Mathematical and Computational Applications

MDPI

Article

A Projection Hestenes–Stiefel Method with Spectral Parameter for Nonlinear Monotone Equations and Signal Processing

Aliyu Muhammed Awwal [1,2], Lin Wang [3,*], Poom Kumam [1,4], Hassan Mohammad [5] and Wiboonsak Watthayu [4,*]

1 Fixed Point Theory and Applications Research Group, Theoretical and Computational Science Center, Science Laboratory Building, Faculty of Science, King Mongkut's University of Technology Thonburi, 126 Pracha-Uthit Road, Bang Mod, Thung Khru, Bangkok 10140, Thailand; aliyumagsu@gmail.com (A.M.A.); poom.kum@kmutt.ac.th (P.K.)
2 Department of Mathematics, Faculty of Science, Gombe State University, Gombe 760214, Nigeria
3 Office of Science and Research, Yunnan University of Finance and Economics, Kunming 650221, Yunnan, China
4 Department of Mathematics, Faculty of Science, King Mongkut's University of Technology Thonburi, 126 Pracha-Uthit Road, Bang Mod, Thung Khru, Bangkok 10140, Thailand
5 Department of Mathematical Sciences, Faculty of Physical Sciences, Bayero University, Kano 700241, Nigeria; hmuhd.mth@buk.edu.ng
* Correspondence: Wl64mail@aliyun.com (L.W.); wiboonsak.wat@kmutt.ac.th (W.W.)

Received: 27 March 2020; Accepted: 20 April 2020; Published: 1 May 2020

Abstract: A number of practical problems in science and engineering can be converted into a system of nonlinear equations and therefore, it is imperative to develop efficient methods for solving such equations. Due to their nice convergence properties and low storage requirements, conjugate gradient methods are considered among the most efficient for solving large-scale nonlinear equations. In this paper, a modified conjugate gradient method is proposed based on a projection technique and a suitable line search strategy. The proposed method is matrix-free and its sequence of search directions satisfies sufficient descent condition. Under the assumption that the underlying function is monotone and Lipschitzian continuous, the global convergence of the proposed method is established. The method is applied to solve some benchmark monotone nonlinear equations and also extended to solve ℓ_1-norm regularized problems to reconstruct a sparse signal in compressive sensing. Numerical comparison with some existing methods shows that the proposed method is competitive, efficient and promising.

Keywords: conjugate gradient method; nonlinear monotone equations; projection method; line search; signal processing

MSC: 65K05; 90C06; 90C56; 92C55

1. Introduction

Let $\mathbb{R}^n$ be an $n-$dimensional Euclidean space with inner product $\langle \cdot, \cdot \rangle$ and norm $\| \cdot \|$. Suppose that D is a nonempty closed convex subset of $\mathbb{R}^n$. We denote by $\mathbb{R}^n_+$ the set $\{(x_1, x_2, ..., x_n)^T \in \mathbb{R}^n \mid x_i \geq 0,\ i = 1, 2, ..., n\}$. In this paper, we consider the problem of finding a point $\hat{x}$ in the set D for which

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

It is interesting to note that nonlinear equations in the form of (1) has various background and applications in science and engineering, such as the first-order necessary condition of the unconstrained convex optimization problems [1], the ℓ_1-norm problem arising from compressing sensing [2], chemical equilibrium systems and optimal power flow equations [3], etc.

The classical methods for solving (1) include Newton's method, quasi-Newton methods and inexact-Newton methods (see, Chapters 3 and 5 in [4], Chapter 11 in [5]). Unfortunately, these methods are not suitable for solving large-scale problems because they require solving systems of linear equations using the Jacobian matrix or its approximation at each iteration. As a result, several matrix-free methods for solving nonlinear systems of equations are proposed (see, e.g., [6–10] among others). On the other hand, Bellavia et al. [11] proposed an affine scaling trust-region iterative algorithm for bound-constrained systems of nonlinear equations. Their algorithm is based on the classical trust-region Newton method for unconstrained nonlinear equations and it possesses global and local fast convergence properties. They provided preliminary numerical experiments to show the efficiency of the method. Based on the nonmonotonic interior backtracking line search technique, Zhu [12] proposed an affine scaling trust-region method for nonlinear equality systems subject to bounds on variables. Under some standard assumptions, he proved the global convergence as well as the fast rate of local convergence of the method and reported some numerical experiments to demonstrate its effectiveness. In [13], Ahookhosh et al. presented a trust-region algorithm to solve symmetric nonlinear system of equations. Their algorithm combines an effective adaptive trust-region radius and a non-monotone strategy. They established the global convergence of the algorithm without the non-degeneracy assumption of the exact Jacobian and presented some numerical comparison to show its performance. For more detail on trust-region methods for nonlinear equations, the reader may refer to the survey paper by Yuan [14].

Among numerous methods for solving large-scale general unconstrained optimization problems, conjugate gradient methods (CG) are popular because they are simple to implement and require low storage with good convergence properties. Based on this, a quite number of researchers have made effort to extend some of the classical conjugate gradient methods to solve nonlinear monotone equations with convex constraints. For example, Xiao and Zhu [15] combined the conjugate gradient method of Hager and Zhang [16] and the projection technique of Solodov and Svaiter [17] to solve constrained system of monotone nonlinear equations. Their numerical experiments show that the method works well and its convergence analysis was established under some reasonable assumptions. The modified Perry conjugate gradient method [18] for general unconstrained optimization was extended by Dai et al. [19] to solve the unconstrained version of problem (1). The numerical performance of their method was compared with the methods in [20,21] and the results show their method stands out. Based on the popular Broyden–Fletcher–Goldfarb–Shanno (BFGS) updating formula for solving general unconstrained optimization, Gao et al. [22] recently proposed two adaptive projection algorithms to solve the system of monotone nonlinear equations with convex constraints. The sequence of the search directions generated by their algorithms satisfy sufficient descent property and under the assumption that the underlying function is Lipschitzian continuous, the convergence analysis of the method was established. Recently, different interesting results on spectral and conjugate gradient methods for addressing unconstrained minimization problems and nonlinear equations were presented in [23–32] and the references therein.

Inspired by the contributions discussed above, in this paper, we extend a modified Hestenes–Stiefel-like proposed by Amini et al. [33] to solve system of monotone nonlinear equations with convex constraints. We built the proposed algorithm by modifying the search direction of Amini et al. [33] and combining it with the projection technique of Solodov and Svaiter [17]. We proved the global convergence of the algorithm using a more general line search strategy that is mostly utilized in literature. Additional contribution of this paper is the application of the proposed algorithm to solve signal processing problem arising from compressive sensing. We present numerical experiments to demonstrate the efficiency of the proposed method and also establish the convergence analysis under standard assumptions. We organize the rest of this paper as follows. In Section 2, we present the algorithm and its global convergence. In Section 3, we present the numerical experiments and give the conclusions in Section 4.

2. Proposed Algorithm for Monotone Equations and Its Convergence Analysis

Definition 1. *Let $x, y \in \mathbb{R}^n$, a function $F : \mathbb{R}^n \to \mathbb{R}^n$ is said to be*

(i) monotone if

$$0 \leq \langle F(x) - F(y),\ x - y\rangle. \tag{2}$$

(ii) Lipschitzian continuous if there exists $L > 0$ such that

$$\|F(x) - F(y)\| \leq L\|x - y\|. \tag{3}$$

We begin by recalling the well-known Hestenes–Stiefel (HS) conjugate gradient method for solving unconstrained optimization problem of the form

$$\min\{f(x) : x \in \mathbb{R}^n\}, \tag{4}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function and bounded from below. HS CG method updates its sequence of iterates using the following recursive formula:

$$x_{k+1} = x_k + \alpha_k d_k, \tag{5}$$

where $\alpha_k > 0$ is a suitable stepsize. The search direction d_k is defined by

$$d_k = \begin{cases} -F_k, & \text{if } k = 0, \\ -F_k + \beta_k^{HS} d_{k-1}, & \text{if } k > 0, \end{cases} \tag{6}$$

where the CG parameter is given by

$$\beta_k^{HS} = \frac{\langle F_k,\ y_{k-1}\rangle}{\langle y_{k-1},\ d_{k-1}\rangle},$$

and $y_{k-1} = F_k - F_{k-1}$ and F_k denotes the gradient of f at x_k.

Definition 2. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be differentiable at $x \in \mathbb{R}^n$ and $d \in \mathbb{R}^n$ satisfies*

$$\langle F,\ d\rangle < 0, \tag{7}$$

where F denotes the gradient of f, then d is called a descent direction of f at x.

It is well known that (7) is a crucial property for iterative methods (5) to be globally convergent. However, the HS CG method does not guarantee (7), and therefore not globally convergent for general nonlinear functions.

The HS-CG method has enjoyed some forms of modifications in order to improve its convergence properties and/or its numerical performance. Recently, Amini et al. [33] studied some modified HS CG for solving (4). Based on the work of Narushima et al. [34] and Dai and Kou [35], Amini et al. proposed a modified HS CG method with the search direction defined as follows

$$d_k = \begin{cases} -F_k, & \text{if } k = 0, \text{ or } \overline{\beta}_k^{MHS} \leq 0 \\ -F_k + \overline{\beta}_k^{MHS} d_{k-1}, & \text{if } k > 0, \end{cases} \tag{8}$$

where

$$\overline{\beta}_k^{MHS} = \frac{\langle F_k,\, y_{k-1}\rangle}{\langle y_{k-1},\, d_{k-1}\rangle}\theta_k - \eta\left(\frac{\|y_{k-1}\|\theta_k}{\langle y_{k-1},\, d_{k-1}\rangle}\right)^2\langle F_k,\, d_{k-1}\rangle, \tag{9}$$

and $\theta_k = 1 - \langle F_k,\, d_{k-1}\rangle^2/\|F_k\|^2\|d_{k-1}\|^2$. They showed that if $\langle y_{k-1},\, d_{k-1}\rangle \neq 0$ and $\eta > 1/4$, then the search direction d_k satisfies the sufficient descent condition (7). The natural question that comes to mind is, can we modified (9) in such a way that these two restrictions are removed and still retain the nice properties associated with (8)?

In this paper we provide answer to the above question. Let $w_{k+1} = x_k + \alpha_k d_k$, we define the proposed search direction as follows $d_0 = -F(x_0)$ and

$$d_k = -v_k F(x_k) + \max\{\beta_k^{PMHS},\, 0\}d_{k-1}, \quad k = 1, 2, ..., \tag{10}$$

where

$$\beta_k^{PMHS} = \frac{\langle F(x_k),\, d_{k-1}\rangle}{\|d_{k-1}\|^2} - \frac{\|\gamma_{k-1}\|^2}{\langle \gamma_{k-1},\, d_{k-1}\rangle^2}\langle F(x_k),\, d_{k-1}\rangle, \tag{11}$$

$$v_k = \frac{\|s_{k-1}\|^2}{\langle \gamma_{k-1},\, s_{k-1}\rangle}, \ \ \gamma_{k-1} = F(w_k) - F(x_{k-1}) + a s_{k-1}, \ \ s_{k-1} = w_k - x_{k-1}, \ \ a > 0. \tag{12}$$

The incorporation of the spectral parameter v_k in the definition of the search direction is to improve the numerical performance of the proposed algorithm. Next, we describe projection operator which is usually used in iterative algorithms for solving problems such as fixed point problem, variational inequality problem, and so on. Let $x \in \mathbb{R}^n$ and define an operator $P_D : \mathbb{R}^n \to D$ by $P_D(x) = \text{argmin}\{\|x - y\| : y \in D\}$. The operator P_D is called a projection onto the feasible set D and it enjoys the nonexpansive property, that is, $\|P_D(x) - P_D(y)\| \leq \|x - y\|, \ \forall \ x, y \in \mathbb{R}^n$. If $y \in D$, then $P_D(y) = y$ and therefore, we have

$$\|P_D(x) - y\| \leq \|x - y\|. \tag{13}$$

We now state the steps of the proposed algorithm in Algorithm 1.

Algorithm 1: Modified Hestene–Stiefel with spectral parameter (HSS).

Input: Given $x_0 \in D$, $0 < \kappa \leq 1$, $Tol > 0$, $a > 0$, $\sigma, \varrho \in (0, 1)$. Set $k = 0$.

Step 1: Compute $F(x_k)$. If $\|F(x_k)\| \leq Tol$, stop, otherwise go to step 2.

Step 2: Compute the search direction as follows. If $k = 0$, $\ d_k = -F(x_k)$.
Else

$$d_k = -v_k F(x_k) + \max\{\beta_k^{PMHS},\, 0\}d_{k-1},$$

$$\beta_k^{PMHS} = \frac{\langle F(x_k),\, d_{k-1}\rangle}{\|d_{k-1}\|^2} - \frac{\|\gamma_{k-1}\|^2}{\langle \gamma_{k-1},\, d_{k-1}\rangle^2}\langle F(x_k),\, d_{k-1}\rangle,$$

$$v_k = \frac{\|s_{k-1}\|^2}{\langle \gamma_{k-1},\, s_{k-1}\rangle}, \ \ \gamma_{k-1} = F(w_k) - F(x_{k-1}) + a s_{k-1}, \ \ s_{k-1} = w_k - x_{k-1}.$$

Step 3: Determine the stepsize $\alpha_k = \kappa\varrho^i$ where i is the smallest nonnegative integer such that

$$-\langle F(x_k + \kappa\varrho^i d_k),\, d_k\rangle \geq \sigma\kappa\varrho^i\|d_k\|^2\|F(x_k + \kappa\varrho^i d_k)\|^{1/r}, \ \ r \geq 1, \tag{14}$$

Step 4: Set $w_{k+1} = x_k + \alpha_k d_k$. If $w_{k+1} \in D$ and $\|F(w_{k+1})\| \leq Tol$ then stop. Else compute the next iterate using

$$x_{k+1} = P_D\left[x_k - \frac{\langle F(w_{k+1}),\, x_k - w_{k+1}\rangle}{\|F(w_{k+1})\|^2}F(w_{k+1})\right]. \tag{15}$$

Step 5: Set $k := k + 1$ and repeat the process from Step 1.

