2.07100	8.54×10^{-6}
	x_6	19	80	0.64773	5.75×10^{-6}	19	89	0.61329	5.54×10^{-6}	139	559	4.02440	9.38×10^{-6}

Table 4. Numerical results for MFRM, ACGD and PDY for problem 4 with given initial points and dimensions.

		MFRM				ACGD				PDY			
Dimension	Initial Point	Iter	Fval	Time	Norm	Iter	Fval	Time	Norm	Iter	Fval	Time	Norm
1000	x_1	6	24	0.00855	1.65×10^{-6}	10	40	0.014662	3.65×10^{-6}	12	48	0.00989	4.60×10^{-6}
	x_2	5	20	0.004234	2.32×10^{-6}	10	40	0.0064115	5.79×10^{-6}	12	48	0.00966	9.57×10^{-6}
	x_3	10	42	0.007426	6.42×10^{-6}	10	40	0.0054818	3.29×10^{-6}	13	52	0.00887	8.49×10^{-6}
	x_4	21	90	0.011603	5.84×10^{-6}	27	110	0.012854	8.97×10^{-6}	12	48	0.01207	5.83×10^{-6}
	x_5	16	71	0.010735	8.48×10^{-6}	26	106	0.015603	5.97×10^{-6}	29	117	0.05371	9.43×10^{-6}
	x_6	1	15	0.005932	0	36	147	0.025039	9.56×10^{-6}	29	117	0.02396	6.65×10^{-6}
5000	x_1	6	24	0.019995	3.68×10^{-6}	10	40	0.018283	8.15×10^{-6}	13	52	0.02503	3.49×10^{-6}
	x_2	5	20	0.00934	5.2×10^{-6}	11	44	0.016733	3.36×10^{-6}	13	52	0.02626	7.24×10^{-6}
	x_3	11	46	0.02156	3.89×10^{-6}	10	40	0.017073	7.37×10^{-6}	14	56	0.03349	6.29×10^{-6}
	x_4	22	94	0.043325	6.81×10^{-6}	29	118	0.047436	7.09×10^{-6}	13	52	0.02258	4.25×10^{-6}
	x_5	18	79	0.096692	6.15×10^{-6}	27	110	0.058405	7.95×10^{-6}	31	125	0.05471	7.59×10^{-6}
	x_6	1	15	0.012199	0	39	159	0.059448	7.33×10^{-6}	63	254	0.10064	8.54×10^{-6}
10,000	x_1	6	24	0.019264	5.2×10^{-6}	11	44	0.026877	3×10^{-6}	13	52	0.03761	4.93×10^{-6}
	x_2	5	20	0.017891	7.35×10^{-6}	11	44	0.03118	4.76×10^{-6}	14	56	0.04100	3.37×10^{-6}
	x_3	11	46	0.036079	5.5×10^{-6}	11	44	0.034673	2.71×10^{-6}	14	56	0.03919	8.90×10^{-6}
	x_4	22	94	0.069778	9.63×10^{-6}	30	122	0.069971	5.97×10^{-6}	32	129	0.09613	6.02×10^{-6}
	x_5	18	79	0.062821	8.69×10^{-6}	28	114	0.066866	6.68×10^{-6}	32	129	0.09177	6.44×10^{-6}
	x_6	1	15	0.017237	0	40	163	0.093749	7.26×10^{-6}	64	258	0.20791	9.39×10^{-6}
50,000	x_1	7	28	0.093473	1.16×10^{-6}	11	44	0.16749	6.7×10^{-6}	14	56	0.17193	3.63×10^{-6}
	x_2	6	24	0.072206	1.64×10^{-6}	12	48	0.11391	2.77×10^{-6}	14	56	0.15237	7.54×10^{-6}
	x_3	12	50	0.14285	3.33×10^{-6}	11	44	0.11036	6.06×10^{-6}	15	60	0.16549	6.66×10^{-6}
	x_4	24	102	0.30313	5.86×10^{-6}	31	126	0.30903	7.94×10^{-6}	67	270	0.76283	7.81×10^{-6}
	x_5	20	87	0.28955	6.31×10^{-6}	29	118	0.30266	8.89×10^{-6}	67	270	0.76157	8.80×10^{-6}
	x_6	1	15	0.061327	0	42	171	0.41158	7.96×10^{-6}	269	1080	2.92510	9.41×10^{-6}
100,000	x_1	7	28	0.15038	1.65×10^{-6}	11	44	0.2434	9.48×10^{-6}	14	56	0.30229	5.13×10^{-6}
	x_2	6	24	0.13126	2.32×10^{-6}	12	48	0.2614	3.91×10^{-6}	15	60	0.31648	3.59×10^{-6}
	x_3	12	50	0.31585	4.71×10^{-6}	11	44	0.2161	8.57×10^{-6}	32	129	0.72838	9.99×10^{-6}
	x_4	24	102	0.58023	8.29×10^{-6}	32	130	0.65289	6.68×10^{-6}	135	543	2.86780	9.73×10^{-6}
	x_5	20	87	0.5122	8.92×10^{-6}	30	122	0.61637	7.48×10^{-6}	272	1092	5.74140	9.91×10^{-6}
	x_6	1	15	0.11696	0	43	175	0.82759	7.88×10^{-6}	548	2197	11.44130	9.87×10^{-6}

Table 5. Numerical results for MFRM, ACGD and PDY for problem 5 with given initial points and dimensions.

		MFRM				ACGD				PDY			
Dimension	**Initial Point**	**Iter**	**Fval**	**Time**	**Norm**	**Iter**	**Fval**	**Time**	**Norm**	**Iter**	**Fval**	**Time**	**Norm**
1000	x_1	26	98	0.023555	3.51×10^{-6}	39	154	0.022285	9.7×10^{-6}	16	63	0.07575	6.03×10^{-6}
	x_2	40	154	0.024539	5.9×10^{-6}	22	85	0.015671	5.03×10^{-6}	16	63	0.01470	5.42×10^{-6}
	x_3	37	144	0.021659	7.11×10^{-6}	43	173	0.029569	7.96×10^{-6}	33	132	0.02208	6.75×10^{-6}
	x_4	49	206	0.030696	9.52×10^{-6}	30	122	0.014942	6.05×10^{-6}	30	121	0.01835	8.39×10^{-6}
	x_5	46	194	0.11589	7.06×10^{-6}	29	118	0.040406	6.5×10^{-6}	32	129	0.02700	8.47×10^{-6}
	x_6	43	182	0.027471	8.7×10^{-6}	40	163	0.0311	9.83×10^{-6}	30	121	0.01712	6.95×10^{-6}
5000	x_1	38	147	0.073315	4.96×10^{-6}	30	117	0.060877	9.56×10^{-6}	17	67	0.04394	5.64×10^{-6}
	x_2	20	77	0.056225	4.98×10^{-6}	16	60	0.027911	5.91×10^{-6}	17	67	0.04635	5.07×10^{-6}
	x_3	41	157	0.082151	8.92×10^{-6}	78	315	0.12774	9.7×10^{-6}	35	140	0.08311	9.74×10^{-6}
	x_4	48	202	0.10166	9.19×10^{-6}	31	126	0.067911	8.39×10^{-6}	33	133	0.08075	6.02×10^{-6}
	x_6	147	562	3.308158	8.44×10^{-7}	31	126	0.067856	7.81×10^{-6}	35	141	0.10091	7.51×10^{-6}
	x_7	45	190	0.090276	7.14×10^{-6}	44	179	0.09371	7.37×10^{-6}	32	129	0.08054	8.55×10^{-6}
10,000	x_1	37	143	0.12665	9.28×10^{-6}	77	308	0.28678	9.85×10^{-6}	17	67	0.06816	8.81×10^{-6}
	x_2	22	84	0.077288	9.78×10^{-6}	16	60	0.071657	7.52×10^{-6}	17	67	0.08833	7.80×10^{-6}
	x_3	39	149	0.1297	6.74×10^{-6}	105	424	0.34212	9.08×10^{-6}	37	148	0.14732	6.36×10^{-6}
	x_4	60	250	0.2175	7.56×10^{-6}	32	130	0.11937	7.17×10^{-6}	37	149	0.14293	8.25×10^{-6}
	x_5	44	186	0.1727	7.68×10^{-6}	32	130	0.11921	8.26×10^{-6}	36	145	0.14719	8.23×10^{-6}
	x_6	46	194	0.1728	8.62×10^{-6}	45	183	0.15634	9.01×10^{-6}	74	298	0.26456	7.79×10^{-6}
50,000	x_1	44	170	0.62202	1×10^{-5}	90	539	31.75299	2.56×10^{-7}	42	169	0.58113	7.78×10^{-6}
	x_2	69	280	0.9662	6.87×10^{-6}	31	122	0.33817	7.09×10^{-6}	42	169	0.58456	7.13×10^{-6}
	x_3	119	464	25.87657	9.34×10^{-7}	260	1047	2.8824	9.67×10^{-6}	41	165	0.58717	8.87×10^{-6}
	x_4	50	210	0.71599	8.38×10^{-6}	33	134	0.39039	9.98×10^{-6}	40	161	0.56431	7.17×10^{-6}
	x_5	46	194	0.65538	8.47×10^{-6}	35	142	0.40807	7.19×10^{-6}	82	330	1.08920	8.44×10^{-6}
	x_6	50	210	0.69117	8.12×10^{-6}	49	199	0.57702	8.97×10^{-6}	80	322	1.06670	7.82×10^{-6}
100,000	x_1	31	121	0.84183	4.48×10^{-6}	88	530	61.97806	5.53×10^{-7}	43	173	1.09620	8.47×10^{-6}
	x_2	135	518	59.19294	8.37×10^{-7}	110	442	2.2661	9.55×10^{-6}	43	173	1.10040	7.77×10^{-6}
	x_3	46	178	1.1322	6.99×10^{-6}	345	1388	7.1938	9.76×10^{-6}	42	169	1.08330	9.66×10^{-6}
	x_4	50	210	1.3737	8.85×10^{-6}	34	138	0.74362	8.65×10^{-6}	85	342	2.11880	9.22×10^{-6}
	x_5	47	198	1.3879	8.31×10^{-6}	36	146	0.79012	8.09×10^{-6}	84	338	2.10640	9.78×10^{-6}
	x_6	52	218	1.4318	7.37×10^{-6}	51	207	1.1601	8.42×10^{-6}	167	671	4.06200	9.90×10^{-6}

Table 6. Numerical Results for MFRM, ACGD and PDY for problem 6 with given initial points and dimensions.

