symmetry

Article

Descent Derivative-Free Method Involving Symmetric Rank-One Update for Solving Convex Constrained Nonlinear Monotone Equations and Application to Image Recovery

Aliyu Muhammed Awwal [1,2], Adamu Ishaku [1,2], Abubakar Sani Halilu [3], Predrag S. Stanimirović [4], Nuttapol Pakkaranang [5,*] and Bancha Panyanak [6,7,*]

[1] GSU-Mathematics for Innovative Research Group, Gombe State University (GSU), Gombe 760214, Nigeria
[2] Department of Mathematics, Faculty of Science, Gombe State University, Gombe 760214, Nigeria
[3] Department of Mathematics, School of Chemical Engineering and Physical Sciences, Lovely Professional University, Phagwara 144411, India
[4] Faculty of Science and Mathematics, University of Nis, Visegradksa 33, 18000 Nis, Serbia
[5] Mathematics and Computing Science Program, Faculty of Science and Technology, Phetchabun Rajabhat University, Phetchabun 67000, Thailand
[6] Research Group in Mathematics and Applied Mathematics, Department of Mathematics, Faculty of Science, Chiang Mai University, Chiang Mai 50200, Thailand
[7] Data Science Research Center, Department of Mathematics, Faculty of Science, Chiang Mai University, Chiang Mai 50200, Thailand
* Correspondence: nuttapol.pak@pcru.ac.th (N.P.); bancha.p@cmu.ac.th (B.P.)

check for updates

Citation: Awwal, A.M.; Ishaku, A.; Halilu, A.S.; Stanimirović, P.S.; Pakkaranang, N.; Panyanak, B. Descent Derivative-Free Method Involving Symmetric Rank-One Update for Solving Convex Constrained Nonlinear Monotone Equations and Application to Image Recovery. *Symmetry* **2022**, *14*, 2375. https://doi.org/10.3390/sym14112375

Academic Editor: Alexander Zaslavski

Received: 18 October 2022
Accepted: 6 November 2022
Published: 10 November 2022

Abstract: Many practical applications in applied sciences such as imaging, signal processing, and motion control can be reformulated into a system of nonlinear equations with or without constraints. In this paper, a new descent projection iterative algorithm for solving a nonlinear system of equations with convex constraints is proposed. The new approach is based on a modified symmetric rank-one updating formula. The search direction of the proposed algorithm mimics the behavior of a spectral conjugate gradient algorithm where the spectral parameter is determined so that the direction is sufficiently descent. Based on the assumption that the underlying function satisfies monotonicity and Lipschitz continuity, the convergence result of the proposed algorithm is discussed. Subsequently, the efficiency of the new method is revealed. As an application, the proposed algorithm is successfully implemented on image deblurring problem.

Keywords: algorithms for nonlinear problems; derivative-free method; nonlinear monotone equations; projection method; line search; image deblurring problem

MSC: 90C30; 90C06; 90C56

1. Preliminaries and Motivation

Consider the space $\mathbb{R}^n$ with a nonempty closed convex subset, A. The symbol $\|\cdot\|$ shall mean the Euclidean norm in the entire paper. Recall that a function V is said to be monotone and Lipschitz continuous if it satisfies the following inequalities

$$0 \leq (x-y)^T(V(x)-V(y)), \tag{1}$$

$$\|V(x)-V(y)\| \leq L\|x-y\|, \;\; L>0, \tag{2}$$

respectively. Let $V:\mathbb{R}^n \to \mathbb{R}^n$ be a monotone function, this paper is concerned with finding a vector $\hat{x} \in A$ that solves the following system of monotone nonlinear equations:

$$V(x)=0. \tag{3}$$

Problem (3) is of paramount importance due to its presence in various fields of applied sciences such as physics, engineering, signal processing, robotic motion control, image deblurring problems, etc. (see, [1–9]). Among the different classes of iterative algorithms for solving (3) such as Newton-like methods, quasi-Newton methods and trust-region methods (see, [10–15]); conjugate gradient (CG) methods (see, [16–20]) stand out. The CG methods are popular, especially for problems with large dimensions, due to the fascinating features like low memory demand, global convergence and simplicity.

Recently, the interesting projection map presented by Solodov and Svaiter [21] has made the CG method for (3) gain more attention. For instance, based on the popular CG_DESCENT method [22], Xiao and Zhu proposed a projection CG method for solving problem (3). The search direction produced by their algorithm is sufficiently descent independent of any line search technique. The sufficient descent property is an important property for any CG method, especially in the proof of its convergence result. However, Liu and Li [17] pointed out the possibility of the CG parameter in [23] being undefined at certain iteration and therefore, proposed another CG projection method. Furthermore, based on the Dai-Yuan CG method [24], Liu and Feng [25] developed an interesting spectral CG method with nice characteristics. Experimental results showed that their method is relatively efficient.

Another set of popular iterative methods for unconstrained optimization problems, $\min\{v(x) \in \mathbb{R} : u \in \mathbb{R}^n\}$, and (3) are the quasi-Newton methods. This class of methods compute their directions of search as $p_k = -M_k V(x_k)$, M_k is an approximation of the Hessian inverse of v at x_k. The matrix M_k is usually computed using different updating formula such as BFGS (Broyden–Fletcher–Goldfarb–Shanno) and SR1 (Symmetric rank-one formula) [26]. Unlike the CG methods, the quasi-Newton method may be unsuitable when the dimension of the problems is large. This inspires Zhang et al. [27] to incorporate the popular Polak–Ribière–Polyak parameter [28,29] into the BFGS updating formula and developed a three-term matrix-free iterative method for minimization problems without constraints. Motivated by the success of the Zhang et al.'s [27] approach, Awwal et al. [30] came up with a spectral CG-like method for (3) where a modified Perry CG parameter [31] is incorporated into a modified BFGS updating formula. The spectral parameter in their method is determined so that the sequence of the search directions is sufficiently descent.

The SR1 quasi-Newton method is a successful iterative method for minimization problems. However, not much has been done to explore its performance on the system of nonlinear Equation (3), perhaps due to the fact that the denominator of the SR1 updating formula could be zero at some point (detail is given in the next section). Motivated by the successes of the methods discussed thus far, in this paper, a new derivative-free spectral projection method for handling problem (3) is proposed with the aid of a modified SR1 updating formula. A few notable contributions of this work are enumerated below:

1. The search direction of the proposed method is set up based on the modified SR1 updating formula.
2. The direction of search has some nice features such as boundedness and so on.
3. The convergence result of the new method is established under some mild conditions.
4. The numerical performance of the new method is explored by implementing it on a set of test problems.
5. The proposed method is implemented on image deblurring problem.

The rest part of this paper is arranged as follows. The proposed method and its motivation, as well as its convergence analysis are expressed in Section 2. The efficiency of the new method is shown in Section 3 and the application is illustrated in Section 4. Finally, some remarks are highlighted in Section 5.

2. DFSR1 Method with Its Convergence Results

Let us start by stating the following assumption.

Assumption 1. *The point $\widehat{x} \in A$ satisfies $V(\widehat{x}) = 0$ and the function V satisfies* (1) *and* (2).

We briefly recall the quasi-Newton iterative methods, which produce $\{x_k\}$ using the following recurring formula:

$$x_{k+1} = x_k + \tau_k p_k, \;\; k = 0, 1, 2, \ldots, \tag{4}$$

τ_k is a step-size obtained via a suitable search line strategy and p_k is a search direction that is usually supposed to satisfy

$$V(x_k)^T p_k \leq -c\|V(x_k)\|^2, \;\; c > 0. \tag{5}$$

One of the popular quasi-Newton matrix updating formula, namely Symmetric rank-one (SR1), is given as follows:

$$M_k = \mu_k M_{k-1} + \frac{(s_{k-1} - M_{k-1}y_{k-1})(s_{k-1} - M_{k-1}y_{k-1})^T}{(s_{k-1} - M_{k-1}y_{k-1})^T y_{k-1}}, \tag{6}$$

$s_{k-1} = x_k - x_{k-1}$, $y_{k-1} = V(x_k) - V(x_{k-1})$ and the parameter $\mu_k > 0$ shall be determined. If $\mu_k = 1$ for all k, then the SR1 update (6) reduces to the classical one. By letting $M_{k-1} \equiv I$ (i.e., identity matrix), then (6) becomes

$$M_k = \mu_k I + \frac{(s_{k-1} - y_{k-1})(s_{k-1} - y_{k-1})^T}{(s_{k-1} - y_{k-1})^T y_{k-1}}. \tag{7}$$

The quasi-Newton search direction is defined as follows

$$p_k = -M_k V(x_k), \;\; k = 0, 1, 2, \ldots. \tag{8}$$

By substituting (7) into (8), we have

$$p_k = -\mu_k V(x_k) + \beta_k u_k, \tag{9}$$

where

$$\beta_k = \frac{-u_k^T V(x_k)}{y_{k-1}^T s_{k-1} - \|y_{k-1}\|^2}, \tag{10}$$

and $u_k = s_{k-1} - y_{k-1}$. This suggests that (9) imitates the actions of the classical spectral CG method. To ensure that the β_k (10) is well-defined, we must ensure that $\|y_{k-1}\|^2 \neq y_{k-1}^T s_{k-1}$. Now, defining $\overline{y}_{k-1} = V(x_k) - V(x_{k-1}) + ts_{k-1}$, $t > 0$, $s_{k-1} = x_k - x_{k-1}$ and using the Lipschitz continuity assumption on V, gives

$$\begin{aligned}\|\overline{y}_{k-1}\| &= \|V(x_k) - V(x_{k-1}) + t(x_k - x_{k-1})\| \\ &\leq \|V(x_k) - V(x_{k-1})\| + t\|x_k - x_{k-1}\| \\ &\leq L\|x_k - x_{k-1}\| + t\|x_k - x_{k-1}\| \\ &\leq (L + t)\|s_{k-1}\|.\end{aligned} \tag{11}$$

Additionally, by the monotonicity of V, we have

$$\begin{aligned}\|\overline{y}_{k-1}\|^2 &= (V(x_k) - V(x_{k-1}) + t(x_k - x_{k-1}))^T(V(x_k) - V(x_{k-1}) + t(x_k - x_{k-1})) \\ &= \|V(x_k) - V(x_{k-1})\|^2 + 2t(V(x_k) - V(x_{k-1}))^T(x_k - x_{k-1}) + t^2\|x_k - x_{k-1}\|^2 \\ &\geq \|V(x_k) - V(x_{k-1})\|^2 + t^2\|x_k - x_{k-1}\|^2 \\ &\geq t^2\|s_{k-1}\|^2.\end{aligned} \tag{12}$$

This, together with (11), gives

$$(t + L)^2\|s_{k-1}\|^2 \geq \|\overline{y}_{k-1}\|^2 \geq t^2\|s_{k-1}\|^2 > 0. \tag{13}$$

However, the monotonicity and Lipschitz continuity of V yield

$$\begin{aligned}\overline{y}_{k-1}^T s_{k-1} &= (V(x_k) - V(x_{k-1}) + t(x_k - x_{k-1}))^T (x_k - x_{k-1}) \\ &= (V(x_k) - V(x_{k-1}))^T (x_k - x_{k-1}) + t\|x_k - x_{k-1}\|^2 \\ &\geq t\|s_{k-1}\|^2 > 0,\end{aligned} \tag{14}$$

and

$$\begin{aligned}\overline{y}_{k-1}^T s_{k-1} &= (V(x_k) - V(x_{k-1}) + t(x_k - x_{k-1}))^T (x_k - x_{k-1}) \\ &= (V(x_k) - V(x_{k-1}))^T (x_k - x_{k-1}) + t\|x_k - x_{k-1}\|^2 \\ &= \|V(x_k) - V(x_{k-1})\| \|x_k - x_{k-1}\| + t\|x_k - x_{k-1}\|^2 \\ &\leq L\|x_k - x_{k-1}\|^2 + t\|x_k - x_{k-1}\|^2 \\ &= (t + L)\|s_{k-1}\|^2.\end{aligned} \tag{15}$$

Combining (14) and (15) gives

$$(t + L)\|s_{k-1}\|^2 \geq \overline{y}_{k-1}^T s_{k-1} \geq t\|s_{k-1}\|^2 > 0. \tag{16}$$

Therefore, it makes sense to redefine the parameter β_k as follows

$$\overline{\beta}_k = \frac{-\overline{u}_k^T V(x_k)}{\max\{\overline{y}_{k-1}^T s_{k-1},\ \|\overline{y}_{k-1}\|^2\}}, \tag{17}$$

where $\overline{u}_k = s_{k-1} - \overline{y}_{k-1}$.