Remark 1. *The line search defined by (14) is more general than that of [36,37]. When $r = 1$, then the line search (14) reduces to the line search in [36] and when r is sufficiently large enough, it reduces to the line search in [37].*

Assumption 1. *Throughout this paper, we assume the following*

(i) The solution set of problem (1) is nonempty.
(ii) The function $F : \mathbb{R}^n \to \mathbb{R}^n$ satisfies (2) and (3).

The following lemma shows that the proposed search direction is well-defined and satisfies (7) independent of line search strategy.

Lemma 1. *The parameters β_k^{PMHS} and v_k defined by (11) and (12), respectively, are well-defined. In addition, $\forall\, k \geq 0$, the search direction d_k defined by (10) satisfies*

$$\langle F(x_k),\, d_k\rangle \leq -t\|F_k\|^2, \;\; t > 0. \tag{16}$$

Proof of Lemma 1. From the monotonicity of F, we have $\langle F(w_k) - F(x_{k-1}),\, w_k - x_{k-1}\rangle \geq 0$. This implies that

$$\langle \gamma_{k-1},\, s_{k-1}\rangle \geq a\|s_{k-1}\|^2 > 0. \tag{17}$$

By the definition of s_{k-1} and w_k in Steps 2 and 4 of Algorithm 1, we have $s_{k-1} = w_k - x_{k-1} = \alpha_{k-1}d_{k-1}$. Therefore, it holds

$$\langle \gamma_{k-1},\, d_{k-1}\rangle \geq a\alpha_{k-1}\|d_{k-1}\|^2 > 0. \tag{18}$$

By (3) we have

$$\langle \gamma_{k-1},\, s_{k-1}\rangle = \langle F(w_k) - F(x_{k-1}),\, s_{k-1}\rangle + a\|s_{k-1}\|^2 \leq (L+a)\|s_{k-1}\|^2. \tag{19}$$

So (17) and (19) give

$$\frac{1}{L+a} \leq v_k \leq \frac{1}{a}. \tag{20}$$

Hence v_k is well-defined and so is β_k^{PMHS} by (18).

Now taking the inner product of the search direction defined by (10) with F_k, for $k = 0$, it follows that $\langle F(x_k),\, d_k\rangle \leq -\|F(x_k)\|^2$, that is $t = 1$. Suppose $k > 0$ and $\beta_k^{PMHS} \leq 0$, we get

$$\langle F(x_k),\, d_k\rangle = -v_k\|F(x_k)\|^2 \leq -\frac{1}{L+a}\|F(x_k)\|^2.$$

Again, suppose $k > 0$ and $\beta_k^{PMHS} > 0$, we have

$$\begin{aligned}
\langle F(x_k),\, d_k\rangle &= -v_k\|F(x_k)\|^2 + \beta_k^{PMHS}\langle F(x_k),\, d_{k-1}\rangle \\
&= -v_k\|F(x_k)\|^2 + \left[\frac{\langle F(x_k),\, d_{k-1}\rangle}{\|d_{k-1}\|^2} - \frac{\|\gamma_{k-1}\|^2}{\langle \gamma_{k-1},\, d_{k-1}\rangle^2}\langle F(x_k),\, d_{k-1}\rangle\right]\langle F(x_k),\, d_{k-1}\rangle \\
&\leq -v_k\|F(x_k)\|^2 + \frac{\langle F(x_k),\, d_{k-1}\rangle^2}{\|d_{k-1}\|^2} - \frac{\|\gamma_{k-1}\|^2}{\|\gamma_{k-1}\|^2\|d_{k-1}\|^2}\langle F(x_k),\, d_{k-1}\rangle^2 \\
&\leq -\frac{1}{L+a}\|F(x_k)\|^2 + \frac{\langle F(x_k),\, d_{k-1}\rangle^2}{\|d_{k-1}\|^2} - \frac{\langle F(x_k),\, d_{k-1}\rangle^2}{\|d_{k-1}\|^2} \\
&= -\frac{1}{L+a}\|F(x_k)\|^2.
\end{aligned}$$

The two inequalities follow from Cauchy–Schwarz inequality and (20) respectively. Hence, the desired result holds. □

Lemma 2. *Let the sequence of iterates $\{x_k\}$ and the search direction $\{d_k\}$ be generated by Algorithm 1, then there always exists a step-size α_k satisfying the line search defined by (14) for any $k \geq 0$.*

Proof of Lemma 2. Suppose on the contrary that there exists some k_0 such that for any $i = 0, 1, 2, \ldots$, the line search (14) does not hold, that is

$$-\langle F(x_{k_0} + \kappa\varrho^i d_{k_0}),\ d_{k_0}\rangle < \sigma\kappa\varrho^i \|d_{k_0}\|^2 \|F(x_{k_0} + \kappa\varrho^i d_{k_0})\|^{1/r}. \tag{21}$$

By the fact that F is continuous, $\varrho \in (0, 1)$ and $\{\|d_k\|\}$ is bounded for all k (see Lemma 3 below), letting $i \to \infty$ yields

$$\langle F(x_{k_0}),\ d_{k_0}\rangle \geq 0. \tag{22}$$

It is clear that the inequality (22) contradicts (16). Hence the line search (14) is well-defined. □

Lemma 3. *Let Assumption 1 hold and $\hat{x}$ be a solution of problem (1). Suppose the sequences $\{w_k\}$, $\{x_k\}$ and $\{d_k\}$ are generated by Algorithm 1. Then the following hold*

(i) $\{x_k\}$ and $\{\|F(x_k)\|\}$ are bounded and $\lim_{k\to\infty} \|x_k - \hat{x}\|$ exists.
(ii) The sequence of the search direction $\{\|d_k\|\}$ is bounded.
(iii) $\{w_k\}$ and $\{\|F(w_k)\|\}$ are bounded.
(iv) $\lim_{k\to\infty} \alpha_k \|d_k\| = 0$.
(v) $\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0$.

Proof of Lemma 3. To show (i), let $k \geq 0$ and $\hat{x}$ be a solution of problem (1), then $\langle F(\hat{x}),\ w_{k+1} - \hat{x}\rangle = 0$. Since F is monotone, we have $\langle F(w_{k+1}),\ w_{k+1} - \hat{x}\rangle \geq \langle F(\hat{x}),\ w_{k+1} - \hat{x}\rangle = 0$. Therefore, it holds that

$$\langle F(w_{k+1}),\ x_k - \hat{x}\rangle = \langle F(w_{k+1}),\ x_k - w_{k+1} + w_{k+1} - \hat{x}\rangle \geq \langle F(w_{k+1}),\ x_k - w_{k+1}\rangle. \tag{23}$$

From the property (13) we have

$$\left\| P_D\left(x_k - \frac{\langle F(w_{k+1}),\ x_k - w_{k+1}\rangle}{\|F(w_{k+1})\|^2} F(w_{k+1})\right) - \hat{x}\right\| \leq \left\| x_k - \frac{\langle F(w_{k+1}),\ x_k - w_{k+1}\rangle}{\|F(w_{k+1})\|^2} F(w_{k+1}) - \hat{x}\right\|. \tag{24}$$

From the definition of x_{k+1}, we have

$$\begin{aligned}
\|x_{k+1} - \hat{x}\|^2 &\leq \left\| x_k - \hat{x} - \frac{\langle F(w_{k+1}),\ x_k - w_{k+1}\rangle}{\|F(w_{k+1})\|^2} F(w_{k+1})\right\|^2 \\
&= \|x_k - \hat{x}\|^2 - 2\frac{\langle F(w_{k+1}),\ x_k - w_{k+1}\rangle}{\|F(w_{k+1})\|^2}\langle F(w_{k+1}),\ x_k - \hat{x}\rangle + \frac{\langle F(w_{k+1}),\ x_k - w_{k+1}\rangle^2}{\|F(w_{k+1})\|^2} \\
&\leq \|x_k - \hat{x}\|^2 - 2\frac{\langle F(w_{k+1}),\ x_k - w_{k+1}\rangle}{\|F(w_{k+1})\|^2}\langle F(w_{k+1}),\ x_k - w_{k+1}\rangle + \frac{\langle F(w_{k+1}),\ x_k - w_{k+1}\rangle^2}{\|F(w_{k+1})\|^2} \\
&= \|x_k - \hat{x}\|^2 - \frac{\langle F(w_{k+1}),\ x_k - w_{k+1}\rangle^2}{\|F(w_{k+1})\|^2} \\
&\leq \|x_k - \hat{x}\|^2.
\end{aligned} \tag{25}$$

The first and second inequalities respectively follow from (24) and (23). The last inequality holds by dropping the second negative term on the right hand side. This implies that $\|x_k - \hat{x}\| \leq \|x_0 - \hat{x}\|$ for all k, and therefore the sequence $\{x_k\}$ is bounded and $\lim_{k\to\infty} \|x_k - \hat{x}\|$ exists. Let $c_1 = L\|x_0 - \hat{x}\|$, since F is Lipschitzian continuous and $\{\|x_k - \hat{x}\|\}$ is decreasing then

$$\|F(x_k)\| = \|F(x_k) - F(\hat{x})\| \leq L\|x_k - \hat{x}\| \leq \cdots \leq L\|x_0 - \hat{x}\| = c_1. \tag{26}$$

To show (ii), let $k = 0$, by the definition of the search direction (10) we have

$$\|d_0\| = \|F(x_0)\| \leq c_1. \tag{27}$$

Suppose $k > 0$ and $\beta_k^{PMHS} \leq 0$, then $\max\{\beta_k^{PMHS}, 0\} = 0$ and (10) reduces

$$d_k = -v_k F(x_k), \quad k = 1, 2, ...,$$

Therefore, combining with (20) and (26) yields

$$\|d_k\| = v_k \|F(x_k)\| \leq \frac{c_1}{a}. \tag{28}$$

By the Lipschitzian continuity, we have

$$\begin{aligned} \|\gamma_{k-1}\| &= \|F(w_k) - F(x_{k-1}) + a(w_k - x_{k-1})\| \\ &\leq L\|w_k - x_{k-1}\| + a\|w_k - x_{k-1}\| \\ &\leq (L+a)\alpha_{k-1}\|d_{k-1}\|. \end{aligned} \tag{29}$$

Again, suppose $k > 0$ and $\beta_k^{PMHS} > 0$, we have

$$\begin{aligned} \|d_k\| &= \| - v_k F(x_k) + \beta_k^{PMHS} d_{k-1}\| \\ &\leq v_k \|F(x_k)\| + |\beta_k^{PMHS}| \|d_{k-1}\| \\ &= v_k \|F(x_k)\| + \left| \frac{\langle F(x_k), d_{k-1}\rangle}{\|d_{k-1}\|^2} - \frac{\|\gamma_{k-1}\|^2}{\langle \gamma_{k-1}, d_{k-1}\rangle^2} \langle F(x_k), d_{k-1}\rangle \right| \|d_{k-1}\| \\ &\leq v_k \|F(x_k)\| + \left[\frac{\|F(x_k)\|\|d_{k-1}\|}{\|d_{k-1}\|^2} + \frac{\|\gamma_{k-1}\|^2}{\langle \gamma_{k-1}, d_{k-1}\rangle^2} \|F(x_k)\|\|d_{k-1}\| \right] \|d_{k-1}\| \\ &= v_k \|F(x_k)\| + \|F(x_k)\| + \frac{\|\gamma_{k-1}\|^2}{\langle \gamma_{k-1}, d_{k-1}\rangle^2} \|F(x_k)\|\|d_{k-1}\|^2 \\ &\leq \frac{1}{a}\|F(x_k)\| + \|F(x_k)\| + \frac{(L+a)^2\alpha_{k-1}^2\|F(x_k)\|\|d_{k-1}\|^4}{a^2\alpha_{k-1}^2\|d_{k-1}\|^4} \\ &= \frac{1}{a}\|F(x_k)\| + \|F(x_k)\| + \frac{(L+a)^2}{a^2}\|F(x_k)\| \\ &= \left[\frac{1}{a} + 1 + \frac{(L+a)^2}{a^2}\right] \|F(x_k)\| \\ &\leq \left[\frac{1}{a} + 1 + \frac{(L+a)^2}{a^2}\right] c_1. \end{aligned}$$