		MFRM				ACGD				PDY			
Dimension	**Initial Point**	**Iter**	**Fval**	**Time**	**Norm**	**Iter**	**Fval**	**Time**	**Norm**	**Iter**	**Fval**	**Time**	**Norm**
1000	x_1	11	44	0.011156	8.32×10^{-6}	12	48	0.02786	7.88×10^{-6}	15	60	0.01671	4.35×10^{-6}
	x_2	11	44	0.016092	7.32×10^{-6}	12	48	0.01042	7.58×10^{-6}	15	60	0.01346	4.18×10^{-6}
	x_3	11	44	0.010446	8.83×10^{-6}	12	48	0.0092	6.68×10^{-6}	15	60	0.01630	3.68×10^{-6}
	x_4	10	40	0.011233	7.38×10^{-6}	12	48	0.013617	4.57×10^{-6}	14	56	0.01339	7.48×10^{-6}
	x_5	9	36	0.011325	8.29×10^{-6}	12	48	0.011492	3.67×10^{-6}	14	56	0.01267	6.01×10^{-6}
	x_6	7	28	0.009452	8.25×10^{-6}	11	44	0.016351	8.32×10^{-6}	14	56	0.01685	3.54×10^{-6}
5000	x_1	8	32	0.026924	1.87×10^{-6}	13	52	0.036025	4.59×10^{-6}	15	60	0.05038	9.73×10^{-6}
	x_2	8	32	0.043488	1.8×10^{-6}	13	52	0.040897	4.42×10^{-6}	15	60	0.04775	9.36×10^{-6}
	x_3	8	32	0.02709	1.59×10^{-6}	13	52	0.039937	3.89×10^{-6}	15	60	0.04923	8.25×10^{-6}
	x_4	8	32	0.026351	1.1×10^{-6}	13	52	0.033013	2.66×10^{-6}	15	60	0.05793	5.64×10^{-6}
	x_5	7	28	0.023442	8.62×10^{-6}	12	48	0.030462	8.22×10^{-6}	15	60	0.04597	4.53×10^{-6}
	x_6	7	28	0.022952	5.08×10^{-6}	12	48	0.028786	4.85×10^{-6}	14	56	0.05070	7.93×10^{-6}
10,000	x_1	8	32	0.061374	2.62×10^{-6}	13	52	0.092372	6.5×10^{-6}	68	274	0.40724	9.06×10^{-6}
	x_2	8	32	0.06285	2.52×10^{-6}	13	52	0.059778	6.25×10^{-6}	68	274	0.41818	8.72×10^{-6}
	x_3	8	32	0.059913	2.22×10^{-6}	13	52	0.077326	5.5×10^{-6}	34	137	0.21905	6.22×10^{-6}
	x_4	8	32	0.057003	1.52×10^{-6}	13	52	0.087745	3.77×10^{-6}	15	60	0.10076	7.98×10^{-6}
	x_5	8	32	0.070377	1.22×10^{-6}	13	52	0.077217	3.02×10^{-6}	15	60	0.12680	6.40×10^{-6}
	x_6	7	28	0.052718	7.18×10^{-6}	12	48	0.067375	6.85×10^{-6}	15	60	0.11984	3.78×10^{-6}
50,000	x_1	8	32	0.21258	5.85×10^{-6}	14	56	0.32965	3.78×10^{-6}	143	575	3.09120	9.42×10^{-6}
	x_2	8	32	0.21203	5.63×10^{-6}	14	56	0.31297	3.63×10^{-6}	143	575	3.06200	9.06×10^{-6}
	x_3	8	32	0.20885	4.96×10^{-6}	14	56	0.30089	3.2×10^{-6}	142	571	3.04950	9.04×10^{-6}
	x_4	8	32	0.20483	3.4×10^{-6}	13	52	0.26855	8.42×10^{-6}	69	278	1.53920	9.14×10^{-6}
	x_5	8	32	0.21467	2.72×10^{-6}	13	52	0.26304	6.76×10^{-6}	68	274	1.49490	9.43×10^{-6}
	x_6	8	32	0.20933	1.61×10^{-6}	13	52	0.26143	3.99×10^{-6}	15	60	0.38177	8.44×10^{-6}
100,000	x_1	8	32	0.41701	8.28×10^{-6}	14	56	0.58853	5.34×10^{-6}	292	1172	13.59530	9.53×10^{-6}
	x_2	8	32	0.41511	7.96×10^{-6}	14	56	0.58897	5.14×10^{-6}	290	1164	13.30930	9.75×10^{-6}
	x_3	8	32	0.44061	7.01×10^{-6}	14	56	0.57318	4.53×10^{-6}	144	579	6.68150	9.96×10^{-6}
	x_4	8	32	0.43805	4.8×10^{-6}	14	56	0.58712	3.1×10^{-6}	141	567	6.50800	9.92×10^{-6}
	x_5	8	32	0.41147	3.85×10^{-6}	13	52	0.56384	9.56×10^{-6}	70	282	3.30510	8.07×10^{-6}
	x_6	8	32	0.43925	2.27×10^{-6}	13	52	0.53343	5.64×10^{-6}	34	137	1.64510	6.37×10^{-6}

Table 7. Numerical Results for MFRM, ACGD and PDY for problem 7 with given initial points and dimensions.

		MFRM				ACGD				PDY			
Dimension	**Initial Point**	**Iter**	**Fval**	**Time**	**Norm**	**Iter**	**Fval**	**Time**	**Norm**	**Iter**	**Fval**	**Time**	**Norm**
1000	x_1	4	21	0.011834	3.24×10^{-7}	10	42	0.008528	2.46×10^{-6}	14	57	0.00953	5.28×10^{-6}
	x_2	4	21	0.006228	1.43×10^{-7}	9	38	0.008289	3.91×10^{-6}	13	53	0.00896	9.05×10^{-6}
	x_3	3	17	0.004096	5.81×10^{-8}	8	34	0.006702	7.43×10^{-6}	3	12	0.00426	8.47×10^{-6}
	x_4	7	34	0.00585	3.89×10^{-6}	11	46	0.009579	5.94×10^{-6}	15	61	0.01169	6.73×10^{-6}
	x_5	7	34	0.006133	6.36×10^{-6}	11	46	0.015328	8.97×10^{-6}	31	126	0.03646	9.03×10^{-6}
	x_6	8	37	0.006106	1.9×10^{-6}	12	49	0.01426	2.87×10^{-6}	15	60	0.01082	3.99×10^{-6}
5000	x_1	4	21	0.015836	7.25×10^{-7}	10	42	0.023953	5.49×10^{-6}	15	61	0.03215	4.25×10^{-6}
	x_2	4	21	0.014521	3.2×10^{-7}	9	38	0.021065	8.74×10^{-6}	14	57	0.02942	7.40×10^{-6}
	x_3	3	17	0.014517	1.3×10^{-7}	9	38	0.025437	4.01×10^{-6}	4	16	0.01107	1.01×10^{-7}
	x_4	7	34	0.028388	8.71×10^{-6}	12	50	0.028607	3.21×10^{-6}	16	65	0.04331	5.43×10^{-6}
	x_5	8	38	0.02787	1.49×10^{-6}	12	50	0.037806	4.84×10^{-6}	33	134	0.09379	7.78×10^{-6}
	x_6	8	37	0.027898	4.26×10^{-6}	12	49	0.029226	6.43×10^{-6}	15	60	0.04077	8.92×10^{-6}
10,000	x_1	4	21	0.028528	1.02×10^{-6}	10	42	0.045585	7.77×10^{-6}	15	61	0.06484	6.01×10^{-6}
	x_2	4	21	0.033782	4.52×10^{-7}	10	42	0.041715	2.98×10^{-6}	15	61	0.07734	3.77×10^{-6}
	x_3	3	17	0.029265	1.84×10^{-7}	9	38	0.036422	5.67×10^{-6}	4	16	0.02707	1.42×10^{-7}
	x_4	8	38	0.043301	1.29×10^{-6}	12	50	0.063527	4.53×10^{-6}	16	65	0.07941	7.69×10^{-6}
	x_5	8	38	0.043741	2.1×10^{-6}	12	50	0.049604	6.85×10^{-6}	34	138	0.14942	6.83×10^{-6}
	x_6	8	37	0.053666	6.02×10^{-6}	12	49	0.050153	9.09×10^{-6}	34	138	0.15224	8.81×10^{-6}
50,000	x_1	4	21	0.10816	2.29×10^{-6}	11	46	0.20624	4.19×10^{-6}	16	65	0.25995	4.89×10^{-6}
	x_2	4	21	0.11969	1.01×10^{-6}	10	42	0.16364	6.67×10^{-6}	15	61	0.24674	8.42×10^{-6}
	x_3	3	17	0.068644	4.11×10^{-7}	10	42	0.1539	3.06×10^{-6}	4	16	0.09405	3.18×10^{-7}
	x_4	8	38	0.16067	2.88×10^{-6}	13	54	0.20728	2.45×10^{-6}	36	146	0.55207	6.39×10^{-6}
	x_5	8	38	0.14484	4.7×10^{-6}	13	54	0.19421	3.69×10^{-6}	35	142	0.54679	9.05×10^{-6}
	x_6	9	41	0.161	1.41×10^{-6}	13	53	0.19386	4.9×10^{-6}	36	146	0.55764	7.59×10^{-6}
100,000	x_1	4	21	0.21825	3.24×10^{-6}	11	46	0.32512	5.93×10^{-6}	17	69	0.52595	5.68×10^{-6}
	x_2	4	21	0.16435	1.43×10^{-6}	10	42	0.30949	9.43×10^{-6}	16	65	0.52102	4.34×10^{-6}
	x_3	3	17	0.13072	5.81×10^{-7}	10	42	0.31031	4.32×10^{-6}	4	16	0.14864	4.50×10^{-7}
	x_4	8	38	0.29012	4.07×10^{-6}	13	54	0.38833	3.46×10^{-6}	36	146	1.05360	9.04×10^{-6}
	x_5	8	38	0.32821	6.65×10^{-6}	13	54	0.3522	5.22×10^{-6}	74	299	2.10730	8.55×10^{-6}
	x_6	9	41	0.43649	1.99×10^{-6}	13	53	0.3561	6.94×10^{-6}	37	150	1.08240	6.66×10^{-6}

Table 8. Numerical results for MFRM, ACGD and PDY for problem 8 with given initial points and dimensions.