Next, we determine the parameter μ_k so that (9) satisfies (5). Now, taking inner product of (9) and $V(x_k)$ where $\overline{\beta}_k$ is given by (17), we have

$$\begin{aligned}V(x_k)^T p_k &= -\mu_k\|V(x_k)\|^2 + \overline{\beta}_k V(x_k)^T \overline{u}_k \\ &= -\mu_k\|V(x_k)\|^2 - \frac{\overline{u}_k^T V(x_k)}{\max\{\overline{y}_{k-1}^T s_{k-1},\ \|\overline{y}_{k-1}\|^2\}} V(x_k)^T \overline{u}_k \\ &= -\left[\mu_k + \frac{(\overline{u}_k^T V(x_k))^2}{\max\{\overline{y}_{k-1}^T s_{k-1},\ \|\overline{y}_{k-1}\|^2\}\|V(x_k)\|^2}\right]\|V(x_k)\|^2.\end{aligned} \tag{18}$$

This means by taking

$$\mu_k \geq c - \frac{(\overline{u}_k^T V(x_k))^2}{\max\{\overline{y}_{k-1}^T s_{k-1},\ \|\overline{y}_{k-1}\|^2\}\|V(x_k)\|^2}, \quad c > 0, \tag{19}$$

we have

$$V(x_k)^T p_k \leq -c\|V(x_k)\|^2, \ \ c > 0,\ k = 0, 1, 2, \ldots. \tag{20}$$

Now, without loss of generality, we set the value of μ_k as

$$\mu_k = c - \frac{(\overline{u}_k^T V(x_k))^2}{\max\{\overline{y}_{k-1}^T s_{k-1},\ \|\overline{y}_{k-1}\|^2\}\|V(x_k)\|^2}. \tag{21}$$

Before we state the steps of the algorithm for the proposed method, the following definition is of great importance.

Definition 1. *Any point x in $\mathbb{R}^n$ can be projected onto the feasible set A via $P_A(x) = argmin\{\|x - y\| : y \in A\}$.*

The operator $P_A(\cdot)$ satisfies the following:

$$\|P_A(x) - y\| \leq \|x - y\|, \;\; \forall y \in A. \tag{22}$$

We now state the steps of the proposed Algorithm 1 for solving problem (3).

Algorithm 1: Derivative-free symmetric rank-one method (DFSR1).

```
Input  :Given $x_0 \in A$, $\rho \in (0,1)$, $t, \sigma, \kappa > 0$, $0 < \ell < 2$, $Tol \geq 0$.
Step 1: Determine $V(x_k)$. If $\|V(x_k)\| \leq Tol$, terminate.
Step 2: Calculate $p_0 = -V(x_0)$ and
    $p_k = -\max\{\mu_k, \; \lambda_k\}V(x_k) + \overline{\beta}_k \overline{u}_k,$    (23)
    where $\overline{u}_k = s_{k-1} - \overline{y}_{k-1}$, $\overline{\beta}_k$ and $\mu_k$ are given by (17) and (21), respectively and
    $\lambda_k = \frac{\|s_{k-1}\|^2}{\overline{y}_{k-1}^T s_{k-1}}.$    (24)
Step 3: Set $h_k = x_k + \tau_k p_k$, $\tau_k = \kappa\rho^i$ with $i$ being the least non-negative integer satisfying
    $-V(x_k + \kappa\rho^i p_k)^T p_k \geq \sigma\kappa\rho^i \|V(x_k + \kappa\rho^i p_k)\|^{1/q}\|p_k\|^2, \;\; q \geq 1.$    (25)
Step 4: If $\|V(h_k)\| = 0$, terminate the process. If not, compute the new point by
    $x_{k+1} = P_A\left[x_k - \ell\frac{V(h_k)^T(x_k - h_k)}{\|V(h_k)\|^2}V(h_k)\right].$    (26)
Step 5: Increase the counter, i.e., $k := k + 1$ and repeat from step 1.
```

Remark 1. *By the definition of* μ_k *in* (21)*, we cannot guarantee* $\mu_k > 0$ *for all k. Considering the modified Barzilai and Borwein spectral parameter* $\lambda_k = \frac{\|s_{k-1}\|^2}{\overline{y}_{k-1}s_{k-1}}$ *and using the inequalities* (13) *and* (16)*, we have*

$$\frac{1}{t} \geq \frac{\|s_{k-1}\|^2}{\overline{y}_{k-1}s_{k-1}} \geq \frac{1}{t+L} > 0. \tag{27}$$

Therefore, the use of $\max\{\overline{\mu}_k, \lambda_k\}$ *in the definition of the search direction* (23) *makes sense since* $\max\{\overline{\mu}_k, \; \lambda_k\} > 0$*, for all k.*

Remark 2. *The direction* (23) *satisfies* (20)*. If* $\max\{\mu_k, \; \lambda_k\} = \mu_k$*, then the result follows from* (18)*. Moreover, if* $\max\{\mu_k, \; \lambda_k\} = \lambda_k$*, then* $\lambda_k \geq \mu_k$ *and therefore from* (19) *we have*

$$\lambda_k \geq \mu_k \geq c - \frac{(\overline{u}_k^T V(x_k))^2}{\max\{\overline{y}_{k-1}^T s_{k-1}, \; \|\overline{y}_{k-1}\|^2\}\|V(x_k)\|^2},$$

which means (20) *holds.*

Remark 3. *The line search* (25) *is adopted from [32] and its well-definedness has been discussed in [33].*

Lemma 1. *Let the sequences* $\{x_k\}$ *and* $\{h_k\}$ *be produced by DFSR1. Suppose the function V is Lipschitzian and monotone, then the* $\lim_{k\to\infty} \|x_k - \widehat{x}\|$ *exists and*

$$\|p_k\| \leq \Gamma, \;\; \textit{for all} \;\; k \geq 0, \;\; \Gamma > 0. \tag{28}$$

$$\lim_{k\to\infty} \tau_k\|p_k\| = 0. \tag{29}$$

Proof. Let $\widehat{x} \in A$ such that $V(\widehat{x}) = 0$, then $V(\widehat{x})^T(x_k - \widehat{x}) = 0$. Therefore, it holds that

$$\begin{aligned} V(h_k)^T(x_k - \widehat{x}) &= V(h_k)^T(x_k - h_k + h_k - \widehat{x}) \\ &= V(h_k)^T(x_k - h_k) + V(h_k)^T(h_k - \widehat{x}) \\ &\geq V(h_k)^T(x_k - h_k) + V(\widehat{x}_k)^T(h_k - \widehat{x}) \\ &= V(h_k)^T(x_k - h_k), \end{aligned} \tag{30}$$

Note that the above inequality has been obtained using (1), that is, $V(h_k)^T(h_k - \widehat{x}) \geq V(\widehat{x})^T(h_k - \widehat{x})$. Next, since $0 < \ell < 2$, then by (22) and (26), we have

$$\begin{aligned} \|x_{k+1} - \widehat{x}\|^2 &= \left\| P_A\left(x_k - \ell \frac{V(h_k)^T(x_k - h_k)}{\|V(h_k)\|^2} V(h_k)\right) - \widehat{x} \right\| \\ &\leq \left\| x_k - \ell \frac{V(h_k)^T(x_k - h_k)}{\|V(h_k)\|^2} V(h_k) - \widehat{x} \right\| \\ &= \left\| (x_k - \widehat{x}) - \ell \frac{V(h_k)^T(x_k - h_k)}{\|V(h_k)\|^2} V(h_k) \right\|^2 \\ &\leq \|x_k - \widehat{x}\|^2 - 2\ell \frac{V(h_k)^T(x_k - h_k)}{\|V(h_k)\|^2} V(h_k)^T(x_k - \widehat{x}) + \ell^2 \frac{[V(h_k)^T(x_k - h_k)]^2}{\|V(h_k)\|^2} \\ &\leq \|x_k - \widehat{x}\|^2 - 2\ell \frac{V(h_k)^T(x_k - h_k)}{\|V(h_k)\|^2} V(h_k)^T(x_k - h_k) + \ell^2 \frac{[V(h_k)^T(x_k - h_k)]^2}{\|V(h_k)\|^2} \\ &= \|x_k - \widehat{x}\|^2 - \ell(2 - \ell) \frac{[V(h_k)^T(x_k - h_k)]^2}{\|V(h_k)\|^2} \qquad (31) \\ &\leq \|x_k - \widehat{x}\|^2. \qquad (32) \end{aligned}$$

Please observe that (32) gives the existence of $\lim\limits_{k\to\infty} \|x_k - \widehat{x}\|$.

Next, suppose the $\max\{\mu_k, \lambda_k\} = \mu_k$, then from (23), we have

$$\begin{aligned} \|p_k\| &= \| - \mu_k V(x_k) + \overline{\beta}_k \overline{u}_k\| \\ &\leq |\mu_k| \|V(x_k)\| + |\overline{\beta}_k| \|\overline{u}_k\| \\ &= \left| c - \frac{(\overline{u}_k^T V(x_k))^2}{\max\{\overline{y}_{k-1}^T s_{k-1}, \|\overline{y}_{k-1}\|^2\} \|V(x_k)\|^2} \right| \|V(x_k)\| + \left| \frac{-\overline{u}_k^T V(x_k)}{\max\{\overline{y}_{k-1}^T s_{k-1}, \|\overline{y}_{k-1}\|^2\}} \right| \|\overline{u}_k\| \\ &\leq \left(c + \frac{\|\overline{u}_k\|^2 \|V(x_k)\|^2}{\max\{\overline{y}_{k-1}^T s_{k-1}, \|\overline{y}_{k-1}\|^2\} \|V(x_k)\|^2} \right) \|V(x_k)\| + \frac{\|\overline{u}_k\| \|V(x_k)\|}{\max\{\overline{y}_{k-1}^T s_{k-1}, \|\overline{y}_{k-1}\|^2\}} \|\overline{u}_k\| \qquad (33) \\ &\leq \left(c + 2 \frac{\|\overline{u}_k\|^2}{\|\overline{y}_{k-1}\|^2} \right) \|V(x_k)\| \\ &\leq \left(c + 2 \frac{(1 + t + L)^2 \|s_{k-1}\|^2}{t^2 \|s_{k-1}\|^2} \right) \|V(x_k)\| \\ &\leq \left(c + 2 \frac{(1 + t + L)^2}{t^2} \right) \|V(x_k)\|. \end{aligned}$$

Since $\{\|x_k - \widehat{x}\|\}$ is a decreasing sequence, then $\|x_k - \widehat{x}\| \leq \|x_0 - \widehat{x}\|$ for all k. Setting $c_1 = L\|x_0 - \widehat{x}\|$, then by the Lipschitzian continuity of V, we have

$$\|V(x_k)\| = \|V(x_k) - V(\widehat{x})\| \leq L\|x_k - \widehat{x}\| \leq \ldots \leq L\|x_0 - \widehat{x}\| = c_1. \tag{34}$$

Combining (33) and (34) gives (28) with $\Gamma = \left(c + 2\frac{(1+t+L)^2}{t^2}\right)c_1$. Moreover, if $\max\{\mu_k, \lambda_k\} = \lambda_k$, then by (27) the result of (28) equally holds.