The first and second inequalities follow from triangle inequality and Cauchy–Schwartz inequality, respectively. The third inequality follows from (18) and (29), while the fourth inequality follows from (26). If we let $c := \left[\frac{1}{a} + 1 + \frac{(L+a)^2}{a^2}\right] c_1$, then it holds that

$$\|d_k\| \leq c, \quad \forall\, k. \tag{30}$$

Next, we show (iii). By the boundedness $\{x_k\}$ and (30), it follows from the definition w_k that $\{w_k\}$ is also bounded. By Lipschitzian continuity of F, there exists some constant c_2 for which

$$\|F(w_k)\| \leq c_2, \quad \forall\, k \geq 0. \tag{31}$$

To show (iv), from (25), we deduce

$$\langle F(w_{k+1}), \alpha_k d_k\rangle^2 \leq \|F(w_{k+1})\|^2(\|x_k - \hat{x}\|^2 - \|x_{k+1} - \hat{x}\|^2). \tag{32}$$

Since the stepsize α_k in Step 3 of Algorithm 1 satisfies $\alpha_k \leq 1, \forall\, k$, then from (14), we have

$$\sigma^2\alpha_k^4\|d_k\|^4\|F(w_{k+1})\|^{2/r} \leq \sigma^2\alpha_k^2\|d_k\|^4\|F(w_{k+1})\|^{2/r} \leq \langle F(w_{k+1}),\, \alpha_k d_k\rangle^2. \tag{33}$$

Combining (32) and (33) gives

$$\sigma^2\alpha_k^4\|d_k\|^4\|F(w_{k+1})\|^{2/r} \leq \|F(w_{k+1})\|^2(\|x_k-\hat{x}\|^2 - \|x_{k+1}-\hat{x}\|^2). \tag{34}$$

Multiplying both sides of (34) by $\|F(w_{k+1})\|^{-2/r}$ and using (31) gives

$$\sigma^2\alpha_k^4\|d_k\|^4 \leq \|F(w_{k+1})\|^{2-2/r}(\|x_k-\hat{x}\|^2 - \|x_{k+1}-\hat{x}\|^2) \leq c_2^{2-2/r}(\|x_k-\hat{x}\|^2 - \|x_{k+1}-\hat{x}\|^2). \tag{35}$$

Taking limits of both sides give

$$\sigma^2\lim_{k\to\infty}\alpha_k^4\|d_k\|^4 = 0. \tag{36}$$

Thus,

$$\lim_{k\to\infty}\alpha_k\|d_k\| = 0. \tag{37}$$

Finally, we show (v). Equation (37), together with the definition of w_{k+1} in Step 4 of Algorithm 1 yields

$$\lim_{k\to\infty}\|w_{k+1}-x_k\| = 0. \tag{38}$$

By the property of projection (13), we have

$$\begin{aligned}
\lim_{k\to\infty}\|x_{k+1}-x_k\| &= \lim_{k\to\infty}\left\|P_D\left[x_k - \frac{\langle F(w_{k+1}),\, x_k-w_{k+1}\rangle}{\|F(w_{k+1})\|^2}F(w_{k+1})\right]-x_k\right\| \\
&\leq \lim_{k\to\infty}\left\|x_k - \frac{\langle F(w_{k+1}),\, x_k-w_{k+1}\rangle}{\|F(w_{k+1})\|^2}F(w_{k+1})-x_k\right\| \\
&\leq \lim_{k\to\infty}\|x_k-w_{k+1}\| \\
&= 0.
\end{aligned} \tag{39}$$

□

Theorem 1. *Let $\{x_k\}$ be the sequence generated by Algorithm 1. Suppose Assumption 1 holds then*

(i) $\liminf_{k\to\infty}\|F(x_k)\| = 0$.
(ii) the sequence $\{x_k\}$ converges to a point $\hat{x}$ which satisfies $F(\hat{x}) = 0$.

Proof of Theorem 1. We prove (i) by contradiction. Now suppose that

$$\liminf_{k\to\infty}\|F(x_k)\| > 0, \quad \forall\, k,$$

then we can find some positive constant, say ϑ such that

$$\|F(x_k)\| \geq \vartheta,\ \forall\, k. \tag{40}$$

Applying the Cauchy–Schwarz inequality to (16) yields $t\|F(x_k)\| \leq \|d_k\|$. This, together with (40) gives $\|d_k\| \geq t\vartheta$. Combining with (37), we obtain

$$\lim_{k\to\infty}\alpha_k = 0. \tag{41}$$

Since F satisfies Lipschitzian continuity, then it holds

$$\begin{aligned} \|F(x_k + \varrho^{-1}\alpha_k d_k)\| &= \|F(x_k + \varrho^{-1}\alpha_k d_k) - F(\hat{x})\| \\ &\leq L\|x_k + \varrho^{-1}\alpha_k d_k - \hat{x}\| \\ &\leq L\|x_k - \hat{x}\| + \varrho^{-1}\alpha_k\|d_k\| \\ &\leq m, \end{aligned} \tag{42}$$

where m is a positive constant. The last inequality follows from the boundedness of $\{x_k\}$ and $\{d_k\}$ together with the definition of α_k. If $\alpha_k \neq \kappa$, since Algorithm 1 uses a backtracking process to compute α_k starting from κ, then $\varrho^{-1}\alpha_k$ does not satisfy (14), that is,

$$\begin{aligned} -\langle F(x_k + \varrho^{-1}\alpha_k d_k),\ d_k\rangle &< \sigma\varrho^{-1}\alpha_k\|d_k\|^2\|F(x_k + \varrho^{-1}\alpha_k d_k)\|^{1/r} \\ &\leq \sigma\varrho^{-1}\alpha_k c^2 m^{1/r} \\ &= M\alpha_k, \end{aligned} \tag{43}$$

where $M := \sigma\varrho^{-1}c^2 m^{1/r}$ and the second inequality follows from (30) and (42). Since $\{x_k\}$ and $\{\|d_k\|\}$ are bounded, there exist some accumulation points $\hat{x}$ and $\hat{d}$ of $\{x_k\}$ and $\{\|d_k\|\}$ respectively and some infinite index sets K and K_* for which $\lim\limits_{k\in K} x_k = \hat{x}$ and $\lim\limits_{k\in K_*} d_k = \hat{d}$ where $K_* \subset K$. Therefore by the continuity of F and taking limit on both sides of inequality (43) for $k \in K_*$ we obtain

$$-\langle F(\hat{x}),\ \hat{d}\rangle \leq 0. \tag{44}$$

On the other hand, (16) and (40) gives $-\langle F_k,\ d_k\rangle \geq t\vartheta^2$. Taking limit for $k \in K_*$ we obtain

$$-\langle F(\hat{x}),\ \hat{d}\rangle > 0. \tag{45}$$

The inequalities (44) and (45) yield contradiction and therefore (i) must hold.

Finally, we show (ii) holds. Now, since F is continuous and the sequence $\{x_k\}$ is bounded, then there is some accumulation point of $\{x_k\}$ say $\hat{x}$ for which $\|F(\hat{x})\| = 0$. By boundedness of $\{x_k\}$, we can find subsequence $\{x_{k_j}\}$ of $\{x_k\}$ for which $\lim\limits_{j\to\infty}\|x_{k_j} - \hat{x}\| = 0$. Since $\lim\limits_{k\to\infty}\|x_k - \hat{x}\|$ exists, from (i) of Lemma 3, we can conclude that $\lim\limits_{k\to\infty}\|x_k - \hat{x}\| = 0$ and the proof is complete. $\square$

3. Experiment on Monotone Equations and Application in Signal Processing

In this section, we demonstrate the numerical performance of Algorithm 1 (HSS) and its computational advantage by comparing with some existing methods. We divide the experiment into two subsections where the first subsection is devoted to solving some test problems while the second subsection discussed the application of HSS algorithm in signal recovery. For the monotone nonlinear equations experiment, all solvers were coded in MATLAB R2019b and run on a PC with intel Core(TM) i5-8250u processor with 4GB of RAM and CPU 1.60GHZ.

3.1. First Experiment on Monotone Equations

In this experiment, we apply HSS algorithm to solve eleven test problems and compared the numerical results with some state of the art existing methods namely: modified Harger and Zhang CG method denoted as CGD (also known as CG_Descent) by Xiao and Zhu [15], modified Dai–Yuan CG method denoted as PDY by Liu and Feng [32] and modified Fletcher–Reeves CG method denoted as MFRM by Abubakar et al. [25]. We considered eleven problems where Problems 1–10 are solved with dimension $n = 1000,\ 5000,\ 10{,}000,\ 50{,}000,\ 100{,}000$ and Problem 11 with dimension $n = 4$ (see Appendix A). We used six different starting points, that is, $x_1 = (0.1, 0.1, 0.1, \cdots, 0.1)^T$, $x_2 =$

$(\frac{1}{2}, \frac{1}{2^2}, \frac{1}{2^3}, \cdots, \frac{1}{2^n})^T$, $x_3 = (2, 2, \ldots, 2)^T$, $x_4 = (1, \frac{1}{2}, \frac{1}{3}, \cdots, \frac{1}{n})^T$, $x_5 = (1 - \frac{1}{n}, 1 - \frac{2}{n}, 1 - \frac{3}{n}, \cdots, 0)^T$ and $x_6 = \text{rand}(0, 1)$. This brings the total number of problem runs in this experiment to 306.

We implemented HSS method using the following parameters $\kappa = 1$, $\sigma = 0.01$, $\varrho = 0.5$, $r = 5$ and $a = 0.01$ while the parameters used for CGD, PDY and MFRM methods come from [15,25,32], respectively. We set the terminating criterion for the iteration process as $\|F(x_k)\| \leq 10^{-6}$ or $\|F(w_k)\| \leq 10^{-6}$ and declare failure (denoted by "-") whenever the number of iterations exceeds 1000 and the terminating criterion has not been satisfied.

We carried out the comparison based on ITER (number of iterations), FVAL (number of function evaluations) and TIME (CPU time(s)) and the numerical results obtained by each solver are reported in Tables 1–11. We see from the NORM (norm of the objective function) reported in Tables 1–11 that the proposed HSS algorithm obtained the solutions of all the test problems in each instance while the other three methods (CGD, PDY and MFRM) failed to obtain the solutions of some problems in some instances. This means our proposed algorithm can serve as an alternative to some existing methods. The numerical results reported show that the proposed HSS algorithm recorded least ITER, FVAL and TIME in most instances. We summarized all the information from Tables 1–11 in Figures 1–3 based on the performance profile by Dolan and Moré [38] which tells the percentage win by each solver. In terms of ITER and FVAL, we see from Figures 1 and 2 that the HSS algorithm performs well and won about 70% of the whole experiments compared to the existing methods. In addition, Figure 3 shows that the HSS algorithm is faster than all the three methods compared with. Therefore, we can regard the proposed HSS algorithm as more efficient than CGD, PDY and MFRM methods with respect to the numerical experiments performed.

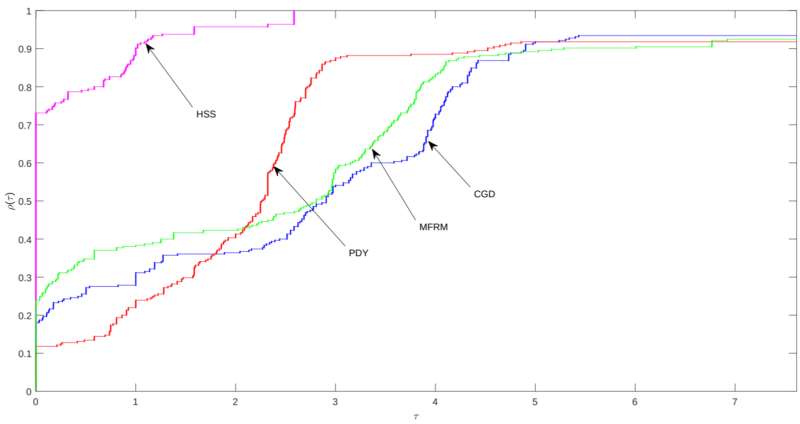

Figure 1. Dolan and Moré performance profile with respect to number of iterations.

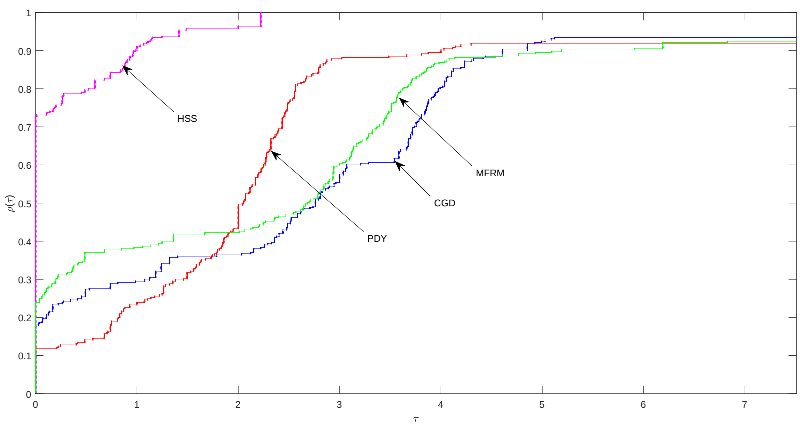

Figure 2. Dolan and Moré performance profile with respect to number of function evaluation.

Table 1. Test results obtained by the four solvers for Problem 1.