		MFRM				ACGD				PDY			
Dimension	**Initial Point**	Iter	**Fval**	**Time**	**Norm**	**Iter**	**Fval**	**Time**	**Norm**	Iter	**Fval**	**Time**	**Norm**
1000	x_1	8	27	0.1502	1.52×10^{-6}	8	26	0.049826	6.09×10^{-6}	69	279	0.05538	8.95×10^{-6}
	x_2	8	27	0.042248	1.52×10^{-6}	8	26	0.017594	6.09×10^{-6}	270	1085	0.18798	9.72×10^{-6}
	x_3	26	114	0.03877	7.85×10^{-6}	8	26	0.010888	6.09×10^{-6}	24	52	0.02439	6.57×10^{-6}
	x_4	26	114	0.017542	7.85×10^{-6}	8	26	0.007873	6.09×10^{-6}	27	58	0.01520	7.59×10^{-6}
	x_5	26	114	0.067692	7.85×10^{-6}	8	26	0.060733	6.09×10^{-6}	28	61	0.04330	9.21×10^{-6}
	x_6	26	114	0.045173	7.85×10^{-6}	8	26	0.006889	6.09×10^{-6}	40	85	0.02116	8.45×10^{-6}
5000	x_1	6	28	0.023925	8.77×10^{-6}	4	13	0.011005	5.76×10^{-6}	658	2639	1.13030	9.98×10^{-6}
	x_2	15	70	0.043512	7.94×10^{-6}	4	13	0.009131	5.76×10^{-6}	27	58	0.05101	7.59×10^{-6}
	x_3	15	70	0.046458	7.94×10^{-6}	4	13	0.011311	5.76×10^{-6}	49	104	0.08035	8.11×10^{-6}
	x_4	15	70	0.044788	7.94×10^{-6}	4	13	0.010475	5.75×10^{-6}	40	85	0.07979	8.45×10^{-6}
	x_5	15	70	0.044639	7.94×10^{-6}	4	13	0.011034	5.77×10^{-6}	18	40	0.09128	9.14×10^{-6}
	x_6	15	70	0.043974	7.94×10^{-6}	4	13	0.00785	5.76×10^{-6}	17	38	0.18528	8.98×10^{-6}
10,000	x_1	11	54	0.06595	6.15×10^{-6}	5	20	0.024232	2.19×10^{-6}	49	104	0.20443	7.62×10^{-6}
	x_2	11	54	0.068125	6.15×10^{-6}	5	20	0.023511	2.19×10^{-6}	40	85	0.15801	8.45×10^{-6}
	x_3	11	54	0.065486	6.15×10^{-6}	5	20	0.023004	2.19×10^{-6}	19	42	0.37880	7.66×10^{-6}
	x_4	11	54	0.064515	6.15×10^{-6}	5	20	0.030435	2.19×10^{-6}	90	187	1.25802	9.7×10^{-6}
	x_5	11	54	0.056261	6.15×10^{-6}	5	20	0.021963	2.19×10^{-6}	988	1988	12.68259	9.93×10^{-6}
	x_6	11	54	0.067785	6.15×10^{-6}	5	20	0.021889	2.21×10^{-6}	27	58	0.32859	7.59×10^{-6}
50,000	x_1	7	38	0.17856	4.5×10^{-6}	5	23	0.087544	2.45×10^{-6}	19	42	0.52291	6.42×10^{-6}
	x_2	7	38	0.17862	4.5×10^{-6}	5	23	0.093227	2.45×10^{-6}	148	304	3.93063	9.92×10^{-6}
	x_3	7	38	0.17746	4.5×10^{-6}	5	23	0.087484	2.45×10^{-6}	937	1886	22.97097	9.87×10^{-6}
	x_4	7	38	0.17392	4.5×10^{-6}	5	23	0.086329	2.4×10^{-6}	27	58	0.68467	7.59×10^{-6}
	x_5	7	38	0.18035	4.5×10^{-6}	5	23	0.08954	2.4×10^{-6}	346	702	8.45043	9.79×10^{-6}
	x_6	7	38	0.17504	4.5×10^{-6}	5	23	0.093203	2.5×10^{-6}	40	85	0.99230	8.45×10^{-6}
100,000	x_1	28	122	0.91448	8.61×10^{-6}	4	20	0.14743	2.71×10^{-6}	-	-	-	-
	x_2	28	122	0.93662	8.61×10^{-6}	4	20	0.14823	2.7×10^{-6}	-	-	-	-
	x_3	28	122	0.90604	8.61×10^{-6}	4	20	0.1497	2.79×10^{-6}	-	-	-	-
	x_4	28	122	0.92351	8.61×10^{-6}	4	20	0.14844	2.37×10^{-6}	-	-	-	-
	x_5	28	122	0.91896	8.61×10^{-6}	4	20	0.12346	1.66×10^{-6}	-	-	-	-
	x_6	28	122	0.91294	8.61×10^{-6}	4	20	0.12522	2.11×10^{-6}	-	-	-	-

Figure 1 reveals that MFRM performed better in terms of number of Iterations, as it solves and wins over 70 percent of the problems with less number of Iterations, while ACGD and PDY solve and win over 40 and almost 10 percent respectively. The story is a little bit different in Figure 2 as ACGD method was very competitive. However, MFRM method performed a little bit better by solving and winning over 50 percent of the problems with less CPU time as against ACGD method which solves and wins less than 50 percent of the problems considered. The PDY method had the least performance with just 10 percent success. The interpretation of Figure 3 was similar to that of Figure 1. Finally, in Table 11 we report numerical results for MFRM, ACGD and PDY for problem 2 with given initial points and dimensions with double float (10^{-16}) accuracy.

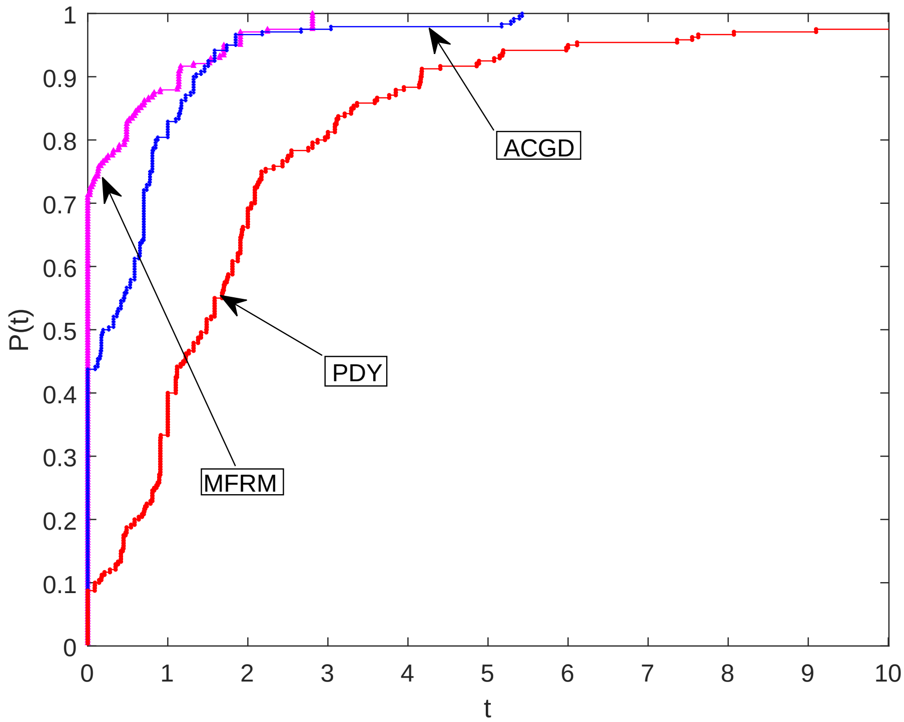

Figure 1. Performance profiles for the number of iterations.

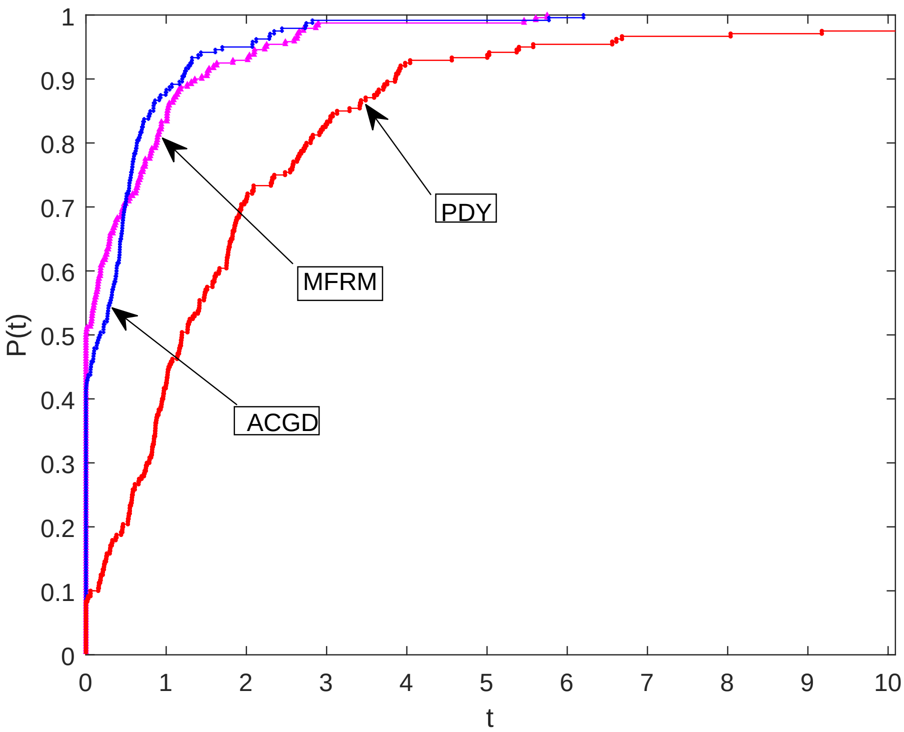

Figure 2. Performance profiles for the CPU time (in seconds).