Furthermore, since $\lim\limits_{k\to\infty} \|x_k - \widehat{x}\|$ exists, then $\{x_k\}$ is bounded. Therefore by (28) and h_k in Step 3 of DFSR1, it means $\{h_k\}$ is bounded. Using similar argument as in (34) gives

$$\|V(h_k)\| \leq c_2, \;\; c_2 > 0. \tag{35}$$

Now, (25) and (31) give

$$\begin{aligned}\sigma^2\tau_k^4\|V(h_k)\|^{2/q}\|p_k\|^4 &\leq \tau_k^2[V(h_k)^T p_k]^2\\ &\leq \frac{\|V(h_k)\|^2}{\ell(2-\ell)}\Big(\|x_k-\widehat{x}\|^2-\|x_{k+1}-\widehat{x}\|^2\Big). \end{aligned} \tag{36}$$

(36) together with (35) gives

$$\begin{aligned}\sigma^2\tau_k^4\|p_k\|^4 &\leq \frac{\|V(h_k)\|^{2-2/q}}{\ell(2-\ell)}\Big(\|x_k-\widehat{x}\|^2-\|x_{k+1}-\widehat{x}\|^2\Big)\\ &\leq \frac{c_2^{2-2/q}}{\ell(2-\ell)}\Big(\|x_k-\widehat{x}\|^2-\|x_{k+1}-\widehat{x}\|^2\Big). \end{aligned} \tag{37}$$

Since $0<\ell<2$, then taking limits of both sides as $k\to\infty$ gives

$$\sigma^2 \lim_{k\to\infty}\tau_k^4\|p_k\|^4 = 0. \tag{38}$$

Hence,

$$\lim_{k\to\infty}\tau_k\|p_k\| = 0.$$

□

Theorem 1. *Let $\{x_k\}$ be a sequence produced by the DFSR1 and $\widehat{x}\in A$ such that $V(\widehat{x})=0$ then $\{x_k\}$ converges to $\widehat{x}$.*

Proof. Let the function V be Lipschitz continuous. First of all, we claim that

$$\lim_{k\to\infty}\inf\|V(x_k)\| = 0. \tag{39}$$

Assume (39) is false, then

$$\|V(x_k)\|\geq\omega,\;\; \omega>0,\;\; \text{holds}\;\; \forall\, k\geq 0. \tag{40}$$

If $\tau_k\neq\kappa$, by setting $h_k'=x_k+\tau_k'p_k$, then $\tau_k'=\rho^{-1}\tau_k$ will not satisfy (25). Therefore,

$$V(h_k')^T p_k+\sigma\tau_k'\|V(h_k')\|^{1/q}\|p_k\|^2>0. \tag{41}$$

Since (41) holds, then using (20) yields

$$\begin{aligned} c\|V(x_k)\|^2 &\leq -V(x_k)^T p_k\\ &< -V(x_k)^T p_k+V(h_k')^T p_k+\sigma\tau_k'\|V(h_k')\|^{1/q}\|p_k\|^2\\ &= (V(h_k')-V(x_k))^T p_k+\sigma\tau_k'\|V(h_k')\|^{1/q}\|p_k\|^2\\ &\leq L\tau_k'\|p_k\|^2+\sigma\tau_k'\|V(h_k')\|^{1/q}\|p_k\|^2\\ &= \rho^{-1}\tau_k(L+\sigma\|V(h_k')\|^{1/q})\|p_k\|^2\\ &\leq \rho^{-1}\tau_k(L+\sigma c_2^{1/q})\Gamma^2. \end{aligned} \tag{42}$$

(42) can be re-written as

$$\tau_k>\frac{\rho c\|V(x_k)\|^2}{\Gamma^2(L+\sigma c_2^{1/q})}\geq\frac{\rho c\omega^2}{\Gamma^2(L+\sigma c_2^{1/q})}. \tag{43}$$

Using Cauchy–Schwartz inequality on (20) gives

$$c\|V(x_k)\|^2\leq -V(x_k)^T p_k\leq\|V(x_k)\|\|p_k\|. \tag{44}$$

This together with (40) yields

$$\|p_k\| \geq c\omega. \tag{45}$$

Combining (43) and (45) gives

$$\tau_k\|p_k\| > \frac{\rho c\omega^2}{\Gamma^2(L + \sigma c_2^{1/q})} c\omega,$$

which contradicts (29). Hence, (39) must hold.

Lastly, by the continuity assumption on V, then $\{x_k\}$ possesses some limit point $\hat{x}$ for which $V(\hat{x}) = 0$. However, we have shown $\{\|x_k - \hat{x}\|\}$ converges, which means that the conclusion holds. □

3. Numerical Experiments

We now illustrate the numerical performance of the DFSR1 in comparison with: (i) PDY [25] and (ii) HCGP [34]. In the course of the experiment, we implement the three algorithms (that is, DFSR1, PDY and HCGP) on a collection of test problems (see, Appendix A). We vary the dimension of each test problem as 1000, 5000, 10,000, 50,000, and 100,000 and solve each with the starting points listed in Table 1.

Table 1. List of the starting point used for the experiment.

Starting Points (SP)	Values
x_1	$(\frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \cdots, \frac{1}{10})^T$
x_2	$(\frac{1}{2}, \frac{1}{2^2}, \frac{1}{2^3}, \cdots, \frac{1}{2^n})^T$
x_3	$(2, 2, 2, \ldots, 2)^T$
x_4	$(1, \frac{1}{2}, \frac{1}{3}, \cdots, \frac{1}{n})^T$
x_5	$(1 - \frac{1}{n}, 1 - \frac{2}{n}, 1 - \frac{3}{n}, \cdots, 0)^T$
x_6	$\text{rand}(0, 1)$

We implement these algorithms in MATLAB R2019b that runs on a PC with intel Core(TM) i5-8250u processor with 4 GB of RAM and CPU 1.60 GHz. We set the termination criterion for the iteration process as $\|V(x_k)\| \leq 10^{-6}$. We use the following parameters for Algorithm 1 (DFSR1): $\rho = 0.5$, $c = 0.1$, $t = 0.01$, $\sigma = 0.01$, $\kappa = 1$, $\ell = 1.99$. The parameters used in the implementation of PDY and HCGP are as given in [25,34].

We use three metrics, namely, ITER (number of iterations), FVAL (number of functions evaluation), and TIME (CPU time), to assess the performance of each algorithm under consideration. We report the ITER, FVAL and TIME recorded by each algorithm in Tables 2–9. In addition, we report the NORM (final norm of the $V(x_k)$ as the iteration process stops) to show whether an algorithm obtains a solution. Observing through the NORM in Tables 2–9, we note that all the algorithms in question satisfied the terminating criteria set. However, the efficiency and performance of each algorithm differ based on the ITER, FVAL and TIME. Perusing through the ITER, FVAL and TIME in Tables 2–9, we see that DFSR1 recorded smallest ITER, FVAL and TIME in most cases of the entire experiment. This means that DFSR1 was able to meet the terminating criteria faster and with the least ITER and FVAL for most of the problems. We present the summary of these information in Figures 1–3 with the aid of the popular Dolan and Moré performance profile [35]. This shows DFSR1 is efficient and outperforms the two existing competitors, that is, PDY [25] and HCGP [34].

Table 2. Numerical results of DFSR1, PDY and HCGP on Problem 1.

Problem 1	**DFSR1**					**PDY**				**HCGP**			
Dimension	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	$\times 1$	6	13	0.05181	2.2×10^{-9}	36	74	0.031651	3.49×10^{-7}	18	38	0.040198	6.82×10^{-7}
	$\times 2$	2	5	0.007194	4.44×10^{-16}	42	85	0.056619	8.84×10^{-7}	20	41	0.014543	2.79×10^{-7}
	$\times 3$	4	9	0.004168	0	46	94	0.024212	2.21×10^{-8}	20	41	0.007144	9.54×10^{-7}
	$\times 4$	5	12	0.003428	4.85×10^{-8}	40	82	0.023213	3.16×10^{-7}	17	36	0.005264	6.81×10^{-7}
	$\times 5$	1	3	0.003915	0	51	104	0.031001	9.89×10^{-7}	19	39	0.008887	7.54×10^{-7}
	$\times 6$	1	3	0.001159	0	48	98	0.031394	7.21×10^{-7}	19	39	0.008946	7.44×10^{-7}
5000	$\times 1$	7	15	0.015875	5.38×10^{-8}	27	56	0.055308	1.3×10^{-8}	19	39	0.042978	7.61×10^{-7}
	$\times 2$	2	5	0.010399	4.44×10^{-16}	42	85	0.11446	8.84×10^{-7}	20	41	0.018623	2.79×10^{-7}
	$\times 3$	4	9	0.010012	0	28	58	0.052897	4.05×10^{-7}	21	43	0.040893	5.3×10^{-7}
	$\times 4$	5	12	0.010918	4.85×10^{-8}	36	74	0.067928	3.44×10^{-7}	17	36	0.02057	6.82×10^{-7}
	$\times 5$	1	3	0.004772	0	51	104	0.14059	7.98×10^{-7}	19	40	0.030715	8.34×10^{-7}
	$\times 6$	1	3	0.003104	0	50	102	0.11267	7.91×10^{-7}	19	40	0.034105	8.28×10^{-7}
10,000	$\times 1$	6	13	0.023978	3.74×10^{-8}	29	60	0.10814	2.14×10^{-8}	18	38	0.05138	5.38×10^{-7}
	$\times 2$	2	5	0.008067	4.44×10^{-16}	42	85	0.1269	8.84×10^{-7}	20	41	0.054415	2.79×10^{-7}
	$\times 3$	4	9	0.013192	0	31	64	0.12693	1.18×10^{-7}	20	41	0.049781	7.49×10^{-7}
	$\times 4$	5	12	0.020775	4.85×10^{-8}	41	84	0.14847	1.1×10^{-7}	17	36	0.066033	6.82×10^{-7}
	$\times 5$	1	3	0.005923	0	47	95	0.19518	5.62×10^{-7}	20	41	0.054006	5.89×10^{-7}
	$\times 6$	1	3	0.006829	0	54	109	0.1683	5.66×10^{-7}	18	37	0.050708	5.87×10^{-7}
50,000	$\times 1$	4	9	0.078887	0	28	58	0.4178	8.48×10^{-8}	19	39	0.22635	6.01×10^{-7}
	$\times 2$	2	5	0.030988	4.44×10^{-16}	42	85	0.5385	8.84×10^{-7}	20	41	0.20855	2.79×10^{-7}
	$\times 3$	4	9	0.088267	0	44	90	0.67125	2.61×10^{-7}	20	42	0.26323	8.36×10^{-7}
	$\times 4$	5	12	0.07949	4.85×10^{-8}	45	92	0.66928	2.84×10^{-7}	17	36	0.22042	6.82×10^{-7}
	$\times 5$	1	3	0.022959	0	51	104	0.83104	6.27×10^{-7}	19	40	0.22258	6.57×10^{-7}
	$\times 6$	1	3	0.020248	0	46	94	0.72275	6.27×10^{-7}	17	36	0.2241	6.57×10^{-7}
100,000	$\times 1$	4	9	0.12576	0	21	44	0.74785	7.7×10^{-7}	18	37	0.38516	8.5×10^{-7}
	$\times 2$	2	5	0.05741	4.44×10^{-16}	42	85	1.0936	8.84×10^{-7}	20	41	0.43528	2.79×10^{-7}
	$\times 3$	4	9	0.13963	0	37	76	1.1562	4.43×10^{-8}	21	43	0.48012	5.91×10^{-7}
	$\times 4$	5	12	0.13249	4.85×10^{-8}	32	66	0.76896	7.38×10^{-7}	17	36	0.34743	6.82×10^{-7}
	$\times 5$	1	3	0.075883	0	51	104	1.4413	7.08×10^{-7}	19	40	0.45343	9.3×10^{-7}
	$\times 6$	1	3	0.036755	0	51	104	1.4801	7.07×10^{-7}	18	38	0.40378	9.29×10^{-7}

Table 3. Numerical results of DFSR1, PDY and HCGP on Problem 2.