Problem 1		**HSS**				**CGD**				**PDY**				**MFRM**			
DIM	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	x1	5	12	0.4454	1.2×10^{-8}	41	84	0.029665	9.97×10^{-7}	36	74	0.0291	3.49×10^{-7}	47	96	0.049823	4.58×10^{-7}
	x2	25	51	0.030772	9.12×10^{-7}	57	116	0.020445	8.17×10^{-7}	42	85	0.015079	8.84×10^{-7}	65	131	0.054894	9.19×10^{-7}
	x3	8	17	0.005867	9.42×10^{-9}	50	102	0.011775	8.56×10^{-7}	46	94	0.017204	2.21×10^{-8}	40	82	0.032647	5.07×10^{-7}
	x4	24	50	0.010883	7.2×10^{-7}	58	118	0.015481	9.51×10^{-7}	40	82	0.018437	3.16×10^{-7}	44	90	0.037726	5.45×10^{-7}
	x5	7	15	0.00825	1.49×10^{-8}	51	104	0.012897	8.46×10^{-7}	51	104	0.019256	9.89×10^{-7}	64	130	0.072206	7.35×10^{-7}
	x6	7	15	0.021386	2.37×10^{-8}	41	84	0.011211	8.86×10^{-7}	46	94	0.018799	7.01×10^{-7}	80	162	0.072448	6.36×10^{-8}
5000	x1	5	11	0.024934	1.79×10^{-7}	40	82	0.040348	8.34×10^{-7}	27	56	0.034997	1.3×10^{-8}	35	72	0.19811	6.59×10^{-7}
	x2	25	51	0.043535	9.12×10^{-7}	57	116	0.04253	8.17×10^{-7}	42	85	0.08315	8.84×10^{-7}	65	131	0.2973	9.19×10^{-7}
	x3	7	16	0.063339	2.09×10^{-8}	48	98	0.05612	8.76×10^{-7}	28	58	0.067063	4.05×10^{-7}	51	104	0.19957	5.68×10^{-7}
	x4	24	50	0.045037	7.2×10^{-7}	58	118	0.083654	9.51×10^{-7}	36	74	0.066975	3.44×10^{-7}	57	116	0.2415	6.8×10^{-7}
	x5	7	15	0.016824	3.17×10^{-8}	49	100	0.069098	8.61×10^{-7}	51	104	0.093807	7.98×10^{-7}	51	104	0.42878	6.81×10^{-7}
	x6	7	15	0.013023	3.37×10^{-8}	38	78	0.061728	8.02×10^{-7}	49	99	0.083743	8.02×10^{-7}	88	178	0.79169	6.7×10^{-7}
10000	x1	5	11	0.025322	1.82×10^{-7}	39	80	0.087834	8.97×10^{-7}	29	60	0.10988	2.14×10^{-8}	37	76	0.3135	3.44×10^{-7}
	x2	25	51	0.092561	9.12×10^{-7}	57	116	0.13838	8.17×10^{-7}	42	85	0.13059	8.84×10^{-7}	65	131	0.58046	9.19×10^{-7}
	x3	7	16	0.021996	2.96×10^{-8}	47	96	0.10667	9.17×10^{-7}	31	64	0.11367	1.18×10^{-7}	93	188	1.4396	2.81×10^{-7}
	x4	24	50	0.077032	7.2×10^{-7}	58	118	0.14476	9.51×10^{-7}	41	84	0.15532	1.1×10^{-7}	48	98	0.38582	8.67×10^{-7}
	x5	7	15	0.020935	4.45×10^{-8}	48	98	0.10544	8.97×10^{-7}	47	95	0.16012	5.62×10^{-7}	97	196	2.8134	9.86×10^{-7}
	x6	7	15	0.02372	4.53×10^{-8}	42	86	0.090851	8.97×10^{-7}	54	109	0.27744	5.59×10^{-7}	114	230	3.2143	2.62×10^{-7}
50000	x1	5	11	0.12861	3.17×10^{-7}	38	78	0.33746	8.43×10^{-7}	28	58	0.44472	8.48×10^{-8}	47	96	1.8109	3.45×10^{-7}
	x2	25	51	0.40895	9.12×10^{-7}	57	116	0.4312	8.17×10^{-7}	42	85	0.58467	8.84×10^{-7}	65	131	2.2362	9.19×10^{-7}
	x3	7	16	0.089772	6.61×10^{-8}	45	92	0.38008	9.78×10^{-7}	44	90	0.76263	2.61×10^{-7}	49	100	3.709	6.73×10^{-7}
	x4	24	50	0.49831	7.2×10^{-7}	58	118	0.49767	9.51×10^{-7}	45	92	0.64548	2.84×10^{-7}	51	104	1.6105	8.18×10^{-7}
	x5	7	15	0.08695	9.9×10^{-8}	46	94	0.40103	9.31×10^{-7}	51	104	0.72555	6.27×10^{-7}	203	408	27.1911	9.8×10^{-7}
	x6	7	15	0.081075	9.63×10^{-8}	41	84	0.37714	9.08×10^{-7}	44	90	0.63276	6.26×10^{-7}	173	348	17.3745	6.02×10^{-7}
100000	x1	5	11	0.1897	4.34×10^{-7}	38	78	0.68124	7.72×10^{-7}	21	44	0.78409	7.7×10^{-7}	40	82	2.3871	7.69×10^{-7}
	x2	25	51	1.1264	9.12×10^{-7}	57	116	1.092	8.17×10^{-7}	42	85	1.2409	8.84×10^{-7}	65	131	3.7292	9.19×10^{-7}
	x3	7	16	0.19532	9.34×10^{-8}	45	92	0.76874	8.38×10^{-7}	37	76	1.5389	4.43×10^{-8}	107	216	16.1536	9.91×10^{-7}
	x4	24	50	0.69604	7.2×10^{-7}	58	118	1.1008	9.51×10^{-7}	32	66	1.0387	7.38×10^{-7}	54	110	3.0635	7.87×10^{-7}
	x5	7	15	0.13571	1.4×10^{-7}	45	92	0.81993	9.9×10^{-7}	51	104	3.0136	7.08×10^{-7}	451	904	155.5752	8.9×10^{-7}
	x6	7	15	0.15849	1.38×10^{-7}	41	84	0.83232	9.94×10^{-7}	49	100	2.741	7.08×10^{-7}	182	366	53.9013	5.58×10^{-7}

Table 2. Test results obtained by the four solvers for Problem 2.

Problem 2		**HSS**				**CGD**				**PDY**				**MFRM**			
DIM	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	x1	4	10	0.065511	4.79×10^{-8}	54	110	0.051537	8.99×10^{-7}	2	6	0.004049	5.17×10^{-7}	2	6	0.00428	5.17×10^{-7}
	x2	7	16	0.004944	3.08×10^{-7}	55	112	0.020758	8.77×10^{-7}	18	38	0.014524	4.14×10^{-7}	60	122	0.030779	8.53×10^{-8}
	x3	8	18	0.007408	5.76×10^{-8}	80	162	0.024411	8.64×10^{-7}	5	12	0.005914	1.74×10^{-8}	5	12	0.002481	1.74×10^{-8}
	x4	8	18	0.004786	5.33×10^{-8}	60	122	0.030894	9.09×10^{-7}	20	42	0.014378	8.06×10^{-7}	86	174	0.041737	6.46×10^{-7}
	x5	8	18	0.006147	2.74×10^{-7}	71	144	0.032592	8.41×10^{-7}	27	56	0.025985	1.81×10^{-7}	96	194	0.044708	2.04×10^{-7}
	x6	8	18	0.007174	2.78×10^{-7}	71	144	0.033933	8.54×10^{-7}	25	52	0.048252	2.21×10^{-7}	89	180	0.051754	8.75×10^{-8}
5000	x1	4	10	0.013144	1.06×10^{-7}	58	118	0.08423	8.02×10^{-7}	2	6	0.011378	1.75×10^{-7}	2	6	0.00446	1.75×10^{-7}
	x2	7	16	0.018096	3.13×10^{-7}	55	112	0.10769	8.7×10^{-7}	30	62	0.19022	1.54×10^{-7}	73	148	0.15676	6.86×10^{-7}
	x3	8	18	0.018438	1.28×10^{-7}	83	168	0.12004	9.71×10^{-7}	5	12	0.011973	2.36×10^{-9}	5	12	0.009605	2.36×10^{-9}
	x4	8	18	0.022149	5.4×10^{-8}	60	122	0.12048	9×10^{-7}	18	38	0.094073	5.88×10^{-8}	61	124	0.13065	8.76×10^{-7}
	x5	8	18	0.034873	9.51×10^{-7}	74	150	0.12629	9.49×10^{-7}	22	46	0.046554	8.16×10^{-7}	77	156	0.20059	1.08×10^{-7}
	x6	8	18	0.035623	9.55×10^{-7}	74	150	0.13564	9.58×10^{-7}	22	46	0.079387	7.45×10^{-7}	87	176	0.2346	4.59×10^{-7}
10000	x1	4	10	0.015467	1.5×10^{-7}	59	120	0.15681	9.04×10^{-7}	2	6	0.012605	1.21×10^{-7}	2	6	0.015088	1.21×10^{-7}
	x2	7	16	0.030136	3.13×10^{-7}	55	112	0.15155	8.69×10^{-7}	28	58	0.52817	1.05×10^{-7}	72	146	0.33387	9.65×10^{-7}
	x3	8	18	0.030859	1.82×10^{-7}	85	172	0.52762	8.77×10^{-7}	5	12	0.10838	3.62×10^{-9}	5	12	0.018115	1.24×10^{-9}
	x4	8	18	0.062129	5.41×10^{-8}	60	122	0.36479	8.99×10^{-7}	15	32	0.052187	5.56×10^{-7}	78	158	0.3405	1.51×10^{-7}
	x5	9	20	0.064209	1.33×10^{-8}	76	154	0.20614	8.58×10^{-7}	25	52	0.089306	9×10^{-7}	101	204	0.53394	5.47×10^{-7}
	x6	9	20	0.067411	1.33×10^{-8}	76	154	0.3443	8.58×10^{-7}	31	64	0.12864	8.79×10^{-7}	73	148	0.44865	6.23×10^{-7}
50000	x1	4	10	0.068733	3.34×10^{-7}	63	128	0.7666	8.26×10^{-7}	2	6	0.03321	6.32×10^{-8}	2	6	0.031681	6.32×10^{-8}
	x2	7	16	0.20861	3.14×10^{-7}	55	112	1.2994	8.68×10^{-7}	21	44	0.30953	6.18×10^{-10}	49	100	0.99731	2.56×10^{-7}
	x3	8	18	0.24123	4.06×10^{-7}	89	180	1.1196	8.02×10^{-7}	6	14	0.25741	9.31×10^{-9}	5	12	0.061125	4.01×10^{-10}
	x4	8	18	0.40721	5.42×10^{-8}	60	122	1.6253	8.98×10^{-7}	15	32	0.42982	2.59×10^{-7}	61	124	1.2389	7.49×10^{-7}
	x5	9	20	0.25666	2.98×10^{-8}	79	160	1.2268	9.81×10^{-7}	23	48	0.34981	1.82×10^{-7}	108	218	1.9382	7.03×10^{-7}
	x6	9	20	0.1428	2.98×10^{-8}	79	160	1.9121	9.81×10^{-7}	24	50	0.40973	7.86×10^{-7}	111	224	2.3737	7.48×10^{-7}
100000	x1	4	10	0.14432	4.73×10^{-7}	64	130	2.8646	9.34×10^{-7}	2	6	0.061232	5.4×10^{-8}	2	6	0.058923	5.4×10^{-8}
	x2	7	16	0.18563	3.14×10^{-7}	55	112	1.6057	8.68×10^{-7}	29	60	0.86796	6.52×10^{-8}	71	144	2.482	9.83×10^{-7}
	x3	8	18	0.26077	5.74×10^{-7}	90	182	2.6208	9.07×10^{-7}	7	16	0.29491	1.1×10^{-9}	5	12	0.11839	2.71×10^{-10}
	x4	8	18	0.30728	5.42×10^{-8}	60	122	1.8968	8.98×10^{-7}	15	32	0.64725	2.34×10^{-7}	61	124	2.7544	4.46×10^{-7}
	x5	9	20	0.29364	4.22×10^{-8}	81	164	2.4882	8.87×10^{-7}	20	42	0.95254	3.99×10^{-7}	86	174	3.8181	8.38×10^{-7}
	x6	9	20	0.30376	4.21×10^{-8}	81	164	2.1676	8.87×10^{-7}	21	44	0.67373	7.01×10^{-8}	130	262	4.741	5.23×10^{-7}

Table 3. Test results obtained by the four solvers for Problem 3.