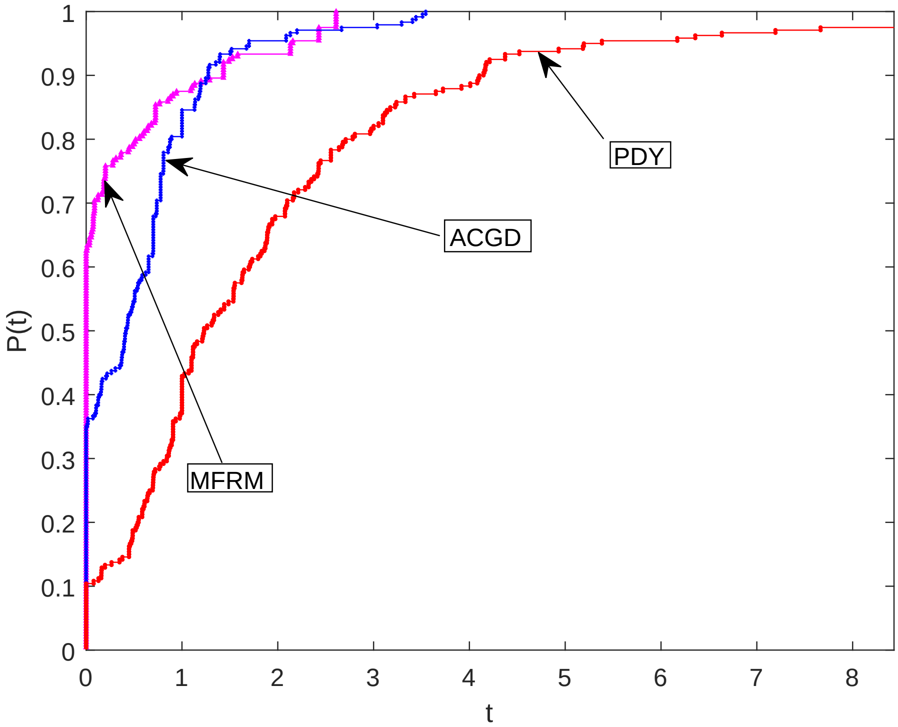

Figure 3. Performance profiles for the number of function evaluations.

4.1. Experiments on Solving Sparse Signal Problems

There were many problems in signal processing and statistical inference involving finding sparse solutions to ill-conditioned linear systems of equations. Among popular approaches was minimizing an objective function which contains quadratic (ℓ_2) error term and a sparse ℓ_1-regularization term, i.e.,

$$\min_x \frac{1}{2}\|y - Bx\|_2^2 + \eta\|x\|_1, \tag{25}$$

where $x \in R^n$, $y \in R^k$ is an observation, $B \in R^{k\times n}$ $(k << n)$ is a linear operator, η is a non-negative parameter, $\|x\|_2$ denotes the Euclidean norm of x and $\|x\|_1 = \sum_{i=1}^n |x_i|$ is the ℓ_1-norm of x. It is easy to see that problem (25) is a convex unconstrained minimization problem. Due to the fact that if the original signal is sparse or approximately sparse in some orthogonal basis, problem (25) frequently appears in compressive sensing, and hence an exact restoration can be produced by solving (25).

Iterative methods for solving (25) have been presented in many papers (see [42–45]). The most popular method among these methods is the gradient-based method and the earliest gradient projection method for sparse reconstruction (GPRS) was proposed by Figueiredo et al. [44]. The first step of the GPRS method is to express (25) as a quadratic problem using the following process. Consider a point $x \in \mathbb{R}^n$ such that $x = u - v$, where $u, v \geq 0$. u and v are chosen in such a way that x is splitted into its positive and negative parts as follows $u_i = (x_i)_+$, $v_i = (-x_i)_+$ for all $i = 1, 2, ..., n$, and $(.)_+ = \max\{0, .\}$. By definition of ℓ_1-norm, we have $\|x\|_1 = e_n^T u + e_n^T v$, where $e_n = (1, 1, ..., 1)^T \in R^n$. Now (25) can be written as

$$\min_{u,v} \frac{1}{2}\|y - B(u - v)\|_2^2 + \eta e_n^T u + \eta e_n^T v, \qquad u \geq 0, \quad v \geq 0, \tag{26}$$

which is a bound-constrained quadratic program. However, from [44], Equation (26) can be written in standard form as

$$\min_z \frac{1}{2} z^T D z + c^T z, \qquad \text{such that} \quad z \geq 0, \tag{27}$$

where $z = \begin{pmatrix} u \\ v \end{pmatrix}$, $\quad c = \omega e_{2n} + \begin{pmatrix} -b \\ b \end{pmatrix}$, $\quad b = B^T y$, $\quad D = \begin{pmatrix} B^T B & -B^T B \\ -B^T B & B^T B \end{pmatrix}$. Clearly, D is a positive semi-definite matrix, which implies that Equation (27) is a convex quadratic problem.

Xiao et al. [20] translated (27) into a linear variable inequality problem which is equivalent to a linear complementarity problem. Moreover, z is a solution of the linear complementarity problem if and only if it is a solution of the following nonlinear equation:

$$F(z) = \min\{z, Dz + c\} = 0. \tag{28}$$

The function F is a vector-valued function and the "min" was interpreted as component wise minimum. Furthermore, F was proved to be continuous and monotone in [46]. Therefore problem (25) can be translated into problem (1) and thus MFRM method can be applied to solve it.

In this experiment, we consider a simple compressive sensing possible situation, where our goal is to reconstruct a sparse signal of length n from k observations. The quality of recovery is assessed by mean of squared error (MSE) to the original signal $\tilde{x}$,

$$MSE = \frac{1}{n}\|\tilde{x} - x_*\|^2,$$

where x_* is the recovered signal. The signal size is chosen as $n = 2^{11}$, $k = 2^9$ and the original signal contains 2^6 randomly nonzero elements. In addition, the measurement y is distributed with noise, that is, $y = B\tilde{x} + \varrho$, where B is a randomly generated Gaussian matrix and ϱ is the Gaussian noise distributed normally with mean 0 and variance 10^{-4}.

To demonstrate the performance of the MFRM method in signal recovery problems, we compare it with the conjugate gradient descent CGD [20] and projected conjugate gradient PCG [23] methods. The parameters in PCG and CGD methods are chosen as $\gamma = 10$, $\sigma = 10^{-4}$, $\rho = 0.5$. However, we chose $\gamma = 1$, $\sigma = 10^{-4}$, $\rho = 0.9$ and $\mu = 0.01$ in MFRM method. For fairness in comparison, each code was run from the same initial point, same continuation technique on the parameter η, and observed only the behavior of the convergence of each method to have a similar accurate solution. The experiment was initialized with $x_0 = B^T y$ and terminates when

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

where $f(x_k) = \frac{1}{2}\|y - Bx_k\|_2^2 + \eta\|x_k\|_1$.

In Figures 4 and 5, MFRM, CGD and PCG methods recovered the disturbed signal almost exactly. The experiment was repeated for 20 different noise samples (see Table 9). It can be observed that the MFRM is more efficient in terms of the number of Iterations and CPU time than CGD and PCG methods in most cases. Furthermore, MFRM was able to achieve the least MSE in nine (9) out of the twenty (20) experiments. To reveal visually the performance of both methods, two figures were plotted to demonstrate their convergence behavior based on MSE, objective function values, the number of Iterations and CPU time (see Figures 6 and 7). It can also be observed that MFRM requires less computing time to achieve similar quality resolution. This can be seen graphically in Figures 6 and 7 which illustrate that the objective function values obtained by MFRM decrease faster throughout the entire Iteration process.

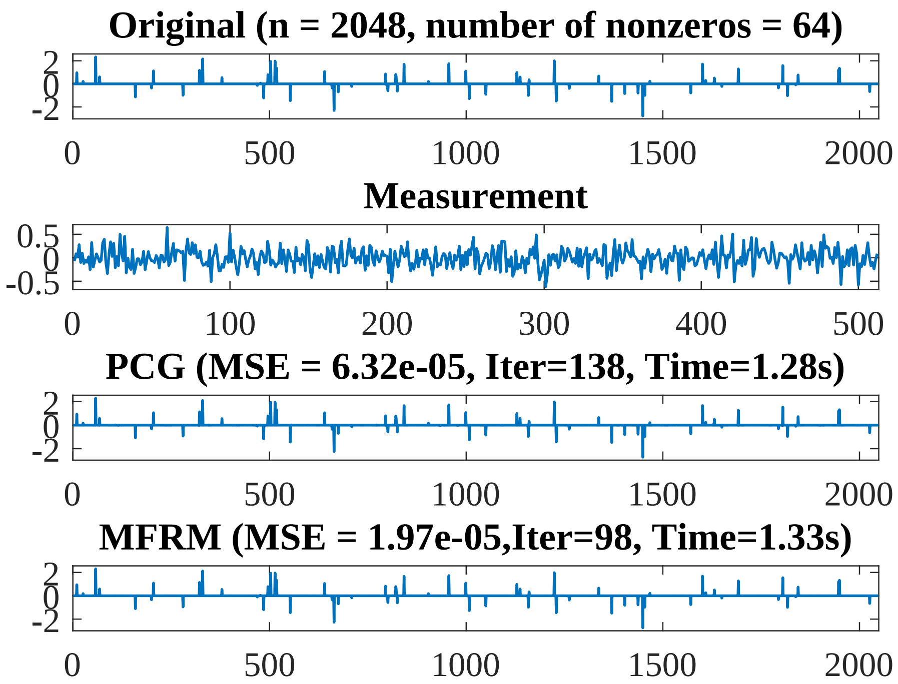

Figure 4. (**top**) to (**bottom**) The original image, the measurement, and the recovered signals by projected conjugate gradient PCG and modified descent Fletcher–Reeves CG method (MFRM) methods.