Problem 2	**DFSR1**					**PDY**				**HCGP**			
Dimension	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	$\times 1$	2	5	0.002032	0	21	44	0.042925	7.52×10^{-7}	11	23	0.012052	7.5×10^{-7}
	$\times 2$	1	3	0.001357	0	19	40	0.019641	5.27×10^{-7}	9	20	0.008822	9.88×10^{-7}
	$\times 3$	1	3	0.000939	0	24	50	0.033042	8.19×10^{-7}	1	3	0.002222	0
	$\times 4$	1	3	0.001431	0	20	42	0.027811	5.35×10^{-7}	10	21	0.006196	8.65×10^{-7}
	$\times 5$	1	3	0.001253	0	23	48	0.007525	9.58×10^{-7}	12	25	0.00593	7.76×10^{-7}
	$\times 6$	1	3	0.000854	0	23	48	0.015592	9.65×10^{-7}	12	25	0.003991	7.92×10^{-7}
5000	$\times 1$	2	5	0.00407	0	22	46	0.031534	2.34×10^{-22}	11	24	0.016444	8.38×10^{-7}
	$\times 2$	1	3	0.003971	0	19	40	0.032489	5.27×10^{-7}	9	20	0.016042	9.88×10^{-7}
	$\times 3$	1	3	0.003208	0	26	54	0.046987	9.45×10^{-7}	1	3	0.002345	0
	$\times 4$	1	3	0.007746	0	20	42	0.03322	5.35×10^{-7}	10	21	0.009836	8.65×10^{-7}
	$\times 5$	1	3	0.003147	0	25	52	0.041525	5.36×10^{-7}	12	26	0.015802	8.68×10^{-7}
	$\times 6$	1	3	0.003627	0	26	54	0.039863	5.31×10^{-7}	12	26	0.024356	8.64×10^{-7}
10,000	$\times 1$	2	5	0.009787	0	22	46	0.06466	3.31×10^{-22}	12	25	0.028574	5.93×10^{-7}
	$\times 2$	1	3	0.004975	0	19	40	0.10797	5.27×10^{-7}	9	20	0.022977	9.88×10^{-7}
	$\times 3$	1	3	0.00518	0	27	56	0.078982	6.68×10^{-7}	1	3	0.004461	0
	$\times 4$	1	3	0.0054	0	20	42	0.066558	5.35×10^{-7}	10	21	0.021541	8.65×10^{-7}
	$\times 5$	1	3	0.005071	0	25	52	0.074131	7.58×10^{-7}	13	27	0.041391	6.14×10^{-7}
	$\times 6$	1	3	0.009867	0	25	52	0.07415	7.57×10^{-7}	13	27	0.032896	6.14×10^{-7}
50,000	$\times 1$	2	5	0.025063	0	24	50	0.52259	6.65×10^{-7}	12	26	0.19102	6.63×10^{-7}
	$\times 2$	1	3	0.04304	0	19	40	0.40302	5.27×10^{-7}	9	20	0.076766	9.88×10^{-7}
	$\times 3$	1	3	0.020766	0	27	56	0.37274	2.37×10^{-20}	1	3	0.013116	0
	$\times 4$	1	3	0.0234	0	20	42	0.20554	5.35×10^{-7}	10	21	0.090916	8.65×10^{-7}
	$\times 5$	1	3	0.016834	0	26	54	0.27215	8.48×10^{-7}	13	28	0.16845	6.86×10^{-7}
	$\times 6$	1	3	0.01886	0	26	54	0.55913	8.47×10^{-7}	13	28	0.12361	6.86×10^{-7}
100,000	$\times 1$	2	5	0.057742	0	19	40	0.81337	3.35×10^{-20}	12	26	0.21312	9.37×10^{-7}
	$\times 2$	1	3	0.038002	0	19	40	0.36734	5.27×10^{-7}	9	20	0.17563	9.88×10^{-7}
	$\times 3$	1	3	0.029571	0	34	70	1.0436	5.62×10^{-7}	1	3	0.019906	0
	$\times 4$	1	3	0.033132	0	20	42	0.8276	5.35×10^{-7}	10	21	0.24443	8.65×10^{-7}
	$\times 5$	1	3	0.048638	0	27	56	0.70974	9.24×10^{-7}	13	28	0.24368	9.7×10^{-7}
	$\times 6$	1	3	0.032773	0	27	56	0.94873	9.24×10^{-7}	13	28	0.31102	9.72×10^{-7}

Table 4. Numerical results of DFSR1, PDY and HCGP on Problem 3.

Problem 3		**DFSR1**				**PDY**				**HCGP**			
Dimension	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	×1	2	5	0.00159	0	1	3	0.023833	0	1	3	0.002252	0
	×2	2	5	0.00307	2.22×10^{-16}	1	3	0.00305	0	9	19	0.002664	7.45×10^{-7}
	×3	3	7	0.003173	0	1	3	0.001358	0	2	5	0.002427	0
	×4	3	7	0.002315	1.73×10^{-8}	5	12	0.002498	7.88×10^{-9}	11	23	0.004118	7.75×10^{-7}
	×5	4	9	0.003867	3.2×10^{-12}	29	59	0.010169	9.22×10^{-7}	12	25	0.003538	9.66×10^{-7}
	×6	4	9	0.003005	5.53×10^{-12}	14	30	0.009398	1.71×10^{-9}	12	25	0.003652	8.58×10^{-7}
5000	×1	2	5	0.003777	0	1	3	0.002194	0	1	3	0.002015	0
	×2	2	5	0.004794	2.22×10^{-16}	1	3	0.002093	0	9	19	0.012483	7.45×10^{-7}
	×3	3	7	0.005791	0	1	3	0.003394	0	2	5	0.014348	0
	×4	3	7	0.007381	1.85×10^{-8}	5	12	0.00686	6.9×10^{-9}	11	23	0.013458	7.88×10^{-7}
	×5	4	9	0.01028	7.25×10^{-12}	25	52	0.019655	6.94×10^{-7}	13	27	0.009709	5.4×10^{-7}
	×6	4	9	0.0095	4.8×10^{-12}	21	43	0.031217	4.46×10^{-7}	12	26	0.014854	9.65×10^{-7}
10,000	×1	2	5	0.007738	0	1	3	0.00811	0	1	3	0.003198	0
	×2	2	5	0.00719	2.22×10^{-16}	1	3	0.008884	0	9	19	0.017369	7.45×10^{-7}
	×3	3	7	0.010795	0	1	3	0.011714	0	2	5	0.005369	0
	×4	3	7	0.009558	1.87×10^{-8}	5	12	0.026699	6.79×10^{-9}	11	23	0.022454	7.9×10^{-7}
	×5	4	9	0.012112	1.03×10^{-11}	27	55	0.083228	6.22×10^{-7}	13	27	0.023442	7.63×10^{-7}
	×6	4	9	0.011263	1.07×10^{-11}	21	43	0.039402	8.19×10^{-7}	13	27	0.023976	7.33×10^{-7}
50,000	×1	2	5	0.021911	0	1	3	0.01112	0	1	3	0.009598	0
	×2	2	5	0.019317	2.22×10^{-16}	1	3	0.009912	0	9	19	0.056567	7.45×10^{-7}
	×3	3	7	0.07312	0	1	3	0.023374	0	2	5	0.016131	0
	×4	3	7	0.033954	1.88×10^{-8}	5	12	0.049476	6.69×10^{-9}	11	23	0.081633	7.92×10^{-7}
	×5	4	9	0.060339	2.3×10^{-11}	1	3	0.014165	0	13	28	0.12374	8.53×10^{-7}
	×6	4	9	0.044374	3.23×10^{-11}	1	3	0.028178	0	13	28	0.10853	8.87×10^{-7}
100,000	×1	2	5	0.044082	0	1	3	0.037395	0	1	3	0.016332	0
	×2	2	5	0.049941	2.22×10^{-16}	1	3	0.040274	0	9	19	0.11363	7.45×10^{-7}
	×3	3	7	0.072729	0	1	3	0.083691	0	2	5	0.04588	0
	×4	3	7	0.071052	1.88×10^{-8}	5	12	0.092381	6.68×10^{-9}	11	23	0.1511	7.92×10^{-7}
	×5	4	9	0.079648	3.25×10^{-11}	1	3	0.045013	0	14	29	0.24708	6.03×10^{-7}
	×6	4	9	0.14616	3.22×10^{-11}	1	3	0.024994	0	14	29	0.18329	5.74×10^{-7}

Table 5. Numerical results of DFSR1, PDY and HCGP on Problem 4.