Problem 3		**HSS**				**CGD**				**PDY**				**MFRM**			
DIM	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	x1	4	10	0.02031	1.97×10^{-8}	67	136	0.033374	9.14×10^{-7}	21	44	0.016594	7.52×10^{-7}	6	14	0.003308	2.51×10^{-20}
	x2	4	10	0.004364	3.86×10^{-8}	59	120	0.014332	9.74×10^{-7}	19	40	0.020992	5.27×10^{-7}	56	114	0.028785	1.95×10^{-7}
	x3	5	12	0.009166	4.62×10^{-7}	79	160	0.020838	9.14×10^{-7}	24	50	0.028946	8.19×10^{-7}	7	16	0.003857	5.28×10^{-7}
	x4	4	10	0.003387	4.2×10^{-7}	63	128	0.015309	8.43×10^{-7}	20	42	0.023507	5.35×10^{-7}	59	119	0.034386	9.47×10^{-7}
	x5	5	12	0.007451	4.72×10^{-8}	75	152	0.018522	8.3×10^{-7}	23	48	0.028748	9.58×10^{-7}	62	126	0.036143	2.58×10^{-7}
	x6	5	12	0.003156	4.83×10^{-8}	75	152	0.018401	8.24×10^{-7}	23	48	0.01997	9.53×10^{-7}	85	172	0.041196	2.93×10^{-7}
5000	x1	4	10	0.00634	4.4×10^{-8}	71	144	0.056879	8.37×10^{-7}	22	46	0.032142	2.34×10^{-22}	6	14	0.013685	6.96×10^{-7}
	x2	4	10	0.007762	3.86×10^{-8}	59	120	0.064447	9.74×10^{-7}	19	40	0.045592	5.27×10^{-7}	56	114	0.1389	1.95×10^{-7}
	x3	6	14	0.036929	1.02×10^{-8}	83	168	0.097078	8.37×10^{-7}	26	54	0.043952	9.45×10^{-7}	8	18	0.017421	0
	x4	4	10	0.013407	4.21×10^{-7}	63	128	0.066394	8.43×10^{-7}	20	42	0.051169	5.35×10^{-7}	47	96	0.14942	4.76×10^{-7}
	x5	5	12	0.010295	1.06×10^{-7}	78	158	0.11486	9.51×10^{-7}	25	52	0.050411	5.36×10^{-7}	84	170	0.23801	5.46×10^{-7}
	x6	5	12	0.016684	1.07×10^{-7}	78	158	0.15325	9.49×10^{-7}	25	52	0.037297	5.39×10^{-7}	58	117	0.22868	5.38×10^{-7}
10000	x1	4	10	0.022699	6.23×10^{-8}	72	146	0.16795	9.47×10^{-7}	22	46	0.060396	3.31×10^{-22}	6	14	0.020364	7.28×10^{-20}
	x2	4	10	0.014139	3.86×10^{-8}	59	120	0.14901	9.74×10^{-7}	19	40	0.18271	5.27×10^{-7}	56	114	0.3439	1.95×10^{-7}
	x3	6	14	0.019224	1.45×10^{-8}	84	170	0.69638	9.47×10^{-7}	27	56	0.65826	6.68×10^{-7}	8	18	0.030973	3.47×10^{-20}
	x4	4	10	0.011995	4.21×10^{-7}	63	128	0.33074	8.43×10^{-7}	20	42	0.076416	5.35×10^{-7}	36	74	0.25376	3.81×10^{-7}
	x5	5	12	0.015141	1.49×10^{-7}	80	162	0.4328	8.61×10^{-7}	25	52	0.10034	7.58×10^{-7}	72	146	0.50642	2.51×10^{-7}
	x6	5	12	0.022858	1.49×10^{-7}	80	162	0.22692	8.63×10^{-7}	25	52	0.16107	7.59×10^{-7}	77	156	0.49567	1.88×10^{-7}
50000	x1	4	10	0.052651	1.39×10^{-7}	76	154	1.2123	8.67×10^{-7}	24	50	1.1795	6.65×10^{-7}	7	16	0.17603	8.43×10^{-20}
	x2	4	10	0.057156	3.86×10^{-8}	59	120	1.0414	9.74×10^{-7}	19	40	0.35856	5.27×10^{-7}	56	114	1.395	1.95×10^{-7}
	x3	6	14	0.063795	3.24×10^{-8}	88	178	1.5448	8.67×10^{-7}	27	56	0.41163	2.37×10^{-20}	7	16	0.10883	4.33×10^{-18}
	x4	4	10	0.04699	4.21×10^{-7}	63	128	0.80096	8.43×10^{-7}	20	42	0.2929	5.35×10^{-7}	42	86	1.0935	9.35×10^{-7}
	x5	5	12	0.17235	3.34×10^{-7}	83	168	1.0549	9.86×10^{-7}	26	54	0.46606	8.48×10^{-7}	83	168	2.0317	3.14×10^{-7}
	x6	5	12	0.061083	3.34×10^{-7}	83	168	1.3178	9.87×10^{-7}	26	54	0.88305	1.54×10^{-22}	86	174	2.0782	4.3×10^{-7}
100000	x1	4	10	0.13629	1.97×10^{-7}	77	156	1.507	9.81×10^{-7}	19	40	0.84867	3.35×10^{-20}	7	16	0.18	3.11×10^{-7}
	x2	4	10	0.11002	3.86×10^{-8}	59	120	1.1762	9.74×10^{-7}	19	40	0.49061	5.27×10^{-7}	56	114	2.3847	1.95×10^{-7}
	x3	6	14	0.17107	4.58×10^{-8}	89	180	2.3631	9.81×10^{-7}	34	70	1.5875	5.62×10^{-7}	7	16	0.20417	5.04×10^{-18}
	x4	4	10	0.17033	4.21×10^{-7}	63	128	2.0171	8.43×10^{-7}	20	42	0.45278	5.35×10^{-7}	53	108	2.3837	2.24×10^{-7}
	x5	5	12	0.27433	4.73×10^{-7}	85	172	2.4078	8.92×10^{-7}	27	56	0.78559	9.24×10^{-7}	80	162	3.2927	3.65×10^{-7}
	x6	5	12	0.42743	4.72×10^{-7}	85	172	2.1616	8.94×10^{-7}	29	60	0.69763	5.07×10^{-7}	86	174	3.0691	2.38×10^{-7}

Table 4. Test results obtained by the four solvers for Problem 4.

Problem 4		HSS				CGD				PDY				MFRM			
DIM	SP	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM
1000	x1	1	3	0.022489	0	1	3	0.047719	0	1	3	0.007245	0	1	3	0.000932	0
	x2	1	3	0.002172	0	1	3	0.002429	0	1	3	0.003757	0	109	219	0.029483	9.28×10^{-7}
	x3	1	3	0.001634	0	1	3	0.00262	0	1	3	0.002875	0	1	3	0.002245	0
	x4	3	8	0.005628	4.47×10^{-8}	1	3	0.00244	0	5	12	0.007667	7.88×10^{-9}	1	3	0.00235	0
	x5	5	12	0.004021	6.49×10^{-7}	1	3	0.003671	0	29	59	0.019814	9.22×10^{-7}	1	3	0.001477	0
	x6	5	12	0.012438	7.19×10^{-7}	1	3	0.001087	0	24	49	0.012885	7.25×10^{-7}	1	3	0.00134	0
5000	x1	1	3	0.005965	0	1	3	0.003018	0	1	3	0.002284	0	1	3	0.002272	0
	x2	1	3	0.004459	0	1	3	0.002205	0	1	3	0.003033	0	109	219	0.13851	9.28×10^{-7}
	x3	1	3	0.002232	0	1	3	0.006697	0	1	3	0.003255	0	1	3	0.007835	0
	x4	3	8	0.007485	4.15×10^{-8}	1	3	0.002504	0	5	12	0.008525	6.9×10^{-9}	1	3	0.003338	0
	x5	6	14	0.015053	1.46×10^{-8}	1	3	0.002627	0	25	52	0.02548	6.94×10^{-7}	1	3	0.003799	0
	x6	6	14	0.010208	1.48×10^{-8}	1	3	0.002331	0	26	53	0.032945	6.87×10^{-7}	1	3	0.004136	0
10000	x1	1	3	0.004832	0	1	3	0.013794	0	1	3	0.003671	0	1	3	0.00363	0
	x2	1	3	0.003793	0	1	3	0.00365	0	1	3	0.004001	0	109	219	0.36648	9.28×10^{-7}
	x3	1	3	0.004974	0	1	3	0.004041	0	1	3	0.005493	0	1	3	0.01389	0
	x4	3	8	0.010592	4.11×10^{-8}	1	3	0.009375	0	5	12	0.026602	6.79×10^{-9}	1	3	0.008508	0
	x5	6	14	0.021774	2.07×10^{-8}	1	3	0.009331	0	27	55	0.05058	6.22×10^{-7}	1	3	0.006177	0
	x6	6	14	0.013876	2.05×10^{-8}	1	3	0.006686	0	23	48	0.048102	8.31×10^{-7}	1	3	0.00721	0
50000	x1	1	3	0.021524	0	1	3	0.017893	0	1	3	0.021182	0	1	3	0.013617	0
	x2	1	3	0.05473	0	1	3	0.021253	0	1	3	0.010713	0	109	219	1.148	9.28×10^{-7}
	x3	1	3	0.021119	0	1	3	0.013569	0	1	3	0.018747	0	1	3	0.04227	0
	x4	3	8	0.073359	4.07×10^{-8}	1	3	0.009532	0	5	12	0.06247	6.69×10^{-9}	1	3	0.017147	0
	x5	6	14	0.063864	4.64×10^{-8}	1	3	0.011356	0	1	3	0.026335	0	1	3	0.020316	0
	x6	6	14	0.10304	4.7×10^{-8}	1	3	0.008921	0	1	3	0.02759	0	1	3	0.022485	0
100000	x1	1	3	0.035424	0	1	3	0.016441	0	1	3	0.034987	0	1	3	0.024402	0
	x2	1	3	0.027929	0	1	3	0.03353	0	1	3	0.036914	0	109	219	2.2239	9.28×10^{-7}
	x3	1	3	0.039706	0	1	3	0.017034	0	1	3	0.11907	0	1	3	0.082234	0
	x4	3	8	0.083596	4.07×10^{-8}	1	3	0.017529	0	5	12	0.083531	6.68×10^{-9}	1	3	0.04628	0
	x5	6	14	0.25359	6.57×10^{-8}	1	3	0.016306	0	1	3	0.033998	0	1	3	0.039002	0
	x6	6	14	0.1416	6.95×10^{-8}	1	3	0.031449	0	1	3	0.024931	0	1	3	0.041211	0

Table 5. Test results obtained by the four solvers for Problem 5.

Problem 5		**HSS**				**CGD**				**PDY**				**MFRM**			
DIM	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	x1	4	10	0.059751	3.96×10^{-7}	82	166	0.049837	8.42×10^{-7}	26	54	0.023436	6.16×10^{-7}	68	138	0.077811	3.54×10^{-7}
	x2	4	10	0.004665	4.11×10^{-7}	82	166	0.036694	8.73×10^{-7}	26	54	0.018791	6.39×10^{-7}	74	150	0.062604	4.38×10^{-7}
	x3	4	10	0.003302	1.09×10^{-7}	76	154	0.058903	8.81×10^{-7}	24	50	0.014953	6.76×10^{-7}	64	130	0.05319	5.6×10^{-7}
	x4	4	10	0.004825	4.1×10^{-7}	82	166	0.036868	8.71×10^{-7}	26	54	0.013522	6.38×10^{-7}	78	158	0.060024	5.4×10^{-7}
	x5	4	10	0.003849	3.39×10^{-7}	81	164	0.031956	8.99×10^{-7}	26	54	0.018961	5.26×10^{-7}	75	152	0.067381	9.61×10^{-7}
	x6	4	10	0.00315	3.39×10^{-7}	81	164	0.077436	9.01×10^{-7}	26	54	0.014281	5.27×10^{-7}	54	110	0.060038	7.5×10^{-7}
5000	x1	4	10	0.012654	8.89×10^{-7}	85	172	0.15576	9.65×10^{-7}	27	56	0.081796	6.9×10^{-7}	54	110	0.24452	8.65×10^{-7}
	x2	4	10	0.011453	9.23×10^{-7}	86	174	0.4407	8.01×10^{-7}	27	56	0.068774	7.16×10^{-7}	58	118	0.35362	8.98×10^{-7}
	x3	4	10	0.011682	2.44×10^{-7}	80	162	0.33484	8.08×10^{-7}	25	52	0.067068	7.57×10^{-7}	43	88	0.25534	7.14×10^{-7}
	x4	4	10	0.013908	9.23×10^{-7}	86	174	0.24378	8.01×10^{-7}	27	56	0.088127	7.16×10^{-7}	63	128	0.30915	7.22×10^{-7}
	x5	4	10	0.011849	7.6×10^{-7}	85	172	0.15641	8.24×10^{-7}	27	56	0.083818	5.89×10^{-7}	55	112	0.28214	7.38×10^{-7}
	x6	4	10	0.013519	7.61×10^{-7}	85	172	0.21864	8.24×10^{-7}	27	56	0.17961	5.9×10^{-7}	87	176	0.33912	9.84×10^{-7}
10000	x1	5	12	0.049344	6.35×10^{-7}	87	176	0.32399	8.73×10^{-7}	28	58	0.33859	7.32×10^{-7}	51	104	0.62071	7.26×10^{-7}
	x2	5	12	0.054254	6.59×10^{-7}	87	176	0.56731	9.06×10^{-7}	29	60	0.28634	5.7×10^{-7}	49	100	0.49179	8.68×10^{-7}
	x3	4	10	0.025027	3.45×10^{-7}	81	164	0.81926	9.14×10^{-7}	26	54	0.13539	5.35×10^{-7}	58	118	0.53457	9.29×10^{-7}
	x4	5	12	0.037107	6.59×10^{-7}	87	176	0.33429	9.06×10^{-7}	29	60	0.18407	5.69×10^{-7}	48	98	0.58512	8.06×10^{-7}
	x5	5	12	0.029731	5.43×10^{-7}	86	174	0.31597	9.33×10^{-7}	28	58	0.18179	6.25×10^{-7}	78	158	0.7064	6.91×10^{-7}
	x6	5	12	0.030679	5.41×10^{-7}	86	174	0.34123	9.33×10^{-7}	28	58	0.13266	6.26×10^{-7}	59	120	0.60942	5.85×10^{-7}
50000	x1	6	14	0.13438	7.17×10^{-7}	90	182	1.9581	1×10^{-6}	32	66	1.3355	0	62	126	2.0853	6.8×10^{-7}
	x2	6	14	0.14676	7.45×10^{-7}	91	184	1.4814	8.3×10^{-7}	33	68	0.71451	0	59	120	1.9564	9.73×10^{-7}
	x3	4	10	0.083175	7.72×10^{-7}	85	172	1.5701	8.37×10^{-7}	26	54	0.69494	0	43	88	1.5166	4.88×10^{-7}
	x4	6	14	0.14168	7.45×10^{-7}	91	184	1.8901	8.3×10^{-7}	33	68	1.1252	0	46	94	1.7868	4.16×10^{-7}
	x5	6	14	0.20631	6.13×10^{-7}	90	182	1.2432	8.54×10^{-7}	31	64	0.60108	0	67	136	1.7509	6.22×10^{-7}
	x6	6	14	0.15371	6.13×10^{-7}	90	182	2.031	8.54×10^{-7}	31	64	0.96708	0	48	98	1.4123	7.24×10^{-7}
100000	x1	7	16	0.36536	5.12×10^{-7}	92	186	3.4223	9.05×10^{-7}	34	70	2.0212	0	37	76	2.2322	8.4×10^{-7}
	x2	7	16	0.41608	5.32×10^{-7}	92	186	3.4177	9.39×10^{-7}	35	72	2.3869	0	66	134	4.3938	9.15×10^{-7}
	x3	5	12	0.26751	5.51×10^{-7}	86	174	3.1756	9.47×10^{-7}	26	54	1.3934	0	34	70	2.8543	7.66×10^{-7}
	x4	7	16	0.36165	5.32×10^{-7}	92	186	2.9337	9.39×10^{-7}	35	72	1.998	0	57	116	2.9496	9.55×10^{-7}
	x5	6	14	0.41128	8.67×10^{-7}	91	184	2.3545	9.66×10^{-7}	33	68	2.1747	0	71	144	3.5477	6.17×10^{-7}
	x6	6	14	0.27328	8.67×10^{-7}	91	184	2.3414	9.66×10^{-7}	33	68	2.2056	0	59	120	3.2639	6.57×10^{-7}