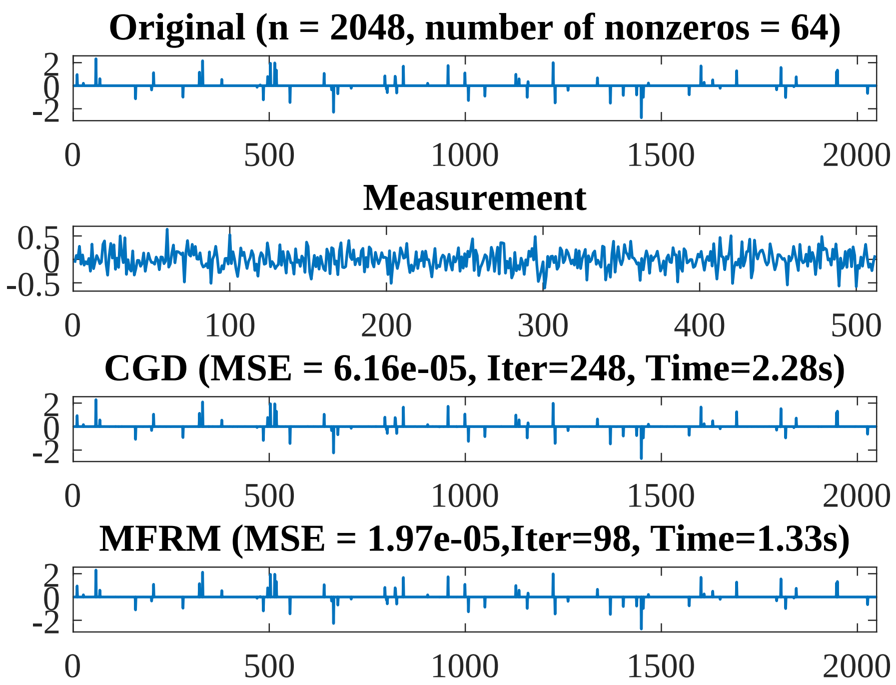

Figure 5. (**top**) to (**bottom**) The original image, the measurement, and the recovered signals by conjugate gradient descent (CGD) and MFRM methods.

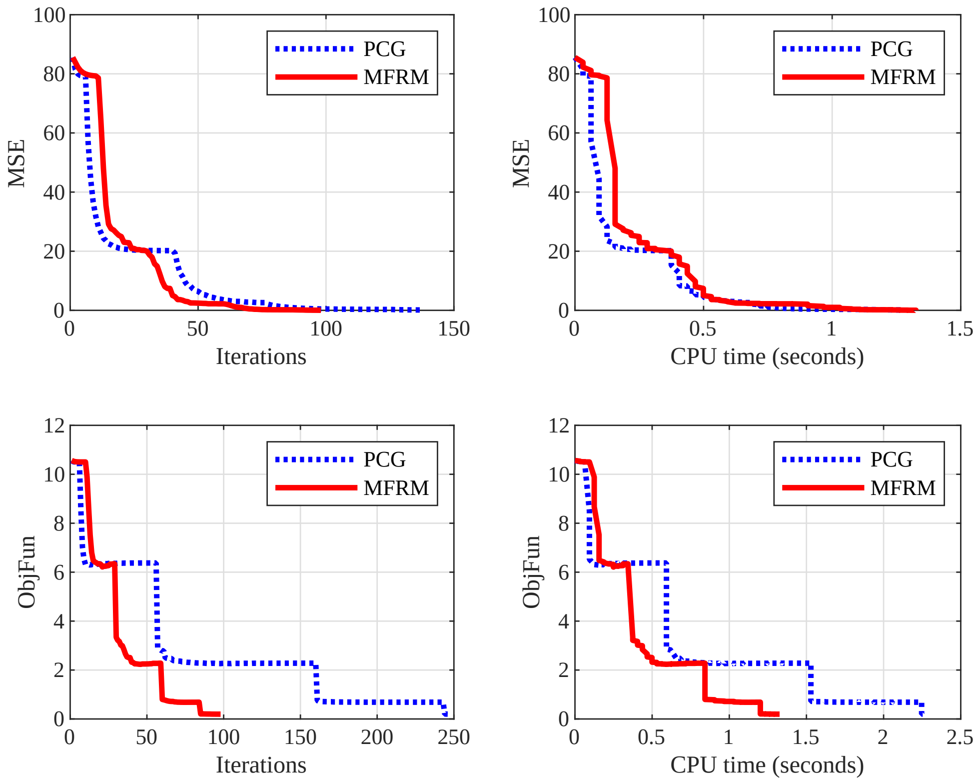

Figure 6. Comparison result of PCG and MFRM. The x-axis represent the number of Iterations ((**top left**) and (**bottom left**)) and CPU time in seconds ((**top right**) and (**bottom right**)). The y-axis represent the MSE ((**top left**) and (**top right**)) and the objective function values ((**bottom left**) and (**bottom right**)).

Table 9. Twenty experiment results of ℓ_1-norm regularization problem for CGD, PCG and MFRM methods.

S/N	**Iter**			**Time**			**MSE**		
	CGD	**PCG**	**MFRM**	**CGD**	**PCG**	**MFRM**	**CGD**	**PCG**	**MFRM**
1	248	138	98	2.28	1.28	1.33	6.16×10^{-5}	6.32×10^{-5}	1.97×10^{-5}
2	234	138	117	3.37	1.26	1.19	4.08×10^{-5}	3.36×10^{-5}	5.40×10^{-5}
3	224	152	104	1.90	1.29	0.97	2.78×10^{-5}	1.78×10^{-5}	1.02×10^{-5}
4	230	143	117	3.21	2.48	1.17	4.08×10^{-5}	3.36×10^{-5}	5.40×10^{-5}
5	152	119	114	1.65	1.03	1.15	1.23×10^{-5}	2.07×10^{-5}	5.49×10^{-5}
6	223	127	110	1.89	2.56	1.83	3.33×10^{-5}	6.08×10^{-5}	6.50×10^{-6}
7	156	120	125	1.37	1.01	1.20	4.25×10^{-5}	3.26×10^{-5}	1.46×10^{-5}
8	213	89	10	1.90	0.78	1.12	1.86×10^{-5}	3.77×10^{-4}	1.31×10^{-5}
9	227	152	118	2.14	1.53	1.45	2.75×10^{-5}	1.54×10^{-5}	8.11×10^{-6}
10	201	142	101	2.22	1.64	1.01	6.75×10^{-5}	1.86×10^{-5}	1.17×10^{-5}
11	200	151	90	1.70	1.42	0.90	2.36×10^{-5}	1.29×10^{-5}	3.81×10^{-5}
12	202	153	91	1.75	1.34	0.84	6.94×10^{-5}	2.99×10^{-5}	9.21×10^{-5}
13	208	128	125	1.89	1.12	1.26	1.71×10^{-5}	1.42×10^{-5}	9.20×10^{-6}
14	161	145	122	1.47	1.28	1.26	1.15×10^{-5}	8.75×10^{-6}	4.36×10^{-6}
15	227	160	100	1.97	1.42	1.00	3.41×10^{-5}	2.40×10^{-5}	1.54×10^{-5}
16	269	172	88	2.51	1.67	0.98	3.90×10^{-5}	6.59×10^{-5}	2.08×10^{-4}
17	210	129	105	1.84	1.19	1.11	2.11×10^{-5}	1.89×10^{-5}	6.22×10^{-5}
18	225	132	96	1.93	1.15	1.00	3.87×10^{-5}	7.78×10^{-5}	9.49×10^{-5}
19	152	120	92	1.37	1.09	0.87	2.12×10^{-5}	1.32×10^{-5}	4.03×10^{-5}
20	151	128	113	1.31	1.15	1.06	4.48×10^{-5}	1.85×10^{-5}	1.71×10^{-5}