Problem 4		**DFSR1**				**PDY**				**HCGP**			
Dimension	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	×1	18	38	0.007132	8.9×10^{-7}	6	14	0.037213	2.09×10^{-7}	8	17	0.00518	9.7×10^{-7}
	×2	13	28	0.005894	8.43×10^{-8}	34	70	0.024662	6.1×10^{-7}	11	24	0.010193	2.47×10^{-7}
	×3	20	42	0.007719	4.95×10^{-7}	6	14	0.006326	6.55×10^{-7}	6	13	0.002903	8.28×10^{-9}
	×4	10	22	0.004068	8.79×10^{-7}	33	68	0.061277	5.47×10^{-7}	10	22	0.007482	4.59×10^{-7}
	×5	8	18	0.004327	6.23×10^{-8}	48	98	0.057661	8.73×10^{-7}	18	38	0.009034	6.91×10^{-7}
	×6	10	22	0.005321	4.02×10^{-7}	49	100	0.078792	5.53×10^{-7}	13	27	0.006522	4.87×10^{-9}
5000	×1	19	40	0.042286	7.94×10^{-7}	6	14	0.027803	4.68×10^{-7}	8	18	0.024965	1.8×10^{-7}
	×2	15	32	0.038477	6.6×10^{-7}	35	72	0.13751	5.59×10^{-7}	11	24	0.023137	9.68×10^{-8}
	×3	21	44	0.05194	4.42×10^{-7}	7	16	0.038339	9.35×10^{-8}	6	13	0.013392	1.85×10^{-8}
	×4	12	26	0.024726	1.1×10^{-7}	36	74	0.13227	6.96×10^{-7}	14	30	0.034095	3.04×10^{-7}
	×5	8	18	0.025379	1.26×10^{-7}	34	70	0.1559	7.37×10^{-7}	13	27	0.017819	3.58×10^{-7}
	×6	8	18	0.01772	3.49×10^{-8}	43	88	0.27922	7.08×10^{-7}	15	32	0.033045	6.32×10^{-8}
10,000	×1	20	42	0.096094	4.48×10^{-7}	6	14	0.060341	6.62×10^{-7}	8	18	0.031653	2.55×10^{-7}
	×2	15	32	0.069529	1×10^{-7}	16	34	0.12798	8.16×10^{-7}	11	24	0.053742	4.33×10^{-7}
	×3	21	44	0.15177	6.25×10^{-7}	7	16	0.059352	1.36×10^{-7}	6	13	0.025402	2.62×10^{-8}
	×4	14	29	0.066288	3.87×10^{-7}	35	72	0.24323	5.62×10^{-7}	15	32	0.066142	1.92×10^{-7}
	×5	8	18	0.047045	1.76×10^{-7}	40	82	0.29859	6.2×10^{-7}	13	27	0.05961	5.05×10^{-7}
	×6	9	19	0.053913	3.79×10^{-7}	48	98	0.39773	8.52×10^{-7}	15	31	0.099451	1.68×10^{-7}
50,000	×1	21	44	0.36637	4×10^{-7}	7	16	0.18304	9.46×10^{-8}	8	18	0.10205	5.7×10^{-7}
	×2	17	35	0.2732	8.95×10^{-7}	28	58	0.70837	7.57×10^{-7}	19	39	0.23808	1.91×10^{-8}
	×3	22	46	0.41411	5.57×10^{-7}	8	18	0.19757	2.69×10^{-7}	6	13	0.098579	5.85×10^{-8}
	×4	12	26	0.25077	8.19×10^{-8}	26	54	0.59532	1.44×10^{-7}	15	32	0.26854	1.64×10^{-7}
	×5	8	18	0.13834	3.89×10^{-7}	37	76	0.93297	7.23×10^{-7}	13	28	0.1878	7.17×10^{-8}
	×6	9	19	0.15433	6.75×10^{-7}	48	98	1.3436	6.36×10^{-7}	16	34	0.31217	3.47×10^{-7}
100,000	×1	21	44	0.62111	5.66×10^{-7}	7	16	0.21426	8.58×10^{-7}	8	18	0.19648	8.06×10^{-7}
	×2	17	36	0.60434	1×10^{-7}	35	72	1.0326	7.61×10^{-7}	12	26	0.31084	5.07×10^{-8}
	×3	22	46	0.6102	7.88×10^{-7}	8	18	0.34764	3.81×10^{-7}	6	13	0.1247	8.28×10^{-8}
	×4	14	30	0.42957	9.3×10^{-8}	29	60	0.87375	5.25×10^{-7}	15	31	0.32466	1.28×10^{-9}
	×5	8	18	0.26303	5.49×10^{-7}	35	72	1.2043	4.43×10^{-7}	13	28	0.40534	1.01×10^{-7}
	×6	8	18	0.25233	1.83×10^{-7}	41	84	1.4854	6.97×10^{-7}	13	28	0.41283	9.35×10^{-8}

Table 6. Numerical results of DFSR1, PDY and HCGP on Problem 5.

Problem 5		**DFSR1**				**PDY**				**HCGP**			
Dimension	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	×1	1	3	0.002308	0	16	34	0.019027	8.17×10^{-7}	6	13	0.004866	3.87×10^{-7}
	×2	1	3	0.002296	2.22×10^{-16}	15	31	0.011457	7.97×10^{-7}	1	3	0.002889	3.14×10^{-16}
	×3	1	3	0.001798	0	18	38	0.011605	8.51×10^{-7}	1	3	0.001549	0
	×4	6	13	0.003828	0	17	36	0.012113	4.98×10^{-7}	9	20	0.008859	5.86×10^{-7}
	×5	61	123	0.075739	8.25×10^{-7}	19	40	0.032744	4.77×10^{-7}	14	30	0.014831	2.7×10^{-7}
	×6	60	121	0.026541	9.82×10^{-7}	19	40	0.042873	4.22×10^{-7}	15	31	0.009647	4.8×10^{-7}
5000	×1	1	3	0.007539	0	17	36	0.12849	6.85×10^{-7}	6	13	0.023056	8.66×10^{-7}
	×2	1	3	0.005604	2.22×10^{-16}	15	31	0.076005	7.97×10^{-7}	1	3	0.004773	3.14×10^{-16}
	×3	1	3	0.005712	0	20	42	0.058499	8.59×10^{-7}	1	3	0.004512	0
	×4	6	13	0.010185	0	17	36	0.048496	4.99×10^{-7}	7	15	0.015412	0
	×5	64	129	0.19584	8.14×10^{-7}	20	42	0.058435	4.02×10^{-7}	15	31	0.037637	1.23×10^{-7}
	×6	64	129	0.15052	7.9×10^{-7}	20	42	0.051701	3.98×10^{-7}	16	33	0.040346	7.64×10^{-7}
10,000	×1	1	3	0.020364	0	17	36	0.18164	9.69×10^{-7}	6	14	0.04793	4.59×10^{-7}
	×2	1	3	0.008206	2.22×10^{-16}	15	31	0.081092	7.97×10^{-7}	1	3	0.006498	3.14×10^{-16}
	×3	1	3	0.014966	0	21	44	0.28984	4.56×10^{-7}	1	3	0.005154	0
	×4	6	13	0.033961	0	17	36	0.1958	4.99×10^{-7}	11	23	0.047613	4.06×10^{-7}
	×5	65	131	0.40617	8.75×10^{-7}	20	42	0.097809	5.69×10^{-7}	15	31	0.068584	3.8×10^{-7}
	×6	65	131	0.36916	8.77×10^{-7}	20	42	0.09549	5.5×10^{-7}	14	29	0.058668	1.06×10^{-7}
50,000	×1	1	3	0.030717	0	18	38	0.47586	8.13×10^{-7}	7	15	0.16787	1.71×10^{-7}
	×2	1	3	0.024848	2.22×10^{-16}	15	31	0.53998	7.97×10^{-7}	1	3	0.017653	3.14×10^{-16}
	×3	1	3	0.039588	0	23	48	0.81555	9.93×10^{-7}	1	3	0.016797	0
	×4	6	13	0.11358	0	17	36	0.34906	4.99×10^{-7}	11	23	0.1745	7.52×10^{-7}
	×5	68	137	1.5711	8.55×10^{-7}	20	42	0.46878	7.76×10^{-7}	15	31	0.25982	7.56×10^{-7}
	×6	68	137	1.4454	8.59×10^{-7}	20	42	0.36411	7.74×10^{-7}	15	31	0.24661	2.61×10^{-7}
100,000	×1	1	3	0.052007	0	19	40	1.222	4.31×10^{-7}	7	15	0.19357	2.42×10^{-7}
	×2	1	3	0.046685	2.22×10^{-16}	15	31	0.50666	7.97×10^{-7}	1	3	0.039055	3.14×10^{-16}
	×3	1	3	0.066953	0	27	56	1.7894	4.72×10^{-7}	1	3	0.031785	0
	×4	6	13	0.29814	0	17	36	0.56445	4.99×10^{-7}	11	23	0.38384	7.66×10^{-7}
	×5	69	139	2.4612	9.17×10^{-7}	21	44	0.96169	4.11×10^{-7}	15	32	0.62553	3.75×10^{-7}
	×6	69	139	2.4641	9.1×10^{-7}	21	44	1.2645	4.12×10^{-7}	15	31	0.49338	8.85×10^{-8}

Table 7. Numerical results of DFSR1, PDY and HCGP on Problem 6.

Problem 6		**DFSR1**				**PDY**				**HCGP**			
Dimension	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	×1	44	89	0.030153	7.26×10^{-7}	139	280	0.081433	9.73×10^{-7}	66	134	0.02907	7.63×10^{-7}
	×2	33	68	0.024583	7.28×10^{-7}	187	376	0.087243	8.84×10^{-7}	36	74	0.015563	6.67×10^{-7}
	×3	47	96	0.033269	8.6×10^{-7}	227	456	0.10668	9.23×10^{-7}	55	112	0.021233	6.9×10^{-7}
	×4	43	88	0.02823	8.87×10^{-7}	218	438	0.099353	9.53×10^{-7}	42	86	0.041643	8.78×10^{-7}
	×5	52	106	0.032009	7.35×10^{-7}	233	468	0.24766	9.99×10^{-7}	43	88	0.023054	6.11×10^{-7}
	×6	65	132	0.10119	8.51×10^{-7}	189	380	0.12749	9.71×10^{-7}	58	118	0.031802	7.64×10^{-7}
5000	×1	47	96	0.18102	6.5×10^{-7}	188	378	0.70868	8.86×10^{-7}	81	164	0.17259	8.38×10^{-7}
	×2	33	68	0.11644	7.28×10^{-7}	187	376	0.66593	8.84×10^{-7}	36	74	0.078733	6.67×10^{-7}
	×3	47	96	0.17374	9.37×10^{-7}	277	556	1.0654	9.68×10^{-7}	58	118	0.16663	6.21×10^{-7}
	×4	39	80	0.15158	9.6×10^{-7}	218	438	0.77647	9.56×10^{-7}	42	86	0.1127	8.8×10^{-7}
	×5	55	112	0.32966	7.8×10^{-7}	181	364	0.76163	9.64×10^{-7}	91	184	0.2795	8.56×10^{-7}
	×6	72	146	0.27598	9.32×10^{-7}	200	402	0.80518	9.59×10^{-7}	62	126	0.15663	5.8×10^{-7}
10,000	×1	47	96	0.40311	8.96×10^{-7}	193	388	1.5151	9.37×10^{-7}	82	166	0.44199	7.28×10^{-7}
	×2	33	68	0.25323	7.28×10^{-7}	187	376	1.3113	8.84×10^{-7}	36	74	0.21537	6.67×10^{-7}
	×3	54	110	0.40319	7.71×10^{-7}	220	442	1.6456	9.39×10^{-7}	45	92	0.23366	9.4×10^{-7}
	×4	40	82	0.29423	8.61×10^{-7}	218	438	1.1552	9.56×10^{-7}	42	86	0.22799	8.8×10^{-7}
	×5	65	132	0.56744	9.45×10^{-7}	232	466	2.0015	9.66×10^{-7}	99	200	0.53545	6.52×10^{-7}
	×6	61	124	0.46567	7.27×10^{-7}	204	410	1.4589	9.71×10^{-7}	62	126	0.33344	7.97×10^{-7}
50,000	×1	35	71	1.083	9.99×10^{-7}	190	382	4.9906	9.44×10^{-7}	94	189	1.8118	9.9×10^{-7}
	×2	33	68	1.0053	7.28×10^{-7}	187	376	4.717	8.84×10^{-7}	36	74	0.67027	6.67×10^{-7}
	×3	33	68	0.82318	7.4×10^{-7}	233	468	5.453	9.9×10^{-7}	46	94	0.95285	8.25×10^{-7}
	×4	38	78	1.045	8.75×10^{-7}	218	438	4.8288	9.56×10^{-7}	42	86	0.78167	8.8×10^{-7}
	×5	66	134	1.9521	6.95×10^{-7}	238	478	5.8409	9.93×10^{-7}	105	212	2.1204	6.84×10^{-7}
	×6	75	152	2.031	8.46×10^{-7}	208	418	5.3067	9.27×10^{-7}	66	134	1.2361	5.7×10^{-7}
100,000	×1	46	94	2.2128	8.51×10^{-7}	197	396	8.3395	9.98×10^{-7}	92	186	3.3487	6.48×10^{-7}
	×2	33	68	1.6353	7.28×10^{-7}	187	376	7.4252	8.84×10^{-7}	36	74	1.2983	6.67×10^{-7}
	×3	53	108	2.2467	9.03×10^{-7}	184	370	7.5987	9.73×10^{-7}	46	94	1.5285	6.79×10^{-7}
	×4	35	72	1.4249	8.8×10^{-7}	218	438	9.1582	9.56×10^{-7}	42	86	1.5365	8.8×10^{-7}
	×5	59	120	2.6404	7.84×10^{-7}	257	516	11.1628	9.96×10^{-7}	130	262	5.2584	6.39×10^{-7}
	×6	84	170	3.6204	7.37×10^{-7}	206	414	9.6512	9.17×10^{-7}	66	134	2.1315	8.17×10^{-7}

Table 8. Numerical results of DFSR1, PDY and HCGP on Problem 7.