Table 6. Test results obtained by the four solvers for Problem 6.

Problem 6		HSS				CGD				PDY				MFRM			
DIM	SP	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM
1000	x1	5	12	0.030248	3.21×10^{-8}	36	74	0.032352	9.48×10^{-7}	6	14	0.006658	2.09×10^{-7}	3	8	0.00372	3.24×10^{-7}
	x2	6	14	0.004253	4.69×10^{-7}	37	76	0.010311	7.65×10^{-7}	34	70	0.018938	6.1×10^{-7}	86	174	0.068292	7.13×10^{-7}
	x3	5	12	0.004081	2.21×10^{-7}	36	74	0.010483	9.07×10^{-7}	6	14	0.005091	6.55×10^{-7}	4	10	0.004645	6.55×10^{-8}
	x4	10	22	0.014077	1.03×10^{-8}	37	76	0.010188	7.56×10^{-7}	33	68	0.013095	5.47×10^{-7}	63	128	0.0778	8.47×10^{-7}
	x5	6	14	0.006538	2.76×10^{-7}	36	74	0.010269	6.45×10^{-7}	48	98	0.018494	8.73×10^{-7}	80	162	0.10296	3.08×10^{-7}
	x6	6	14	0.005977	2.91×10^{-7}	36	74	0.012178	6.54×10^{-7}	52	106	0.02832	3.6×10^{-7}	88	178	0.12078	8.96×10^{-7}
5000	x1	5	12	0.024024	7.18×10^{-8}	38	78	0.044492	8.3×10^{-7}	6	14	0.014241	4.68×10^{-7}	3	8	0.018603	7.25×10^{-7}
	x2	6	14	0.018634	6.56×10^{-7}	39	80	0.059781	6.7×10^{-7}	35	72	0.084409	5.59×10^{-7}	69	140	0.39768	6.38×10^{-7}
	x3	5	12	0.017453	4.95×10^{-7}	38	78	0.06979	7.94×10^{-7}	7	16	0.02862	9.35×10^{-8}	4	10	0.017763	1.46×10^{-7}
	x4	8	18	0.026507	5.07×10^{-7}	39	80	0.053907	6.68×10^{-7}	36	74	0.15347	6.96×10^{-7}	71	144	0.47127	9.1×10^{-7}
	x5	6	14	0.019105	6.22×10^{-7}	37	76	0.063182	9.02×10^{-7}	34	70	0.13452	7.37×10^{-7}	85	172	0.66379	3.47×10^{-7}
	x6	6	14	0.014968	6.44×10^{-7}	37	76	0.051971	9.06×10^{-7}	48	98	0.22496	9.43×10^{-7}	83	168	0.55329	9.71×10^{-7}
10000	x1	5	12	0.023248	1.02×10^{-7}	39	80	0.098655	7.34×10^{-7}	6	14	0.0289	6.62×10^{-7}	4	10	0.044346	5.12×10^{-9}
	x2	6	14	0.027323	7.49×10^{-7}	39	80	0.10691	9.48×10^{-7}	16	34	0.058902	8.16×10^{-7}	48	98	0.55498	7.33×10^{-7}
	x3	5	12	0.02135	7×10^{-7}	39	80	0.10212	7.02×10^{-7}	7	16	0.029555	1.36×10^{-7}	4	10	0.035127	2.07×10^{-7}
	x4	8	18	0.030169	4.59×10^{-7}	39	80	0.22112	9.46×10^{-7}	35	72	0.1597	5.62×10^{-7}	83	168	0.87343	6.35×10^{-7}
	x5	6	14	0.028248	8.8×10^{-7}	38	78	0.32903	7.98×10^{-7}	40	82	0.13693	6.2×10^{-7}	83	168	0.80412	7.74×10^{-7}
	x6	6	14	0.026041	7.92×10^{-7}	38	78	0.2385	8.04×10^{-7}	42	86	0.2604	7.67×10^{-7}	79	160	0.89436	7.84×10^{-7}
50000	x1	5	12	0.15962	2.27×10^{-7}	41	84	0.41018	6.42×10^{-7}	7	16	0.16754	9.46×10^{-8}	4	10	0.14045	1.15×10^{-8}
	x2	6	14	0.12	9.08×10^{-7}	41	84	0.48069	8.3×10^{-7}	28	58	0.67966	7.57×10^{-7}	84	170	2.9466	2.31×10^{-7}
	x3	6	14	0.25283	8.32×10^{-9}	40	82	0.55524	9.82×10^{-7}	8	18	0.1135	2.69×10^{-7}	4	10	0.13051	4.63×10^{-7}
	x4	8	18	0.11363	4.12×10^{-7}	41	84	0.73687	8.29×10^{-7}	26	54	0.34848	1.44×10^{-7}	105	212	3.7842	6.72×10^{-7}
	x5	7	16	0.1164	1.05×10^{-8}	40	82	0.41344	6.98×10^{-7}	37	76	0.53249	7.23×10^{-7}	86	174	3.0183	5.09×10^{-7}
	x6	7	16	0.11197	1.04×10^{-8}	40	82	0.47039	6.99×10^{-7}	50	102	1.4389	7.13×10^{-7}	88	178	2.9197	3.2×10^{-7}
100000	x1	5	12	0.3068	3.21×10^{-7}	41	84	1.417	9.08×10^{-7}	7	16	0.18772	8.58×10^{-7}	4	10	0.21746	1.62×10^{-8}
	x2	6	14	0.56886	9.63×10^{-7}	42	86	0.81291	7.34×10^{-7}	35	72	1.0428	7.61×10^{-7}	-	-	-	-
	x3	6	14	0.1754	1.18×10^{-8}	41	84	0.88265	8.69×10^{-7}	8	18	0.36603	3.81×10^{-7}	4	10	0.18537	6.55×10^{-7}
	x4	8	18	0.2628	4.34×10^{-7}	42	86	1.285	7.34×10^{-7}	29	60	1.0324	5.25×10^{-7}	92	186	5.9597	7.9×10^{-7}
	x5	7	16	0.21732	1.48×10^{-8}	40	82	0.69398	9.87×10^{-7}	35	72	1.1286	4.43×10^{-7}	116	234	6.096	9.91×10^{-8}
	x6	7	16	0.28834	1.52×10^{-8}	40	82	1.4087	9.87×10^{-7}	50	102	1.9037	6.08×10^{-7}	87	176	4.7662	9.35×10^{-7}

Table 7. Test results obtained by the four solvers for Problem 7.

Problem 7		HSS				CGD				PDY				MFRM			
DIM	SP	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM
1000	x1	4	10	0.012183	1.68×10^{-8}	151	304	0.18691	9.17×10^{-7}	16	34	0.014957	8.17×10^{-7}	6	14	0.012322	9.16×10^{-8}
	x2	5	11	0.004785	1.65×10^{-7}	133	268	0.056032	9.21×10^{-7}	15	31	0.011109	7.97×10^{-7}	39	79	0.065169	8.34×10^{-7}
	x3	1	3	0.001547	0	2	5	0.004122	0	18	38	0.013079	8.51×10^{-7}	1	3	0.002491	0
	x4	7	15	0.004476	7.97×10^{-8}	140	282	0.046527	9.19×10^{-7}	17	36	0.011909	4.98×10^{-7}	273	547	6.0835	2.24×10^{-10}
	x5	10	21	0.012394	2.24×10^{-9}	164	330	0.056094	9.22×10^{-7}	19	40	0.018864	4.77×10^{-7}	-	-	-	-
	x6	10	21	0.010066	6.38×10^{-7}	164	330	0.067306	9.42×10^{-7}	19	40	0.014743	4.64×10^{-7}	358	717	4.6658	7.11×10^{-7}
5000	x1	4	10	0.11789	3.76×10^{-8}	158	318	0.21518	9.8×10^{-7}	17	36	0.073395	6.85×10^{-7}	6	14	0.042878	2.05×10^{-7}
	x2	5	11	0.015099	1.65×10^{-7}	133	268	0.17588	9.21×10^{-7}	15	31	0.039081	7.97×10^{-7}	39	79	0.25795	8.34×10^{-7}
	x3	6	14	0.037396	1.09×10^{-7}	2	5	0.00964	0	20	42	0.060058	8.59×10^{-7}	1	3	0.006658	0
	x4	7	15	0.032192	7.96×10^{-8}	140	282	0.59813	9.19×10^{-7}	17	36	0.068599	4.99×10^{-7}	229	459	82.1077	0
	x5	10	21	0.020827	7.11×10^{-7}	171	344	0.46259	9.87×10^{-7}	20	42	0.068094	4.02×10^{-7}	172	345	21.112	0
	x6	12	25	0.028397	8.18×10^{-9}	171	344	0.24954	9.82×10^{-7}	20	42	0.11606	4.01×10^{-7}	211	423	92.8425	0
10000	x1	4	10	0.017055	5.32×10^{-8}	162	326	0.45674	9.1×10^{-7}	17	36	0.19021	9.69×10^{-7}	6	14	0.11361	2.9×10^{-7}
	x2	5	11	0.01981	1.65×10^{-7}	133	268	0.56473	9.21×10^{-7}	15	31	0.16351	7.97×10^{-7}	39	79	0.54726	8.34×10^{-7}
	x3	6	14	0.064627	1.55×10^{-7}	2	5	0.016405	0	21	44	0.22851	4.56×10^{-7}	1	3	0.016089	0
	x4	7	15	0.025851	7.96×10^{-8}	140	282	0.8344	9.19×10^{-7}	17	36	0.085903	4.99×10^{-7}	850	1701	193.6507	0
	x5	11	23	0.32958	6.94×10^{-9}	175	352	0.49858	9.16×10^{-7}	20	42	0.10045	5.69×10^{-7}	-	-	-	-
	x6	10	21	0.076488	3.74×10^{-7}	175	352	0.5223	9.16×10^{-7}	20	42	0.10389	5.68×10^{-7}	82	165	6.8805	0
50000	x1	4	10	0.078496	1.19×10^{-7}	169	340	2.5024	9.73×10^{-7}	18	38	0.6056	8.13×10^{-7}	6	14	0.37763	6.48×10^{-7}
	x2	5	11	0.099397	1.65×10^{-7}	133	268	2.1159	9.21×10^{-7}	15	31	0.68363	7.97×10^{-7}	39	79	2.5718	8.34×10^{-7}
	x3	6	14	0.15118	3.46×10^{-7}	2	5	0.026231	0	23	48	0.61079	9.93×10^{-7}	1	3	0.058223	0
	x4	7	15	0.10366	7.96×10^{-8}	140	282	2.1722	9.19×10^{-7}	17	36	0.39142	4.99×10^{-7}	47	95	17.9721	5.55×10^{-9}
	x5	11	23	0.35229	3.92×10^{-7}	182	366	2.3322	9.79×10^{-7}	20	42	0.62861	7.76×10^{-7}	-	-	-	-
	x6	12	25	0.18608	1.01×10^{-8}	182	366	2.4855	9.81×10^{-7}	20	42	0.92	7.75×10^{-7}	-	-	-	-
100000	x1	4	10	0.20729	1.68×10^{-7}	173	348	4.6402	9.03×10^{-7}	19	40	0.75335	4.31×10^{-7}	-	-	-	-
	x2	5	11	0.14682	1.65×10^{-7}	133	268	2.9435	9.21×10^{-7}	15	31	0.8933	7.97×10^{-7}	-	-	-	-
	x3	6	14	0.19741	4.89×10^{-7}	2	5	0.045153	0	27	56	1.7879	4.72×10^{-7}	-	-	-	-
	x4	7	15	0.20939	7.96×10^{-8}	140	282	3.452	9.19×10^{-7}	17	36	0.92705	4.99×10^{-7}	-	-	-	-
	x5	12	25	0.60932	6.89×10^{-9}	186	374	5.0285	9.09×10^{-7}	21	44	1.23	4.11×10^{-7}	-	-	-	-
	x6	13	27	0.3519	1.84×10^{-8}	186	374	4.6665	9.08×10^{-7}	21	44	0.93468	4.11×10^{-7}	-	-	-	-

Table 8. Test results obtained by the four solvers for Problem 8.