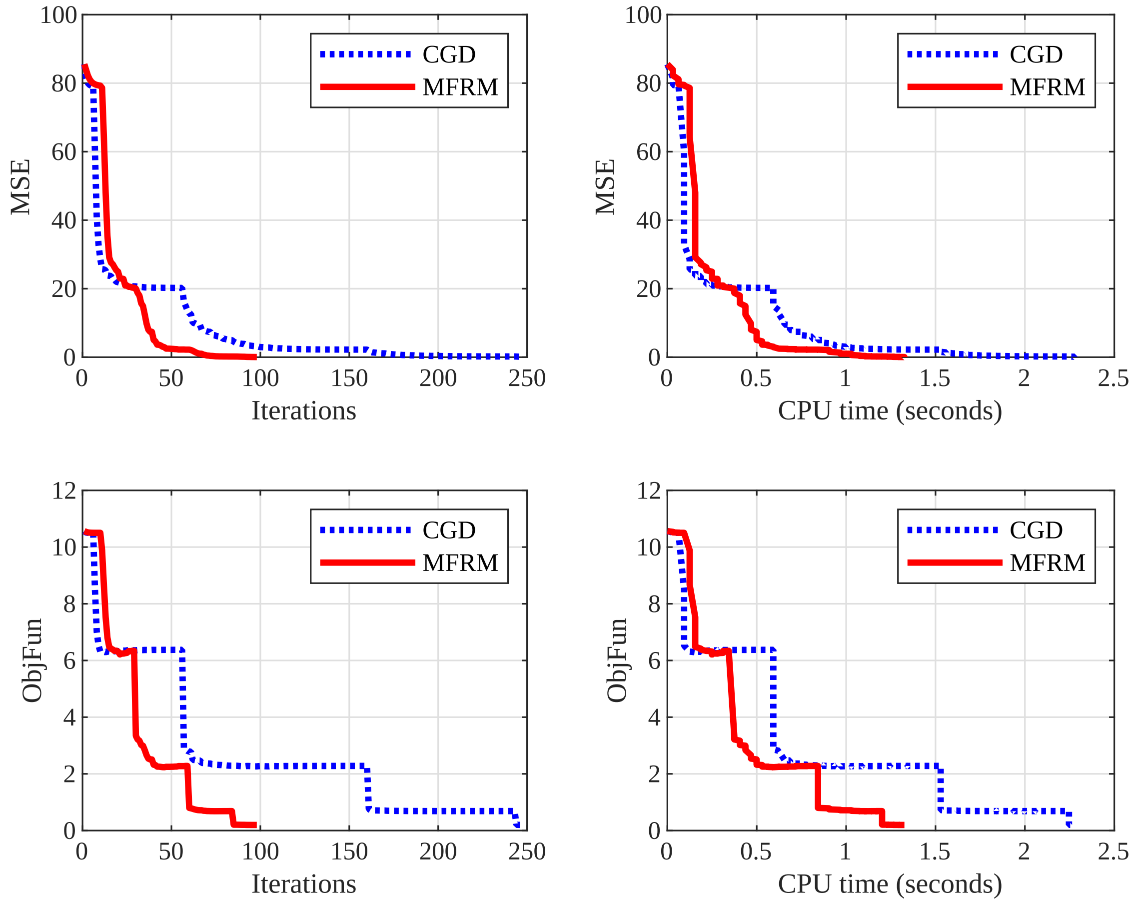

Figure 7. Comparison result of PCG and MFRM. The x-axis represent the number of Iterations ((**top left**) and (**bottom left**)) and CPU time in seconds ((**top right**) and (**bottom right**)). The y-axis represent the MSE ((**top left**) and (**top right**)) and the objective function values ((**bottom left**) and (**bottom right**)).

4.2. Experiments on Blurred Image Restoration

In this subsection, we test the performance of MFRM in restoring a blurred image. We use the following well-known gray test images; (P1) Cameraman, (P2) Lena, (P3) House and (P4) Peppers for the experiments. We use 4 different Gaussian blur kernels with a standard deviation v to compare the robustness of MFRM method with CGD method proposed in [20].

To assess the performance of each algorithm tested with respect to the metrics that indicate better quality of restoration, in Table 10 we reported the objective function (ObjFun) at the approximate solution, the MSE, the signal-to-noise-ratio (SNR) which is defined as

$$\mathrm{SNR} = 20 \times \log_{10}\left(\frac{\|\bar{x}\|}{\|x - \bar{x}\|}\right),$$

and the structural similarity (SSIM) index that measure the similarity between the original image and the restored image [47] for each of the 16 experiments. The MATLAB implementation of the SSIM index can be obtained at http://www.cns.nyu.edu/~lcv/ssim/.

The original, blurred and restored images by each of the algorithm are given in Figures 8–11. The figures demonstrate that both the two algorithms can restore the blurred images. In contrast to the CGD, the quality of the restored image by MFRM is superior in most cases. Table 11 reported numerical results for MFRM, ACGD and PDY for problem 2.

Table 10. Efficiency comparison based on the value of the objective function (ObjFun) mean-square-error (MSE), SNR and the SSIM index under different Pi(v).

Image	ObjFun		MSE		SNR		SSIM	
	MFRM	CGD	MFRM	CGD	MFRM	CGD	MFRM	CGD
P1(1×10^{-4})	1.43×10^6	1.47×10^6	133.90	177.57	21.28	20.05	0.86	0.83
P1(1×10^{-1})	1.43×10^6	1.48×10^6	130.60	177.69	21.39	20.5	0.86	0.83
P1(0.25)	1.47×10^6	1.48×10^6	145.27	177.72	20.93	20.05	0.85	0.83
P1(6.25)	1.58×10^6	1.65×10^6	146.06	183.96	20.9	19.9	0.75	0.79
P2(1×10^{-4})	1.61×10^6	1.65×10^6	36.88	57.55	27.59	25.65	0.88	0.86
P2(1×10^{-1})	1.61×10^6	1.65×10^6	36.85	57.61	27.59	25.65	0.88	0.86
P2(0.25)	1.62×10^6	1.66×10^6	37.78	57.68	27.48	25.64	0.88	0.86
P2(6.25)	1.77×10^6	1.82×10^6	56.65	58.96	25.72	25.55	0.76	0.83
P3(1×10^{-4})	5.74×10^6	5.89×10^6	41.63	44.48	26.26	25.97	0.9	0.88
P3(1×10^{-1})	5.75×10^6	5.90×10^6	42.42	44.54	26.17	25.96	0.89	0.88
P3(0.25)	5.76×10^6	5.91×10^6	43.33	44.65	26.08	25.95	0.88	0.88
P3(6.25)	6.35×10^6	6.60×10^6	106.79	48.47	22.16	25.6	0.63	0.85
P4(1×10^{-4})	1.40×10^6	1.48×10^6	88.81	122.44	22.9	21.5	0.87	0.84
P4(1×10^{-1})	1.41×10^6	1.48×10^6	89.22	122.56	22.88	21.5	0.87	0.84
P4(0.25)	1.41×10^6	1.49×10^6	89.86	122.56	22.85	21.5	0.87	0.84
P4(6.25)	1.56×10^6	1.69×10^6	116.79	138.97	21.71	20.95	0.76	0.82

Table 11. Numerical results for modified Fletcher-Reeves method MFRM, accelerated conjugate gradient descent (ACGD) and projected Dai-Yuan (PDY) methods for problem 2 with given initial points and dimensions with double float (10^{-16}) accuracy.

		MFRM				ACGD				PDY			
Dimension	Initial Point	Iter	Fval	Time	Norm	Iter	Fval	Time	Norm	Iter	Fval	Time	Norm
1000	$x1$	8	27	0.14061	9.47×10^{-19}	12	53	0.030479	3.32×10^{-18}	30	119	0.04027	4.76×10^{-19}
	$x2$	8	36	0.010782	1.49×10^{-18}	7	20	0.013503	1.08×10^{-18}	36	153	0.034454	3.51×10^{-18}
	$x3$	7	20	0.008263	1.21×10^{-18}	13	56	0.021302	3.26×10^{-18}	38	161	0.038168	3.51×10^{-18}
	$x4$	8	23	0.015654	1.80×10^{-19}	12	51	0.02056	3.31×10^{-18}	39	165	0.057793	3.51×10^{-18}
	$x5$	11	38	0.018461	1.59×10^{-18}	14	59	0.088858	3.34×10^{-18}	41	173	0.069756	3.51×10^{-18}
	$x6$	10	34	0.016788	1.07×10^{-18}	10	32	0.012069	5.83×10^{-19}	40	169	0.03311	3.50×10^{-18}
5000	$x1$	9	33	0.028658	7.22×10^{-19}	12	54	0.041685	1.52×10^{-18}	35	149	0.10692	1.57×10^{-18}
	$x2$	7	23	0.024046	2.18×10^{-19}	9	41	0.049194	1.55×10^{-18}	37	157	0.12219	1.57×10^{-18}
	$x3$	6	17	0.03436	3.89×10^{-19}	14	61	0.094129	1.47×10^{-18}	33	131	0.10635	1.06×10^{-19}
	$x4$	8	26	0.03133	7.17×10^{-19}	14	60	0.065147	1.47×10^{-18}	39	165	0.18361	1.57×10^{-18}
	$x5$	9	31	0.036727	5.84×10^{-19}	10	43	0.1165	1.47×10^{-18}	36	144	0.2178	7.43×10^{-20}
	$x6$	10	34	0.030168	6.41×10^{-19}	12	51	0.038218	1.51×10^{-18}	38	161	0.13144	1.57×10^{-18}
10,000	$x1$	8	28	0.064617	1.89×10^{-19}	11	50	0.068567	1.03×10^{-18}	35	149	0.2253	1.11×10^{-18}
	$x2$	6	19	0.044204	1.90×10^{-19}	14	62	0.15949	1.09×10^{-18}	32	128	0.34325	8.21×10^{-20}
	$x3$	6	17	0.045192	1.45×10^{-19}	18	78	0.10766	1.04×10^{-18}	39	165	0.23899	1.11×10^{-18}
	$x4$	10	35	0.055408	4.99×10^{-19}	12	52	0.061589	1.06×10^{-18}	39	165	0.23162	1.11×10^{-18}
	$x5$	7	20	0.038439	2.06×10^{-19}	14	60	0.087394	1.05×10^{-18}	40	169	0.28998	1.11×10^{-18}
	$x6$	9	29	0.065318	5.27×10^{-19}	16	68	0.09917	1.03×10^{-18}	40	170	0.22564	1.11×10^{-18}
50,000	$x1$	7	26	0.21017	1.93×10^{-19}	23	100	0.51879	4.79×10^{-19}	34	145	0.92896	4.96×10^{-19}
	$x2$	6	21	0.24752	2.09×10^{-19}	25	108	0.64677	4.90×10^{-19}	36	153	0.9954	4.96×10^{-19}
	$x3$	6	17	0.11243	6.27×10^{-20}	23	99	0.50402	4.93×10^{-19}	38	161	0.96768	4.96×10^{-19}
	$x4$	7	20	0.13442	1.02×10^{-19}	24	102	0.63664	4.75×10^{-19}	79	326	1.7542	4.96×10^{-19}
	$x5$	9	30	0.20288	7.25×10^{-20}	25	106	0.51116	4.78×10^{-19}	78	322	1.7246	4.96×10^{-19}
	$x6$	12	52	0.36526	2.28×10^{-19}	23	97	0.56342	4.76×10^{-19}	80	330	1.6812	4.96×10^{-19}
100,000	$x1$	7	27	0.36065	6.53×10^{-20}	23	100	0.88236	3.26×10^{-19}	30	119	1.2102	9.26×10^{-21}
	$x2$	5	14	0.20041	3.91×10^{-20}	25	108	0.90777	3.27×10^{-19}	35	149	1.5699	3.51×10^{-19}
	$x3$	7	24	0.34075	1.47×10^{-19}	25	107	0.95898	3.26×10^{-19}	40	170	1.7126	3.51×10^{-19}
	$x4$	8	31	0.40444	2.09×10^{-20}	24	102	0.83332	3.38×10^{-19}	151	614	5.8306	3.51×10^{-19}
	$x5$	8	26	0.52598	5.03×10^{-20}	25	106	1.0223	3.47×10^{-19}	151	614	5.6777	3.50×10^{-19}
	$x6$	7	20	0.33434	1.45×10^{-19}	23	97	0.87438	3.33×10^{-19}	153	622	5.7906	3.51×10^{-19}