Problem 7		**DFSR1**				**PDY**				**HCGP**			
Dimension	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	×1	61	123	0.023673	9.91×10^{-7}	225	452	0.12358	9.68×10^{-7}	154	310	0.063765	9.18×10^{-7}
	×2	50	102	0.021414	7.3×10^{-7}	205	412	0.11853	9.84×10^{-7}	149	300	0.040999	8.68×10^{-7}
	×3	69	140	0.026574	7.38×10^{-7}	152	306	0.10356	9.82×10^{-7}	104	210	0.028425	9.8×10^{-7}
	×4	54	110	0.021371	9.54×10^{-7}	226	454	0.10959	9.59×10^{-7}	145	292	0.040416	8.83×10^{-7}
	×5	45	92	0.017501	9.38×10^{-7}	204	410	0.087927	9.77×10^{-7}	148	298	0.042045	9.49×10^{-7}
	×6	73	148	0.030156	8.61×10^{-7}	325	652	0.16361	9.68×10^{-7}	238	478	0.080769	9.94×10^{-7}
5000	×1	57	116	0.17496	9×10^{-7}	208	418	0.37694	9.98×10^{-7}	151	304	0.27362	9×10^{-7}
	×2	65	132	0.14268	9.6×10^{-7}	208	418	0.44897	9.66×10^{-7}	149	300	0.22391	8.85×10^{-7}
	×3	59	120	0.12642	9.47×10^{-7}	157	316	0.33535	9.77×10^{-7}	108	218	0.1743	8.38×10^{-7}
	×4	64	130	0.1561	8.9×10^{-7}	202	406	0.44525	9.74×10^{-7}	145	292	0.22877	9.89×10^{-7}
	×5	66	134	0.14988	8.6×10^{-7}	205	412	0.37069	9.48×10^{-7}	145	292	0.39989	9.42×10^{-7}
	×6	88	178	0.1901	8.22×10^{-7}	345	692	0.9217	9.68×10^{-7}	260	522	0.41205	9.17×10^{-7}
10,000	×1	51	103	0.31461	5.68×10^{-7}	219	440	1.2146	9.71×10^{-7}	150	302	0.61638	9.48×10^{-7}
	×2	55	112	0.29121	9.62×10^{-7}	200	402	1.1583	9.98×10^{-7}	145	292	0.63647	9.38×10^{-7}
	×3	76	154	0.41032	9.63×10^{-7}	148	298	0.78906	9.58×10^{-7}	108	218	0.54539	8.86×10^{-7}
	×4	66	134	0.35086	9.07×10^{-7}	222	446	1.2542	9.6×10^{-7}	142	286	0.63658	9.31×10^{-7}
	×5	56	114	0.30977	7.82×10^{-7}	197	396	1.1353	9.86×10^{-7}	145	292	0.64456	9.89×10^{-7}
	×6	86	174	0.5027	8.36×10^{-7}	348	698	2.0085	9.91×10^{-7}	268	538	1.2242	8.22×10^{-7}
50,000	×1	57	116	1.1288	9.51×10^{-7}	204	410	3.8161	9.76×10^{-7}	147	296	2.1916	9.65×10^{-7}
	×2	55	112	1.0816	9.99×10^{-7}	201	404	3.6254	9.87×10^{-7}	145	292	2.1226	9.5×10^{-7}
	×3	74	150	1.4655	7.6×10^{-7}	152	306	2.6166	9.9×10^{-7}	104	210	1.5508	8.79×10^{-7}
	×4	72	146	1.4134	9.83×10^{-7}	198	398	2.9667	9.96×10^{-7}	143	288	2.0497	9.18×10^{-7}
	×5	68	138	1.3541	8.71×10^{-7}	196	394	2.9459	9.62×10^{-7}	142	286	1.8212	9.97×10^{-7}
	×6	87	176	2.0557	7.86×10^{-7}	364	730	5.3453	9.81×10^{-7}	279	560	3.3756	9.43×10^{-7}
100,000	×1	65	132	2.8844	9.84×10^{-7}	210	422	6.9991	9.9×10^{-7}	147	296	3.6906	9.98×10^{-7}
	×2	65	132	2.6734	7.82×10^{-7}	215	432	6.9132	1×10^{-6}	146	294	3.7073	9.9×10^{-7}
	×3	70	142	2.6297	8.73×10^{-7}	155	312	5.2351	9.74×10^{-7}	104	210	2.7255	9.15×10^{-7}
	×4	61	124	2.2149	8.42×10^{-7}	211	424	7.0719	9.6×10^{-7}	142	286	3.5859	9.94×10^{-7}
	×5	69	140	2.5933	8×10^{-7}	212	426	6.8722	9.92×10^{-7}	146	294	3.6764	8.11×10^{-7}
	×6	102	206	3.9598	9.32×10^{-7}	374	750	11.9519	9.55×10^{-7}	288	578	7.2411	8.77×10^{-7}

Table 9. Numerical results of DFSR1, PDY and HCGP on Problem 8.

Problem 8		**DFSR1**				**PDY**				**HCGP**			
Dimension	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	×1	18	38	0.007958	9.14×10^{-7}	5	12	0.035715	6.7×10^{-7}	6	13	0.002637	7.39×10^{-8}
	×2	18	37	0.007423	4.41×10^{-7}	56	114	0.022696	4.91×10^{-7}	18	38	0.00669	2.4×10^{-7}
	×3	19	40	0.012079	5.7×10^{-7}	7	16	0.006145	3.73×10^{-7}	7	15	0.002588	3.49×10^{-8}
	×4	16	33	0.012515	7.78×10^{-7}	53	108	0.033416	7.74×10^{-7}	18	38	0.006571	2.09×10^{-7}
	×5	26	53	0.011984	8.23×10^{-7}	55	112	0.051453	9.95×10^{-7}	20	42	0.009647	3.18×10^{-7}
	×6	18	37	0.008318	7.92×10^{-7}	65	132	0.038824	7.53×10^{-7}	20	42	0.007383	5.83×10^{-7}
5000	×1	19	40	0.026268	8.15×10^{-7}	6	14	0.017025	9.57×10^{-8}	6	13	0.016436	1.65×10^{-7}
	×2	18	38	0.042577	7.68×10^{-7}	44	90	0.10485	2.79×10^{-7}	17	36	0.039178	2.35×10^{-7}
	×3	20	42	0.090895	5.09×10^{-7}	7	16	0.013239	8.34×10^{-7}	7	15	0.014297	7.81×10^{-8}
	×4	49	100	0.15176	9.77×10^{-7}	43	88	0.16041	9.7×10^{-7}	20	42	0.050196	2.3×10^{-7}
	×5	18	38	0.042496	5.14×10^{-7}	45	92	0.12203	8.5×10^{-7}	23	47	0.049253	5.73×10^{-7}
	×6	19	40	0.048408	9.02×10^{-7}	77	156	0.2765	7.67×10^{-7}	21	44	0.050957	2.39×10^{-7}
10,000	×1	20	42	0.10739	4.6×10^{-7}	6	14	0.044056	1.35×10^{-7}	6	13	0.021673	2.34×10^{-7}
	×2	34	70	0.14954	9.02×10^{-7}	51	104	0.20259	3.85×10^{-7}	19	40	0.077444	2.08×10^{-7}
	×3	20	42	0.10281	7.2×10^{-7}	8	18	0.036341	7.53×10^{-8}	7	15	0.022822	1.1×10^{-7}
	×4	60	121	0.33672	8.6×10^{-7}	37	76	0.29144	5.71×10^{-7}	20	42	0.14996	6.11×10^{-7}
	×5	17	36	0.085697	8.71×10^{-7}	44	90	0.21218	9.11×10^{-7}	17	36	0.071395	3.08×10^{-7}
	×6	20	42	0.1003	5.94×10^{-7}	87	176	0.40799	7.95×10^{-7}	22	46	0.12034	2.88×10^{-7}
50,000	×1	21	44	0.52854	4.11×10^{-7}	6	14	0.095905	3.03×10^{-7}	6	13	0.081578	5.22×10^{-7}
	×2	32	66	0.75814	8.53×10^{-7}	46	94	1.1108	9.07×10^{-7}	17	36	0.30372	2.58×10^{-7}
	×3	21	44	0.39951	6.42×10^{-7}	10	22	0.21688	1.43×10^{-7}	7	15	0.093075	2.47×10^{-7}
	×4	24	50	0.45453	6.22×10^{-7}	56	114	1.1769	9.66×10^{-7}	21	44	0.34356	7.02×10^{-7}
	×5	18	38	0.36846	8.95×10^{-7}	46	94	0.84367	7.11×10^{-7}	21	44	0.42353	2.59×10^{-7}
	×6	21	44	0.49905	9.61×10^{-7}	96	194	1.8368	6.46×10^{-7}	21	44	0.38195	5.7×10^{-7}
100,000	×1	21	44	0.81437	5.81×10^{-7}	7	16	0.22941	3.45×10^{-7}	6	13	0.14466	7.39×10^{-7}
	×2	34	70	1.3899	9.14×10^{-7}	53	108	2.1976	8.8×10^{-7}	19	40	0.66113	2.42×10^{-7}
	×3	21	44	0.75336	9.08×10^{-7}	12	26	0.46681	8.81×10^{-7}	7	15	0.20851	3.49×10^{-7}
	×4	18	38	0.68936	4.98×10^{-7}	58	118	2.407	5.95×10^{-7}	19	40	0.67528	1.64×10^{-7}
	×5	18	37	0.5929	8.63×10^{-7}	45	92	1.6446	8.52×10^{-7}	22	46	0.64709	8.69×10^{-8}
	×6	22	46	0.77784	7.98×10^{-7}	108	218	3.8486	3.84×10^{-7}	23	47	0.89078	6.62×10^{-7}

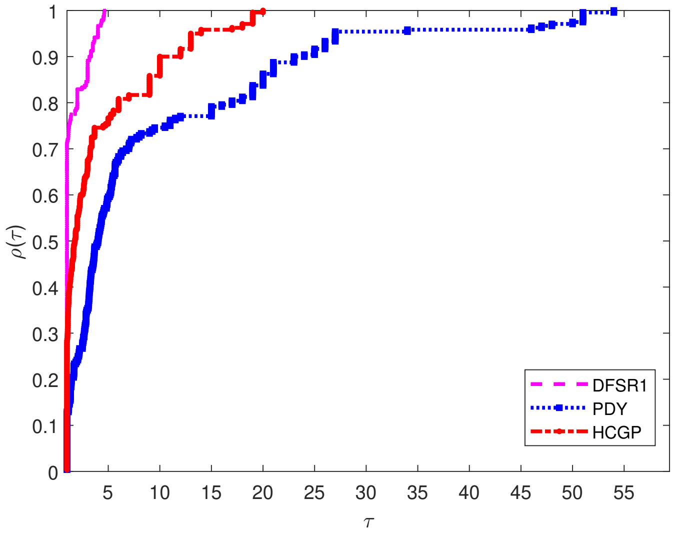

Figure 1. Performance profile for ITER.