Problem 8		**HSS**				**CGD**				**PDY**				**MFRM**			
DIM	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	x1	66	134	0.24751	9.78×10^{-7}	-	-	-	-	-	-	-	-	-	-	-	-
	x2	27	56	0.034158	9.5×10^{-7}	808	1618	0.31885	9.99×10^{-7}	-	-	-	-	-	-	-	-
	x3	1	2	0.003785	0	2	5	0.001531	0	1	2	0.001223	0	1	2	0.006734	0
	x4	67	136	0.049339	9.88×10^{-7}	-	-	-	-	-	-	-	-	67	135	0.035797	5.66×10^{-7}
	x5	73	148	0.0461	9.68×10^{-7}	-	-	-	-	-	-	-	-	64	129	0.065585	9.67×10^{-7}
	x6	72	146	0.04761	9.93×10^{-7}	-	-	-	-	-	-	-	-	63	127	0.035655	9.71×10^{-7}
5000	x1	94	190	0.54224	9.91×10^{-7}	-	-	-	-	-	-	-	-	-	-	-	-
	x2	27	56	0.062239	9.5×10^{-7}	808	1618	1.2269	9.99×10^{-7}	-	-	-	-	-	-	-	-
	x3	1	2	0.002594	0	2	5	0.012859	0	1	2	0.005997	0	1	2	0.00257	0
	x4	80	162	0.38038	9.91×10^{-7}	-	-	-	-	-	-	-	-	73	147	0.18682	9.29×10^{-7}
	x5	101	204	0.43745	9.83×10^{-7}	-	-	-	-	-	-	-	-	108	217	0.30825	9.98×10^{-7}
	x6	101	204	0.31897	9.88×10^{-7}	-	-	-	-	-	-	-	-	108	217	0.28176	9.91×10^{-7}
10000	x1	110	222	0.75291	9.94×10^{-7}	-	-	-	-	-	-	-	-	-	-	-	-
	x2	27	56	0.1057	9.5×10^{-7}	808	1618	3.0397	9.99×10^{-7}	-	-	-	-	-	-	-	-
	x3	1	2	0.003731	0	2	5	0.009001	0	1	2	0.007766	0	1	2	0.003949	0
	x4	83	168	0.4733	9.88×10^{-7}	-	-	-	-	-	-	-	-	74	149	0.4244	9.69×10^{-7}
	x5	117	236	0.76712	9.86×10^{-7}	-	-	-	-	-	-	-	-	152	305	0.70872	9.97×10^{-7}
	x6	117	236	0.82138	9.85×10^{-7}	-	-	-	-	-	-	-	-	153	307	0.702	9.95×10^{-7}
50000	x1	160	322	4.4256	9.95×10^{-7}	-	-	-	-	-	-	-	-	-	-	-	-
	x2	27	56	0.52296	9.5×10^{-7}	808	1618	9.7793	9.99×10^{-7}	-	-	-	-	-	-	-	-
	x3	1	2	0.016082	0	2	5	0.069327	0	1	2	0.019848	0	1	2	0.010272	0
	x4	84	170	2.3099	9.99×10^{-7}	-	-	-	-	-	-	-	-	75	151	1.7006	9.31×10^{-7}
	x5	166	334	4.1716	9.99×10^{-7}	-	-	-	-	-	-	-	-	396	793	8.1566	9.99×10^{-7}
	x6	166	334	4.2682	9.99×10^{-7}	-	-	-	-	-	-	-	-	396	793	6.9611	9.99×10^{-7}
100000	x1	189	380	8.5193	9.9×10^{-7}	-	-	-	-	-	-	-	-	-	-	-	-
	x2	27	56	0.84001	9.5×10^{-7}	808	1618	17.381	9.99×10^{-7}	-	-	-	-	-	-	-	-
	x3	1	2	0.033166	0	2	5	0.12617	0	1	2	0.01928	0	1	2	0.029227	0
	x4	85	172	6.6857	9.79×10^{-7}	-	-	-	-	-	-	-	-	75	151	3.7352	9.32×10^{-7}
	x5	195	392	11.7601	9.93×10^{-7}	-	-	-	-	-	-	-	-	623	1247	21.2654	1×10^{-6}
	x6	195	392	7.9429	9.93×10^{-7}	-	-	-	-	-	-	-	-	624	1249	21.5608	9.99×10^{-7}

Table 9. Test results obtained by the four solvers for Problem 9.

Problem 9		**HSS**				**CGD**				**PDY**				**MFRM**			
DIM	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	x1	61	124	0.066702	9.54×10^{-7}	67	136	0.099921	7.96×10^{-7}	139	280	0.076923	9.73×10^{-7}	317	636	0.71829	9.8×10^{-7}
	x2	31	64	0.018303	8.25×10^{-7}	35	72	0.043891	8.99×10^{-7}	187	376	0.09677	8.84×10^{-7}	245	492	0.52225	9.87×10^{-7}
	x3	52	106	0.029299	7.74×10^{-7}	92	186	0.10058	9.64×10^{-7}	227	456	0.12142	9.23×10^{-7}	291	584	0.6824	9.91×10^{-7}
	x4	36	74	0.017934	9.87×10^{-7}	51	104	0.018526	8.96×10^{-7}	218	438	0.11408	9.53×10^{-7}	282	566	0.94291	9.88×10^{-7}
	x5	49	100	0.023288	9.36×10^{-7}	98	198	0.038489	9.43×10^{-7}	233	468	0.22177	9.99×10^{-7}	320	642	1.3394	9.96×10^{-7}
	x6	52	106	0.029525	8.02×10^{-7}	139	280	0.080835	9.97×10^{-7}	188	378	0.11787	9.81×10^{-7}	-	-	-	-
5000	x1	63	128	0.32998	8.55×10^{-7}	69	140	0.089325	9.55×10^{-7}	188	378	0.57537	8.86×10^{-7}	295	592	3.8478	9.96×10^{-7}
	x2	31	64	0.18052	8.25×10^{-7}	35	72	0.059832	8.99×10^{-7}	187	376	1.2573	8.84×10^{-7}	245	492	2.4301	9.87×10^{-7}
	x3	66	134	0.26308	8.96×10^{-7}	84	170	0.13572	9.76×10^{-7}	277	556	0.8517	9.68×10^{-7}	306	614	3.3586	9.83×10^{-7}
	x4	36	74	0.1387	9.89×10^{-7}	51	104	0.09048	8.94×10^{-7}	218	438	1.1212	9.56×10^{-7}	282	566	2.8587	9.88×10^{-7}
	x5	60	122	0.35995	7.44×10^{-7}	87	176	0.5147	8.48×10^{-7}	181	364	0.76997	9.64×10^{-7}	385	771	4.8374	9.86×10^{-7}
	x6	54	110	0.22668	8.29×10^{-7}	115	232	0.48916	8.87×10^{-7}	199	400	0.66965	9.62×10^{-7}	-	-	-	-
10000	x1	65	132	0.57593	9.3×10^{-7}	71	144	0.2565	9.49×10^{-7}	193	388	2.0299	9.37×10^{-7}	291	584	6.5063	9.97×10^{-7}
	x2	31	64	0.19486	8.25×10^{-7}	35	72	0.11428	8.99×10^{-7}	187	376	1.1546	8.84×10^{-7}	245	492	5.7929	9.87×10^{-7}
	x3	71	144	0.58288	7.63×10^{-7}	83	168	0.30524	9.96×10^{-7}	220	442	2.048	9.39×10^{-7}	313	628	9.0926	9.72×10^{-7}
	x4	36	74	0.25617	9.89×10^{-7}	51	104	0.18279	8.94×10^{-7}	218	438	2.1529	9.56×10^{-7}	282	566	7.5672	9.88×10^{-7}
	x5	64	130	0.44209	8.52×10^{-7}	88	178	0.6975	8.93×10^{-7}	232	466	1.4547	9.66×10^{-7}	385	771	13.336	9.98×10^{-7}
	x6	55	112	0.31903	8.87×10^{-7}	121	244	0.69565	9.14×10^{-7}	202	406	1.9326	9.15×10^{-7}	-	-	-	-
50000	x1	56	114	1.443	9.12×10^{-7}	75	152	1.0305	9.7×10^{-7}	190	382	4.94	9.44×10^{-7}	292	586	22.2723	9.85×10^{-7}
	x2	31	64	0.6458	8.25×10^{-7}	35	72	0.77561	8.99×10^{-7}	187	376	4.2758	8.84×10^{-7}	245	492	18.3378	9.87×10^{-7}
	x3	77	156	2.3404	9.2×10^{-7}	83	168	1.4405	9.09×10^{-7}	233	468	6.0204	9.9×10^{-7}	313	628	31.4896	9.86×10^{-7}
	x4	36	74	0.76958	9.89×10^{-7}	51	104	0.9568	8.94×10^{-7}	218	438	5.508	9.56×10^{-7}	282	566	21.0092	9.88×10^{-7}
	x5	77	156	1.839	7.95×10^{-7}	81	164	1.4263	8.13×10^{-7}	238	478	6.0323	9.93×10^{-7}	477	956	66.1866	3.76×10^{-7}
	x6	59	120	1.4313	6.21×10^{-7}	130	262	2.69	9.73×10^{-7}	207	416	5.3511	9.34×10^{-7}	-	-	-	-
100000	x1	56	114	2.7481	9.28×10^{-7}	77	156	2.7769	8.94×10^{-7}	197	396	9.2178	9.98×10^{-7}	290	582	44.5019	9.93×10^{-7}
	x2	31	64	1.4145	8.25×10^{-7}	35	72	0.97134	8.99×10^{-7}	187	376	8.4259	8.84×10^{-7}	245	492	36.7053	9.87×10^{-7}
	x3	92	186	5.4545	8.29×10^{-7}	85	172	3.0767	9.13×10^{-7}	184	370	8.5041	9.73×10^{-7}	357	716	118.6675	9.98×10^{-7}
	x4	36	74	1.654	9.89×10^{-7}	51	104	2.0814	8.94×10^{-7}	218	438	10.4207	9.56×10^{-7}	282	566	43.7452	9.88×10^{-7}
	x5	69	140	4.4259	8.51×10^{-7}	83	168	2.9639	8.07×10^{-7}	257	516	12.9142	9.96×10^{-7}	331	664	111.4667	9.84×10^{-7}
	x6	59	120	2.7119	8.79×10^{-7}	141	284	5.8394	8.87×10^{-7}	206	414	10.0669	9.33×10^{-7}	-	-	-	-

Table 10. Test results obtained by the four solvers for Problem 10.