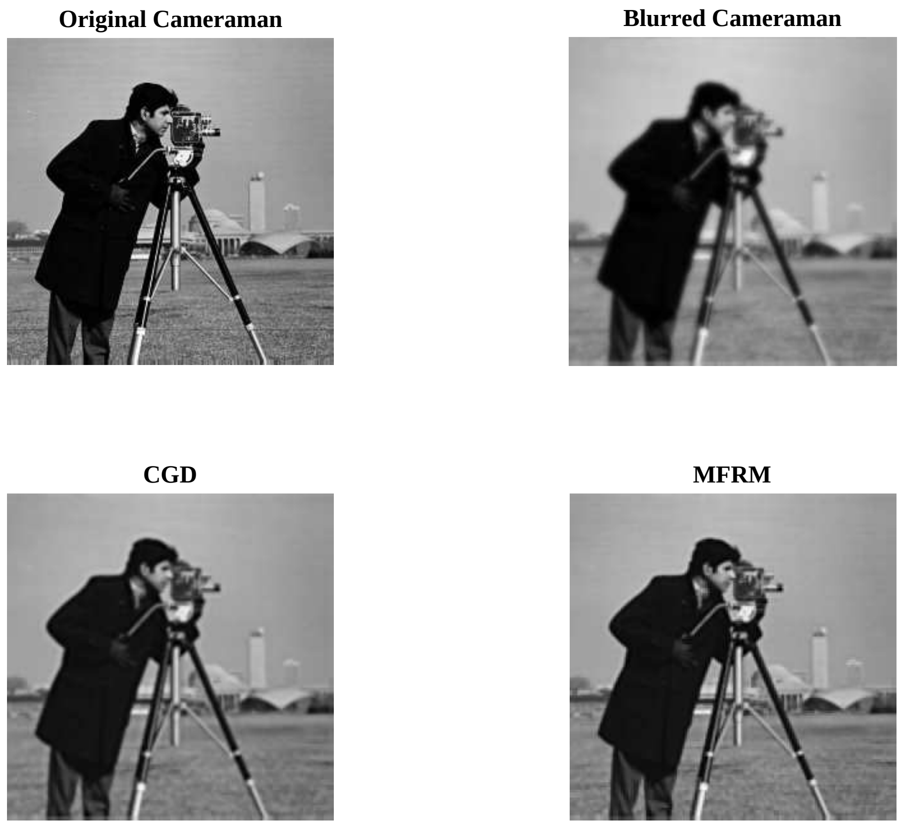

Figure 8. The original image (**top left**), the blurred image (**top right**), the restored image by CGD (**bottom left**) with time = 3.70, signal-to-noise-ratio (SNR) = 20.05 and structural similarity (SSIM) = 0.83, and by MFRM (**bottom right**) with time = 1.97, SNR = 21.28 and SSIM = 0.86.

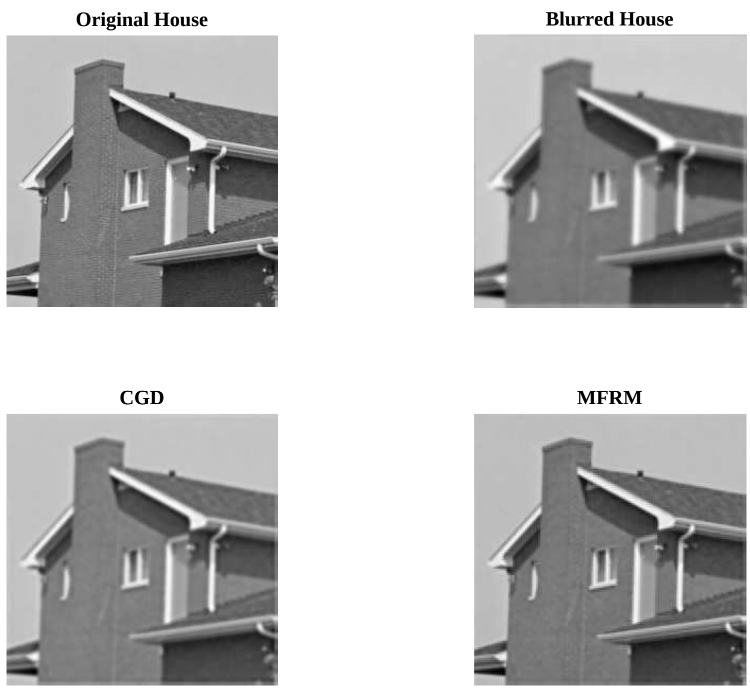

Figure 9. The original image (**top left**), the blurred image (**top right**), the restored image by CGD (**bottom left**) with Time = 1.95, SNR = 25.65 and SSIM = 0.86, and by MFRM (**bottom right**) with Time = 3.59, SNR = 27.59 and SSIM = 0.88.

Figure 10. The original image (**top left**), the blurred image (**top right**), the restored image by CGD (**bottom left**) with time = 5.38, SNR = 25.97 and SSIM = 0.88, and by MFRM (**bottom right**) with time = 38.77, SNR = 26.26 and SSIM = 0.90.

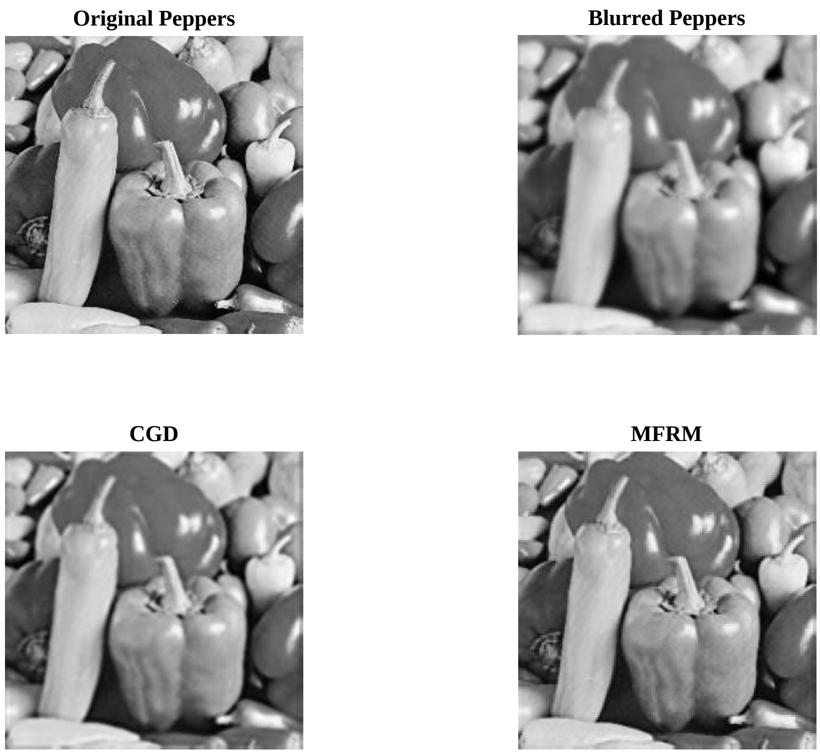

Figure 11. The original image (**top left**), the blurred image (**top right**), the restored image by CGD (**bottom left**) with Time = 2.48, SNR = 21.50 and SSIM = 0.84, and by MFRM (**bottom right**) with Time = 4.93, SNR = 22.90 and SSIM = 0.87.

5. Conclusions

In this paper, a modified conjugate gradient method for solving monotone nonlinear equations with convex constraints was presented which is similar to that in [3]. The proposed method is suitable for non-smooth equations. Under some suitable assumptions, the global convergence of the proposed method was demonstrated. Numerical results were presented to show the effectiveness of the MFRM method compared to the ACGD and PDY methods for the given constrained monotone equation problems. Finally, the MFRM was also shown to be effective in decoding sparse signals and restoration of blurred images.

Author Contributions: conceptualization, A.B.A.; methodology, A.B.A.; software, H.M.; validation, P.K., A.M.A. and K.S.; formal analysis, P.K. and K.S.; investigation, P.K. and H.M.; resources, P.K. and K.S.; data curation, H.M. and A.M.A.; writing–original draft preparation, A.B.A.; writing–review and editing, H.M.; visualization, A.M.A. and K.S.; supervision, P.K.; project administration, P.K. and K.S.; funding acquisition, P.K. and K.S.

Funding: Petchra Pra Jom Klao Doctoral Scholarship for Ph.D. program of King Mongkut's University of Technology Thonburi (KMUTT). This project was partially supported by the Thailand Research Fund (TRF) and the King Mongkut's University of Technology Thonburi (KMUTT) under the TRF Research Scholar Award (Grant No. RSA6080047). Moreover, Kanokwan Sitthithakerngkiet was supported by Faculty of Applied Science, King Mongkuts University of Technology North Bangkok. Contract no. 6242104.