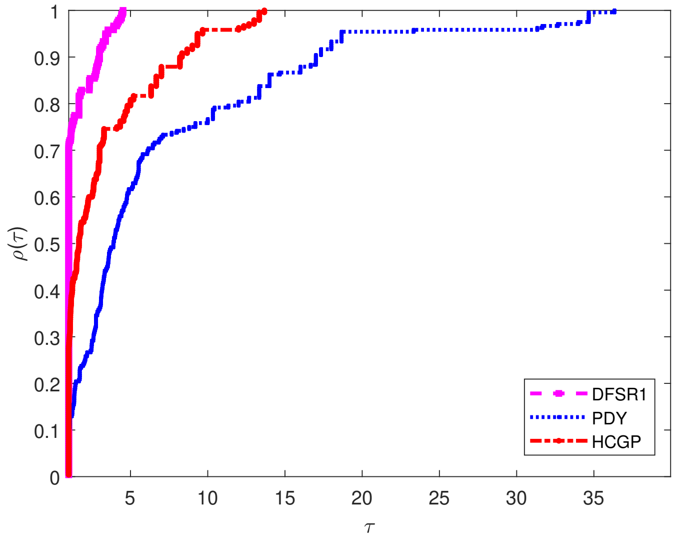

Figure 2. Performance profile for FVAL.

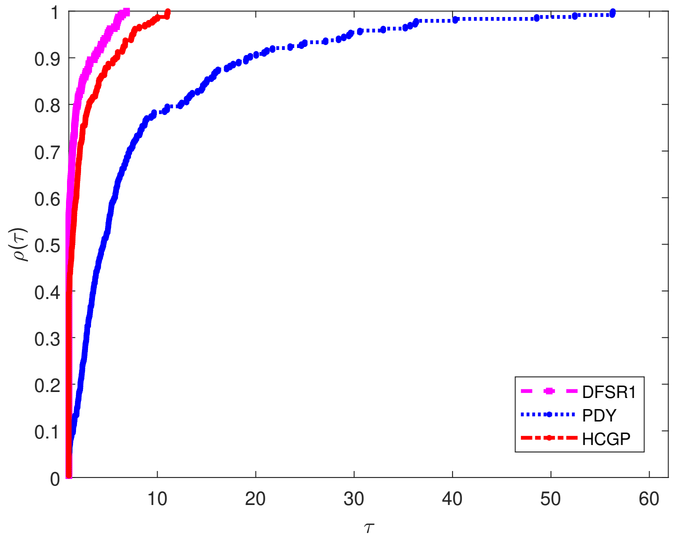

Figure 3. Performance profile for TIME.

4. Application in Image Deblurring

Consider the convex unconstrained minimization problem

$$\min_{x\in\mathbb{R}^n} v(x), \quad v(x)=\frac{1}{2}\|y-Qx\|_2^2+\eta\|x\|_1, \quad \eta>0 \tag{46}$$

$y\in\mathbb{R}^k$ is an observation and the linear function Q is in $\mathbb{R}^{k\times n}$ ($k << n$). The above model (46) has an interesting application in image deblurring.

The ℓ_1 regularization model (46) has attracted a great deal of attention and some researchers have developed different iterative algorithms for handling it (see, [36–38]). By reformulating (46) into a bound-constrained quadratic program, Figueiredo et al. [39] developed gradient-based projection procedure. The reformulation in [39] is as follows: let the vector x be split into two parts, i.e., $x=a-b$, where $a_i=(x_i)_+$, $b_i=(-x_i)_+$ for all $i=1,2,\dots,n$, and $(.)_+=\max\{0, .\}$, then (46) is translated into

$$\min_{w\geq 0}\frac{1}{2}w^TZw+r^Tw, \tag{47}$$

$$w=[a \;\; b]^T, \quad r=\eta e_{2n}+\left[-Q^Ty \quad Q^Ty\right]^T \text{ and } \quad Z=\begin{bmatrix} Q^TQ & -Q^TQ \\ -Q^TQ & Q^TQ\end{bmatrix}.$$

Taking it further, Xiao et al. [40] showed that (47) is equivalent to the following system of nonlinear equation

$$V(w)=\min\{w, \, Zw+r\}=0, \tag{48}$$

the "min" means componentwise minimum. Finally, while Pang [41] showed that V satisfies (2), Xiao et al. [40] proved that it also satisfies (1).

One of the immediate advantages associated with the system of monotone nonlinear equations (48) over (46) is that derivative-free algorithms can be employed to efficiently solve it. One of the derivative-free algorithms applied on (48) to recover some disturbed signals as well as restore some blurred images is the SGCS algorithm [40] by Xiao et al. A few years later, based on the popular CG_DESCENT method [42], Xiao and Zhu [23] came up with another derivative-free algorithm (CGD), which performs better than the SGCS algorithm on signal recovery problem. Following this trend, we apply the DFSR1 to recover some blurred color images by solving the model (48). We compare the performance of the DFSR1 with the CGD of Xiao and Zhu [23]. Details of the image deblurring experiment are given as follows.

We coded the two algorithms in MATLAB R2019b and carried out the image deblurring experiment on a PC with an intel Core(TM) i5-8250u processor with 4 GB of RAM and CPU 1.60 GHz. We run each algorithm from the same initial point $x_0=Q^Ty$ and terminate it if $\frac{|v_k-v_{k-1}|}{|v_{k-1}|}<10^{-5}$, is satisfied, where v_k is the merit function evaluation at x_k, with $v(x)=\frac{1}{2}\|y-Qx\|_2^2+\eta\|x\|_1$. The parameters used for both DFSR1 and CGD in this experiment are taken from [23]. We used four color images for the experiment and the following metrics are used for assessment: (i) ITER, (ii) TIME, (iii) SNR (signal-to-noise-ratio), and (iv) SSIM (structural similarity index measure).

The two algorithms, that is, DFSR1 and CGD, successfully recovered all the four images after being blurred, albeit with different rapidity and level of quality. Original and blurred images, along with images restored by each of the considered algorithms are presented in Figure 4. Furthermore, the data on ITER, TIME, SNR and SSIM are presented in Table 10. In addition, the size of each image used in the experiment are given. The values in Table 10 revealed that DFSR1 recorded less ITER and TIME than CGD in the experiment. As for the quality of the restoration, the SNR and SSIM recorded by DFSR1 reveal that the quality of recoveries by DFSR1 are slightly better than those by the CGD. This shows that the proposed DFSR1 is efficient and applicable.

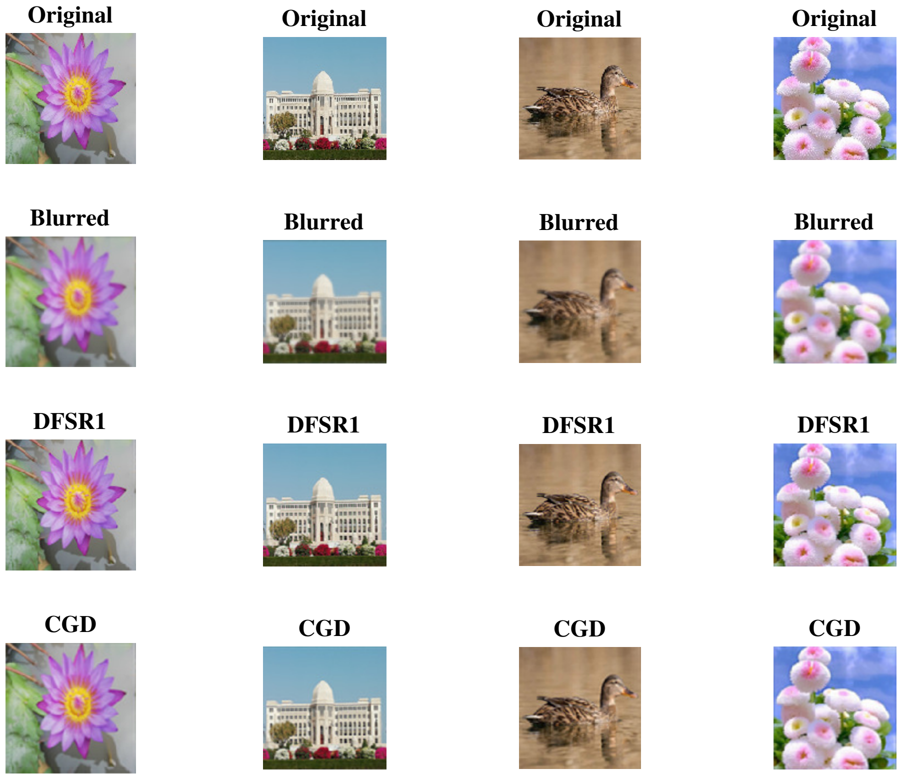

Figure 4. First row (original images), second row (blurred images), third and last rows (recovered images by methods DFSR1 and CGD, respectively).

Table 10. Test results for DFSR1 and CGD the colored image deblurring experiment.

		DFSR1				CGD			
Image	**Size**	**ITER**	**TIME(s)**	**SNR**	**SSIM**	**ITER**	**TIME(s)**	**SNR**	**SSIM**
1st Image	256 × 256	43	1.84	26.87	0.91	44	1.95	25.55	0.89
2nd Image	256 × 256	52	3.53	19.23	0.81	52	3.52	17.94	0.74
3rd Image	256 × 256	31	2.61	19.82	0.86	35	3.05	18.45	0.81
4th Image	256 × 256	20	1.77	25.42	0.83	23	2.01	24.75	0.81

5. Conclusions

Based on a modified symmetric rank-one matrix updating formula and a quasi-Newton search direction, a new descent direction of searching has been proposed. The proposed algorithm, in this paper, incorporated the popular Solodov and Svaiter projection techniques and a suitable line search used for obtaining the step length. The boundedness of the new direction and the theoretical analysis of the algorithm has been established based on some mild assumptions. The efficiency of the new method as well as its applicability have been illustrated. As future research, the proposed DFSR1 can be modified and implemented on nonlinear least squares problems [43,44].

Author Contributions: Conceptualization, A.M.A. and A.I.; methodology, A.M.A.; software, A.M.A. and N.P.; validation, A.M.A., A.I., A.S.H., P.S.S., N.P. and B.P.; formal analysis, A.M.A., P.S.S. and N.P.; investigation, A.M.A., A.I., A.S.H., P.S.S., N.P. and B.P.; resources, A.M.A., N.P. and B.P.; data curation, A.M.A., A.I., A.S.H. and P.S.S.; writing-original draft preparation, A.M.A. and A.S.H.; writing-review and editing, A.M.A., A.I., A.S.H., P.S.S., N.P. and B.P.; visualization, A.M.A., A.I., A.S.H. and P.S.S.; supervision, A.M.A. and P.S.S.; project administration, A.M.A., N.P. and B.P.; funding acquisition, N.P. and B.P. All authors have read and agreed to the published version of the manuscript.

Funding: The fifth author was partially supported by Phetchabun Rajabhat University. The sixth author was funded by Chiang Mai University and the NSRF via the Program Management Unit for Human Resources & Institutional Development, Research and Innovation (grant number B05F640183).

Acknowledgments: The authors would like to thank the anonymous referees and editor for reading this paper carefully, providing valuable suggestions and comments, which grately improved the final version.

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

Appendix A

We use the following monotone nonlinear equation for the second experiments where $V(x) = (v_1(x), v_2(x), \ldots, v_n(x))^T$, and $x = (x_1, x_2, \ldots, x_n)^T$.

Problem A1 ([45]).

$$\begin{aligned} v_1(x) &= e^{x_1} - 1 \\ v_i(x) &= e^{x_i} + x_{i-1} - 1, \;\; i = 1, 2, \cdots, n-1, \end{aligned}$$

where $A = \mathbb{R}^n_+$.

Problem A2 ([46]).

$$v_i(x) = 2x_i - \sin|x_i|, \;\; i = 1, 2, \cdots, n,$$

where $A = \mathbb{R}^n_+$.

Problem A3 ([47]).

$$v_i(x) = e^{x_i} - 1, \;\; i = 1, 2, \cdots, n,$$

where $A = \mathbb{R}^n_+$.