Acknowledgments: We thank Associate Professor Jin Kiu Liu for providing us with the access of the CGD-CS MATLAB codes. The authors acknowledge the financial support provided by King Mongkut's University of Technology Thonburi through the "KMUTT 55th Anniversary Commemorative Fund". The first author was supported by the "Petchra Pra Jom Klao Ph.D. Research Scholarship from King Mongkut's University of Technology Thonburi".

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Abubakar, A.B.; Kumam, P.; Awwal, A.M. A Descent Dai-Liao Projection Method for Convex Constrained Nonlinear Monotone Equations with Applications. *Thai J. Math.* **2018**, *17*, 128–152.
2. Abubakar, A.B.; Kumam, P. A descent Dai-Liao conjugate gradient method for nonlinear equations. *Numer. Algorithms* **2019**, *81*, 197–210. [CrossRef]
3. Abubakar, A.B.; Kumam, P. An improved three-term derivative-free method for solving nonlinear equations. *Comput. Appl. Math.* **2018**, *37*, 6760–6773. [CrossRef]
4. Mohammad, H.; Abubakar, A.B. A positive spectral gradient-like method for nonlinear monotone equations. *Bull. Comput. Appl. Math.* **2017**, *5*, 99–115.
5. Muhammed, A.A.; Kumam, P.; Abubakar, A.B.; Wakili, A.; Pakkaranang, N. A New Hybrid Spectral Gradient Projection Method for Monotone System of Nonlinear Equations with Convex Constraints. *Thai J. Math.* **2018**, *16*, 125–147.
6. Zhou, W.J.; Li, D.H. A globally convergent BFGS method for nonlinear monotone equations without any merit functions. *Math. Comput.* **2008**, *77*, 2231–2240. [CrossRef]
7. Yan, Q.R.; Peng, X.Z.; Li, D.H. A globally convergent derivative-free method for solving large-scale nonlinear monotone equations. *J. Comput. Appl. Math.* **2010**, *234*, 649–657. [CrossRef]
8. DiRksEandM, S.P.; FERRis, C. A collection of nonlinear mixed complementarity problems. *Optim. Methods Softw.* **1995**, *5*, 319–345.
9. Meintjes, K.; Morgan, A.P. A methodology for solving chemical equilibrium systems. *Appl. Math. Comput.* **1987**, *22*, 333–361. [CrossRef]
10. Bellavia, S.; Macconi, M.; Morini, B. STRSCNE: A Scaled Trust-Region Solver for Constrained Nonlinear Equations. *Comput. Optim. Appl.* **2004**, *28*, 31–50. [CrossRef]
11. Kanzow, C.; Yamashita, N.; Fukushima, M. Levenberg–Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints. *J. Comput. Appl. Math.* **2004**, *172*, 375–397. [CrossRef]
12. Papp, Z.; Rapajić, S. FR type methods for systems of large-scale nonlinear monotone equations. *Appl. Math. Comput.* **2015**, *269*, 816–823. [CrossRef]

13. Zhou, W.; Wang, F. A PRP-based residual method for large-scale monotone nonlinear equations. *Appl. Math. Comput.* **2015**, *261*, 1–7. [CrossRef]
14. Dai, Y.H.; Yuan, Y. A nonlinear conjugate gradient method with a strong global convergence property. *SIAM J. Optim.* **1999**, *10*, 177–182. [CrossRef]
15. Polak, E.; Ribiere, G. Note sur la convergence de méthodes de directions conjuguées. *Revue Française D'informatique et de Recherche Opérationnelle Série Rouge* **1969**, *3*, 35–43. [CrossRef]
16. Polyak, B.T. The conjugate gradient method in extremal problems. *USSR Comput. Math. Math. Phys.* **1969**, *9*, 94–112. [CrossRef]
17. Hager, W.W.; Zhang, H. A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM J. Optim.* **2005**, *16*, 170–192. [CrossRef]
18. Dai, Y.H.; Liao, L.Z. New conjugacy conditions and related nonlinear conjugate gradient methods. *Appl. Math. Optim.* **2001**, *43*, 87–101. [CrossRef]
19. Fletcher, R.; Reeves, C.M. Function minimization by conjugate gradients. *Comput. J.* **1964**, *7*, 149–154. [CrossRef]
20. Xiao, Y.; Zhu, H. A conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing. *J. Math. Anal. Appl.* **2013**, *405*, 310–319. [CrossRef]
21. Solodov, M.V.; Svaiter, B.F. A globally convergent inexact Newton method for systems of monotone equations. In *Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods*; Springer: Boston, MA, USA, 1998; pp. 355–369.
22. Liu, S.Y.; Huang, Y.Y.; Jiao, H.W. Sufficient descent conjugate gradient methods for solving convex constrained nonlinear monotone equations. In *Abstract and Applied Analysis*; Hindawi: New York, NY, USA, 2014; Volume 2014.
23. Liu, J.; Li, S. A projection method for convex constrained monotone nonlinear equations with applications. *Comput. Math. Appl.* **2015**, *70*, 2442–2453. [CrossRef]
24. Sun, M.; Liu, J. Three derivative-free projection methods for nonlinear equations with convex constraints. *J. Appl. Math. Comput.* **2015**, *47*, 265–276. [CrossRef]
25. Sun, M.; Liu, J. New hybrid conjugate gradient projection method for the convex constrained equations. *Calcolo* **2016**, *53*, 399–411. [CrossRef]
26. Ou, Y.; Li, J. A new derivative-free SCG-type projection method for nonlinear monotone equations with convex constraints. *J. Appl. Math. Comput.* **2018**, *56*, 195–216. [CrossRef]
27. Ding, Y.; Xiao, Y.; Li, J. A class of conjugate gradient methods for convex constrained monotone equations. *Optimization* **2017**, *66*, 2309–2328. [CrossRef]
28. Liu, J.; Feng, Y. A derivative-free iterative method for nonlinear monotone equations with convex constraints. *Numer. Algorithms* **2018**, 1–18. [CrossRef]
29. Kabanikhin, S.I. Definitions and examples of inverse and ill-posed problems. *J. Inverse Ill-Posed Probl.* **2008**, *16*, 317–357. [CrossRef]
30. Belishev, M.I.; Kurylev, Y.V. Boundary control, wave field continuation and inverse problems for the wave equation. *Comput. Math. Appl.* **1991**, *22*, 27–52. [CrossRef]
31. Beilina, L.; Klibanov, M.V. A Globally Convergent Numerical Method for a Coefficient Inverse Problem. *SIAM J. Sci. Comput.* **2008**, *31*, 478–509. [CrossRef]
32. Kabanikhin, S.; Shishlenin, M. Boundary control and Gel'fand–Levitan–Krein methods in inverse acoustic problem. *J. Inverse Ill-Posed Probl.* **2004**, *12*, 125–144. [CrossRef]
33. Lukyanenko, D.; Grigorev, V.; Volkov, V.; Shishlenin, M. Solving of the coefficient inverse problem for a nonlinear singularly perturbed two dimensional reaction diffusion equation with the location of moving front data. *Comput. Math. Appl.* **2019**, *77*, 1245–1254. [CrossRef]
34. Van, S.B.; Elber, G. Solving piecewise polynomial constraint systems with decomposition and a subdivision-based solver. *Computer-Aided Design* **2017**, *90*, 37–47.
35. Aizenshtein, M.; Bartoň, M.; Elber, G. Global solutions of well-constrained transcendental systems using expression trees and a single solution test. *Computer Aided Geometric Design* **2012**, *29*, 265–279.
36. Bartoň, M. Solving polynomial systems using no-root elimination blending schemes. *Computer-Aided Design* **2011**, *43*, 1870–1878.
37. Wang, X.Y.; Li, S.J.; Kou, X.P. A self-adaptive three-term conjugate gradient method for monotone nonlinear equations with convex constraints. *Calcolo* **2016**, *53*, 133–145. [CrossRef]

38. La Cruz, W.; Martínez, J.; Raydan, M. Spectral residual method without gradient information for solving large-scale nonlinear systems of equations. *Math. Comput.* **2006**, *75*, 1429–1448. [CrossRef]
39. Bing, Y.; Lin, G. An Efficient Implementation of Merrills Method for Sparse or Partially Separable Systems of Nonlinear Equations. *SIAM J. Optim.* **1991**, *1*, 206–221, doi:10.1137/0801015. [CrossRef]
40. Yu, Z.; Lin, J.; Sun, J.; Xiao, Y.H.; Liu, L.Y.; Li, Z.H. Spectral gradient projection method for monotone nonlinear equations with convex constraints. *Appl. Numer. Math.* **2009**, *59*, 2416–2423. [CrossRef]
41. Dolan, E.D.; Moré, J.J. Benchmarking optimization software with performance profiles. *Math. Program.* **2002**, *91*, 201–213. [CrossRef]
42. Hale, E.T.; Yin, W.; Zhang, Y. A fixed-point continuation method for ℓ_1-regularized minimization with applications to compressed sensing. *CAAM TR07-07 Rice Univ.* **2007**, *43*, 44.
43. Beck, A.; Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.* **2009**, *2*, 183–202. [CrossRef]
44. Figueiredo, M.A.; Nowak, R.D.; Wright, S.J. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE J. Sel. Top. Signal Process.* **2007**, *1*, 586–597. [CrossRef]
45. Birgin, E.G.; Martínez, J.M.; Raydan, M. Nonmonotone spectral projected gradient methods on convex sets. *SIAM J. Optim.* **2000**, *10*, 1196–1211. [CrossRef]
46. Xiao, Y.; Wang, Q.; Hu, Q. Non-smooth equations based method for ℓ_1-norm problems with applications to compressed sensing. *Nonlinear Anal. Theory Methods Appl.* **2011**, *74*, 3570–3577. [CrossRef]
47. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef] [PubMed]