Problem A4 ([48]).

$$v_i(x) = e^{x_i^2} + \frac{3}{2}\sin(2x_i) - 1, \;\; i = 1, 2, \cdots, n,$$

where $A = \mathbb{R}^n_+$.

Problem A5 ([49]).

$$v_i(x) = x_i - \sin(|x_i - 1|), \;\; i = 1, 2, \cdots, n-1,$$

where $A = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i \leq n, \; x_i \geq -1, \; i = 1, 2, \cdots, n\}$.

Problem A6.

$$\begin{aligned} v_1(x) &= 2x_1 - x_2 + e^{x_1} - 1, \\ v_i(x) &= -x_{i-1} + 2x_i - x_{i+1} + e^{x_i} - 1, \;\; i = 2, \cdots, n-1, \\ v_n(x) &= -x_{n-1} + 2x_n + e^{x_n} - 1, \end{aligned}$$

where $A = \mathbb{R}^n_+$.

Problem A7 ([25]).

$$\begin{aligned} v_1(x) &= \frac{5}{2}x_1 + x_2 - 1, \\ v_i(x) &= x_{i-1} + \frac{5}{2}x_i + x_{i+1} - 1, \;\; i = 2, \cdots, n-1, \\ v_n(x) &= x_{n-1} + \frac{5}{2}x_n - 1, \end{aligned}$$

where $A = \mathbb{R}^n_+$.

Problem A8.

$$\begin{aligned} v_1(x) &= x_1 + \sin x_1 - 1 \\ v_i(x) &= -x_{i-1} + 2x_i + \sin x_i - 1,\ i = 2, \ldots, n-1 \\ v_n(x) &= x_n + \sin x_n - 1. \end{aligned}$$

where $A = \mathbb{R}^n_+$.

References

1. Sulaiman, I.M.; Awwal, A.M.; Malik, M.; Pakkaranang, N.; Panyanak, B. A Derivative-Free MZPRP Projection Method for Convex Constrained Nonlinear Equations and Its Application in Compressive Sensing. *Mathematics* **2022**, *10*, 2884. [CrossRef]
2. Al-Sawalha, M.M.; Agarwal, R.P.; Shah, R.; Ababneh, O.Y.; Weera, W. A reliable way to deal with fractional-order equations that describe the unsteady flow of a polytropic gas. *Mathematics* **2022**, *10*, 2293. [CrossRef]
3. Mohammad, H.; Awwal, A.M. Globally convergent diagonal Polak–Ribière–Polyak like algorithm for nonlinear equations. *Numer. Algorithms* **2022**, 1–20. [CrossRef]
4. Al-Sawalha, M.M.; Alshehry, A.S.; Nonlaopon, K.; Shah, R.; Ababneh, O.Y. Approximate analytical solution of time-fractional vibration equation via reliable numerical algorithm. *AIMS Math.* **2022**, *7*, 19739–19757. [CrossRef]
5. Halilu, A.S.; Majumder, A.; Waziri, M.Y.; Ahmed, K.; Awwal, A.M. Motion control of the two joint planar robotic manipulators through accelerated Dai–Liao method for solving system of nonlinear equations. *Eng. Comput.* **2021**, *39*, 1802–1840. [CrossRef]
6. Awwal, A.M.; Kumam, P.; Sitthithakerngkiet, K.; Bakoji, A.M.; Halilu, A.S.; Sulaiman, I.M. Derivative-free method based on DFP updating formula for solving convex constrained nonlinear monotone equations and application. *AIMS Math.* **2021**, *6*, 8792–8814. [CrossRef]
7. Halilu, A.S.; Majumder, A.; Waziri, M.Y.; Awwal, A.M.; Ahmed, K. On solving double direction methods for convex constrained monotone nonlinear equations with image restoration. *Comput. Appl. Math.* **2021**, *40*, 1–27. [CrossRef]
8. Al-Sawalha, M.M.; Khan, A.; Ababneh, O.; Botmart, T. Fractional view analysis of Kersten-Krasil'shchik coupled KdV-mKdV systems with non-singular kernel derivatives. *AIMS Math.* **2022**, *7*, 18334–18359. [CrossRef]
9. Al-Sawalha, M.M.; Alshehry, A.S.; Nonlaopon, K.; Shah, R.; Ababneh, O.Y. Fractional view analysis of delay differential equations via numerical method. *AIMS Math.* **2022**, *7*, 20510–20523. [CrossRef]
10. Wan, Z.; Chen, Y.; Huang, S.; Feng, D. A modified nonmonotone BFGS algorithm for solving smooth nonlinear equations. *Optim. Lett.* **2014**, *8*, 1845–1860. [CrossRef]
11. Leong, W.J.; Hassan, M.A.; Waziri, M.Y. A matrix-free quasi-Newton method for solving large-scale nonlinear systems. *Comput. Math. Appl.* **2011**, *62*, 2354–2363. [CrossRef]
12. Mohammad, H.; Waziri, M.Y. On Broyden-like update via some quadratures for solving nonlinear systems of equations. *Turk. J. Math.* **2015**, *39*, 335–345. [CrossRef]
13. Nocedal, J.; Wright, S. *Numerical Optimization*; Springer Science & Business Media: New York, NY, USA, 2006.
14. Ahookhosh, M.; Esmaeili, H.; Kimiaei, M. An effective trust-region-based approach for symmetric nonlinear systems. *Int. J. Comput. Math.* **2013**, *90*, 671–690. [CrossRef]
15. Yuan, Y.X. Recent advances in trust region algorithms. *Math. Program.* **2015**, *151*, 249–281. [CrossRef]
16. Cheng, W.; Xiao, Y.; Hu, Q. A family of derivative-free conjugate gradient methods for large-scale nonlinear systems of equations. *J. Comput. Appl. Math.* **2009**, *224*, 11–19. [CrossRef]
17. Liu, J.K.; Li, S.J. A projection method for convex constrained monotone nonlinear equations with applications. *Comput. Math. Appl.* **2015**, *70*, 2442–2453. [CrossRef]
18. Waziri, M.Y.; Muhammad, H.U.; Halilu, A.S.; Ahmed, K. Modified matrix-free methods for solving system of nonlinear equations. *Optimization* **2021**, 70, 2321 –2340. [CrossRef]
19. Awwal, A.M.; Kumam, P.; Abubakar, A.B. Spectral modified Polak–Ribiére–Polyak projection conjugate gradient method for solving monotone systems of nonlinear equations. *Appl. Math. Comput.* **2019**, *362*, 124514. [CrossRef]
20. Abubakar, A.B.; Kumam, P.; Awwal, A.M.; Thounthong, P. A modified self-adaptive conjugate gradient method for solving convex constrained monotone nonlinear equations for signal recovery problems. *Mathematics* **2019**, *7*, 693. [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. Hager, W.W.; Zhang, H. Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent. *ACM Trans. Math. Softw. (TOMS)* **2006**, *32*, 113–137. [CrossRef]
23. 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]
24. Dai, Y.; Yuan, Y. A nonlinear conjugate gradient method with a strong global convergence property. *SIAM J. Optim.* **1999**, *10*, 177–182. [CrossRef]
25. Liu, J.; Feng, Y. A derivative-free iterative method for nonlinear monotone equations with convex constraints. *Numer. Algorithms* **2019**, *82*, 245–262. [CrossRef]

26. Sun, W.; Yuan, Y.X. *Optimization Theory and Methods: Nonlinear Programming*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2006; Volume 1.
27. Zhang, L.; Zhou, W.; Li, D.H. A descent modified Polak–Ribière–Polyak conjugate gradient method and its global convergence. *IMA J. Numer. Anal.* **2006**, *26*, 629–640. [CrossRef]
28. Pola, E.; Ribiere, G. Note sur la convergence de methodes de directions conjugées. *Rev. Fr. Informat Rech. Oper. Annee* **1969**, *16*, 35–43.
29. Polyak, B.T. The conjugate gradient method in extremal problems. *USSR Comput. Math. Math. Phys.* **1969**, *9*, 94–112. [CrossRef]
30. Awwal, A.M.; Kumam, P.; Mohammad, H.; Watthayu, W.; Abubakar, A. A Perry-type derivative-free algorithm for solving nonlinear system of equations and minimizing ℓ_1 regularized problem. *Optimization* **2020**, *70*, 1231–1259. [CrossRef]
31. Perry, A. A modified conjugate gradient algorithm. *Oper. Res.* **1978**, *26*, 1073–1078. [CrossRef]
32. Awwal, A.M.; Wang, L.; Kumam, P.; Mohammad, H.; Watthayu, W. A projection Hestenes–Stiefel method with spectral parameter for nonlinear monotone equations and signal processing. *Math. Comput. Appl.* **2020**, *25*, 27. [CrossRef]
33. Awwal, A.M.; Wang, L.; Kumam, P.; Mohammad, H. A two-step spectral gradient projection method for system of nonlinear monotone equations and image deblurring problems. *Symmetry* **2020**, *12*, 874. [CrossRef]
34. Koorapetse, M.; Kaelo, P. An efficient hybrid conjugate gradient-based projection method for convex constrained nonlinear monotone equations. *J. Interdiscip. Math.* **2019**, *22*, 1031–1050. [CrossRef]
35. Dolan, E.D.; Moré, J.J. Benchmarking optimization software with performance profiles. *Math. Program. Ser.* **2002**, *91*, 201–213. [CrossRef]
36. Figueiredo, M.A.; Nowak, R.D. An EM algorithm for wavelet-based image restoration. *IEEE Trans. Image Process.* **2003**, *12*, 906–916. [CrossRef] [PubMed]
37. De Mol, C.; Defrise, M. A note on wavelet-based inversion algorithms. *Contemp. Math.* **2002**, *313*, 85–96.
38. Yang, J.; Yin, W.; Zhang, Y.; Wang, Y. A fast algorithm for edge-preserving variational multichannel image restoration. *SIAM J. Imaging Sci.* **2009**, *2*, 569–592. [CrossRef]
39. Figueiredo, M.; 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]
40. 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]
41. Pang, J. Inexact Newton methods for the nonlinear complementarity problem. *Math. Program.* **1986**, *36*, 54–71. [CrossRef]
42. 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]
43. Yahaya, M.M.; Kumam, P.; Awwal, A.M.; Aji, S. Alternative structured spectral gradient algorithms for solving nonlinear least-squares problems. *Heliyon* **2021**, *7*, e07499. [CrossRef]
44. Awwal, A.M.; Kumam, P.; Wang, L.; Yahaya, M.M.; Mohammad, H. On the Barzilai–Borwein gradient methods with structured secant equation for nonlinear least squares problems. *Optim. Methods Softw.* **2020**, 1–20. [CrossRef]
45. La Cruz, W.; Martínez, J.M.; Raydan, M. Spectral residual method without gradient information for solving large-scale nonlinear systems: Theory and experiments. *Citeseer* **2004**. Available online: http://kuainasi.ciens.ucv.ve/mraydan/download_papers/TechRep.pdf (accessed on 17 October 2022).
46. La Cruz, W. A spectral algorithm for large-scale systems of nonlinear monotone equations. *Numer. Algorithms* **2017**, *76*, 1109–1130. [CrossRef]
47. Zhou, W.; Shen, D. An Inexact PRP Conjugate Gradient Method for Symmetric Nonlinear Equations. *Numer. Funct. Anal. Optim.* **2014**, *35*, 370–388. [CrossRef]
48. Gao, P.; He, C.; Liu, Y. An adaptive family of projection methods for constrained monotone nonlinear equations with applications. *Appl. Math. Comput.* **2019**, *359*, 1–16. [CrossRef]
49. Yu, G.; Niu, S.; Ma, J. Multivariate spectral gradient projection method for nonlinear monotone equations with convex constraints. *J. Ind. Manag. Optim.* **2013**, *9*, 117–129. [CrossRef]