Problem 11		HSS				CGD				PDY				MFRM			
DIM	**SP**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**	**ITER**	**FVAL**	**TIME**	**NORM**
1000	x1	72	146	0.046063	9.69×10^{-7}	38	78	0.13552	5.47×10^{-7}	225	452	0.09038	9.68×10^{-7}	41	84	0.063262	8.97×10^{-7}
	x2	69	140	0.026	9.32×10^{-7}	37	76	0.024512	6.72×10^{-7}	205	412	0.080801	9.84×10^{-7}	46	94	0.064366	8.96×10^{-7}
	x3	77	156	0.027549	9.89×10^{-7}	40	82	0.012066	8.38×10^{-7}	152	306	0.060964	9.82×10^{-7}	47	96	0.065376	7.43×10^{-7}
	x4	63	128	0.02025	9.03×10^{-7}	42	86	0.013116	6.97×10^{-7}	226	454	0.089011	9.59×10^{-7}	39	80	0.05465	8.98×10^{-7}
	x5	73	148	0.024453	9.01×10^{-7}	37	76	0.016001	9.83×10^{-7}	204	410	0.082852	9.77×10^{-7}	38	78	0.054682	6.94×10^{-7}
	x6	93	188	0.029533	9.47×10^{-7}	45	92	0.037255	9.99×10^{-7}	328	658	0.14417	9.93×10^{-7}	47	96	0.1194	8.09×10^{-7}
5000	x1	69	140	0.18051	8.25×10^{-7}	37	76	0.072643	7.15×10^{-7}	208	418	0.38838	9.98×10^{-7}	37	76	0.2563	7.23×10^{-7}
	x2	75	152	0.13023	8.3×10^{-7}	37	76	0.044212	4.24×10^{-7}	208	418	0.65663	9.66×10^{-7}	38	78	0.2476	9×10^{-7}
	x3	83	168	0.12786	9.58×10^{-7}	46	94	0.079937	8.06×10^{-7}	157	316	0.84669	9.77×10^{-7}	44	90	0.29073	9.45×10^{-7}
	x4	66	134	0.14343	8.4×10^{-7}	31	64	0.027742	5.17×10^{-7}	202	406	0.41872	9.74×10^{-7}	43	88	0.30094	4.26×10^{-7}
	x5	76	154	0.11437	9.54×10^{-7}	41	84	0.058961	4.94×10^{-7}	205	412	0.39337	9.48×10^{-7}	47	96	0.50356	8.33×10^{-7}
	x6	98	198	0.16855	8.42×10^{-7}	50	102	0.070835	7.33×10^{-7}	351	704	1.3398	9.62×10^{-7}	50	102	0.43595	3.95×10^{-7}
10000	x1	74	150	0.31335	8.07×10^{-7}	41	84	0.11908	3.11×10^{-7}	219	440	1.3276	9.71×10^{-7}	41	84	0.85407	4.69×10^{-7}
	x2	77	156	0.33062	9.84×10^{-7}	42	86	0.14335	7.17×10^{-7}	200	402	1.8449	9.98×10^{-7}	47	96	0.98298	6.67×10^{-7}
	x3	73	148	0.2738	9.79×10^{-7}	39	80	0.12366	6.46×10^{-7}	148	298	0.71134	9.58×10^{-7}	45	92	0.89528	8.1×10^{-7}
	x4	77	156	0.28779	9.88×10^{-7}	39	80	0.12037	4.26×10^{-7}	222	446	1.9026	9.6×10^{-7}	41	84	1.0111	9.98×10^{-7}
	x5	79	160	0.30745	9.61×10^{-7}	35	72	0.10851	9.48×10^{-7}	197	396	0.93803	9.86×10^{-7}	36	74	0.78185	9.23×10^{-7}
	x6	101	204	0.44826	9.82×10^{-7}	51	104	0.14368	6.93×10^{-7}	348	698	2.515	9.97×10^{-7}	42	86	0.88774	5.76×10^{-7}
50000	x1	80	162	1.7087	7.83×10^{-7}	41	84	0.49621	4.59×10^{-7}	204	410	5.1793	9.76×10^{-7}	40	82	3.1302	4.85×10^{-7}
	x2	81	164	2.2268	9.19×10^{-7}	41	84	0.46078	2.73×10^{-7}	201	404	4.2643	9.87×10^{-7}	46	94	3.4281	6.69×10^{-7}
	x3	76	154	1.742	9.9×10^{-7}	42	86	0.56419	9.12×10^{-7}	152	306	2.9933	9.9×10^{-7}	49	100	3.3687	7.99×10^{-7}
	x4	83	168	1.3065	7.66×10^{-7}	37	76	0.57785	6.02×10^{-7}	198	398	4.3892	9.96×10^{-7}	40	82	2.4788	4.09×10^{-7}
	x5	81	164	1.4513	8.87×10^{-7}	40	82	0.54615	9.17×10^{-7}	196	394	3.7983	9.62×10^{-7}	50	102	3.0168	8.14×10^{-7}
	x6	106	214	2.0765	9.35×10^{-7}	50	102	0.6377	7.26×10^{-7}	364	730	7.3788	9.52×10^{-7}	49	100	2.9609	6.42×10^{-7}
100000	x1	80	162	3.7128	9.06×10^{-7}	36	74	1.2562	6.72×10^{-7}	210	422	8.4059	9.9×10^{-7}	47	96	6.3303	4.09×10^{-7}
	x2	81	164	2.8053	9.51×10^{-7}	42	86	1.3892	5.03×10^{-7}	215	432	7.851	1×10^{-6}	40	82	5.602	6.53×10^{-7}
	x3	85	172	3.2804	9.1×10^{-7}	46	94	1.406	3.58×10^{-7}	155	312	6.2393	9.74×10^{-7}	48	98	6.8649	7.43×10^{-7}
	x4	83	168	3.2079	9.94×10^{-7}	43	88	1.207	9.09×10^{-7}	211	424	8.7274	9.6×10^{-7}	58	118	7.2979	6.17×10^{-7}
	x5	81	164	3.1215	8.28×10^{-7}	42	86	1.1724	4.39×10^{-7}	212	426	8.7216	9.92×10^{-7}	49	100	6.5793	9.17×10^{-7}
	x6	109	220	3.7274	9.75×10^{-7}	54	110	1.4392	7×10^{-7}	373	748	13.9937	9.54×10^{-7}	50	102	6.8552	4.66×10^{-7}

Table 11. Test results obtained by the four solvers for Problem 11.

Problem 11	HSS				CGD				PDY				MFRM			
SP	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM	ITER	FVAL	TIME	NORM
x1	69	139	0.071482	9.81×10^{-7}	243	487	0.060688	9.56×10^{-7}	57	115	0.033502	8.4×10^{-7}	48	97	0.059039	8.64×10^{-7}
x2	63	127	0.009891	9.41×10^{-7}	228	457	0.025727	9.77×10^{-7}	52	105	0.018045	8.24×10^{-7}	57	115	0.054443	8.39×10^{-7}
x3	70	141	0.011627	9.32×10^{-7}	246	493	0.028826	9.75×10^{-7}	59	119	0.014909	8.08×10^{-7}	51	103	0.04544	7.37×10^{-7}
x4	68	137	0.010198	9.27×10^{-7}	235	471	0.02545	9.49×10^{-7}	57	115	0.015413	8.21×10^{-7}	61	123	0.052551	9.35×10^{-7}
x5	42	85	0.007375	7.69×10^{-7}	193	387	0.021762	9.83×10^{-7}	39	79	0.013396	9.28×10^{-7}	50	101	0.047834	9.42×10^{-7}
x6	68	137	0.008927	8.96×10^{-7}	250	501	0.026878	9.68×10^{-7}	56	113	0.014667	8.47×10^{-7}	61	123	0.050977	7.61×10^{-7}

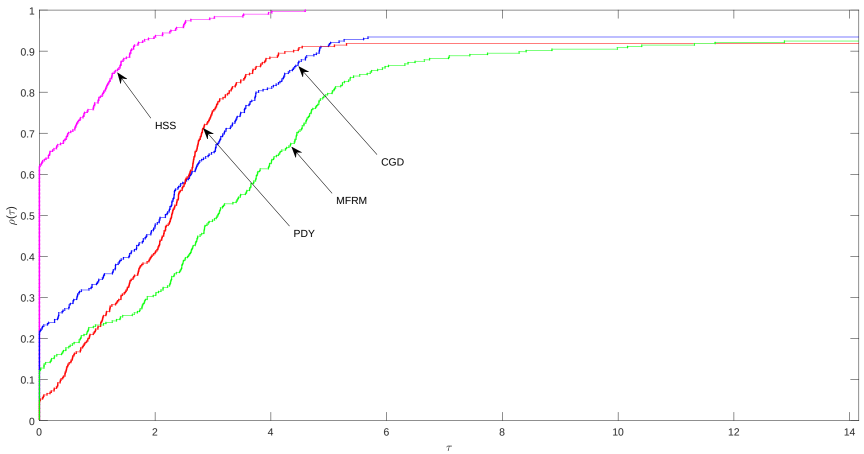

Figure 3. Dolan and Moré performance profile with respect to CPU time.

3.2. Second Experiment on Signal Processing

The major advantage of the proposed HSS algorithm is that it does not require the knowledge of the derivative of an objective function and therefore suitable for solving nonsmooth functions. As mentioned in the introduction section, many applications can be converted into monotone nonlinear equation. In particular, we consider the reconstruction of an original signal, say $\hat{x} \in \mathbb{R}^n$ by minimizing an objective function that contains a linear least square error term and a sparseness-including ℓ_1 regularization term

$$\min_x \frac{1}{2}\|y - Ex\|_2^2 + \mu\|x\|_1, \tag{46}$$

where $x \in \mathbb{R}^n$ $y \in \mathbb{R}^k$ is an observation, $E \in R^{k\times n}$ $(k << n)$ is a linear operator and $\mu \geq 0$ a parameter. Problem (46) has received much attention and different iterative methods for solving it have been proposed by many researchers (see, [39–45]). One of the popular methods for solving (46) is the gradient projection method for sparse reconstruction (also known as GPSR) proposed by Figueiredo et al. [46]. Other gradient-based iterative algorithms include the coordinate gradient descent [47] and the accelerating gradient projection methods [48] among others. These methods enjoyed good performance, even though they required the knowledge of the gradient.

The derivative-free nature of our proposed algorithm makes suitable to deal with the nonsmooth problem (46). However, Algorithm 1 is designed to handle problems in the form of (1) and therefore we need to rewrite problem (46) into the form of problem (1). Fortunately, the work of Figueiredo et al. [46] shows that if we let $q = [u \;\; v]^T$, then (46) can be translated into the following bound-constrained quadratic programming problem

$$\begin{aligned} &\min_q \frac{1}{2}q^T G q + c^T q, \\ &\text{s.t } \; q \geq 0, \end{aligned} \tag{47}$$

where $c = \mu e_{2n} + \begin{pmatrix} -b \\ b \end{pmatrix}, \quad b = E^T y, \quad G = \begin{pmatrix} E^T E & -E^T E \\ -E^T E & E^T E \end{pmatrix}.$

It is not difficult to see that the matrix G is a positive semi-definite. On the other hand, Xiao et al. [2] solve problem (47) in a different way by writing it as the following linear variable inequality problem

$$\begin{cases} \text{Find } q \in \mathbb{R}^n \text{ such that} \\ \langle Gq + c,\, q' - q\rangle \geq 0, \;\; \forall\, q' \geq 0. \end{cases} \tag{48}$$

By taking the special structure of the feasible region of q into consideration, they further showed that problem (48) is equivalent to the following linear complementary problem

$$\begin{cases} \text{Find } q \in \mathbb{R}^n \text{ such that} \\ q \geq 0, \;\; Gq + c \geq 0 \;\text{ and }\; \langle Gq + c,\, q\rangle = 0. \end{cases} \tag{49}$$

We can see that $q \in \mathbb{R}^n$ is a solution of problem (49) if and only if it satisfies the following

$$F(q) := \min\{q, Gq + c\} = 0. \tag{50}$$

In (50), the function F is a vector-valued and the "min" operator denotes the componentwise minimum of two vectors. Problem (50) is in the form of problem (1) and interestingly, Lemma 3 of [49] and Lemma 2.2 of [2] show that the function F satisfies Assumption 1 (ii) that is, Lipschitzian continuity and monotonicity. Hence, the proposed derivative-free HSS algorithm can be applied to solve it. At each iteration, our proposed algorithm is applied to the resulting problem (50) without requiring the Jacobian matrix information.

We compared HSS algorithm and the modified self-adaptive CG method (MSCG) [50] based on their performance in restoring a length-n sparse signal from k observations. We used three metrics; namely ITER (number of iterations), CPU (CPU time) and mean of squared error (MSE) to evaluate the performance of each method. The MSE is usually used to assess the quality of reconstruction and is calculated using the following formula

$$\mathrm{MSE} = \frac{1}{n}\|x - \hat{x}\|^2,$$

where x and $\hat{x}$ denote the original and restored signal respectively. The measurement y contains noise, $y = Gx + \omega$, where ω is the Gaussian noise disturbed as $N(0, 10^{-4})$ and G is the Gaussian matrix generated by command randn(m, n), in MATLAB. The size of the signal is selected with $n = 2^{15}$ and $m = 2^{13}$. The original signal contains 2^7 random nonzero elements. For the signal recovery experiment, the algorithms were coded in MATLAB R2019b and run on a PC with intel(R) Core(TM) i7-10510U processor with 8.00 GB (7.80 GB usable) of RAM and CPU 1.80GHZ–2.30GHz.

We started the iteration process by the measurement signal, i.e., $x_0 = G^T y$, and used $f(x) = \frac{1}{2}\|y - Ax\|_2^2 + \mu\|x\|_1$ as the merit function. We used the same parameters as in the first experiment for HSS method except for $a = 0.2$ while the parameters for MSCG come from [50]. We terminated the iteration process when the relative change of the objective function satisfies $\left|\frac{f(x_k) - f(x_{k-1})}{f(x_{k-1})}\right| < 10^{-5}$. In order to have relatively fair assessment for both methods, we run each code with the same initial point, using the same continuation technique on parameter μ and observed the convergence behaviour of each method to obtain a solution with similar accuracy. We presented numerical results of the experiment for fifteen different noise samples in Table 12 and the average of each column is also computed. In addition, we presented the graphical view of the original, disturbed and recovered signals in Figures 4 and 5. From Table 12, we see that HSS algorithm restored the disturbed signal with least ITER values and also converged faster than MSCG based on the CPU time(s). Moreover, the quality of reconstruction by HSS algorithm is slightly better than that of MSCG method since the MSE with respect to the former is a little less than that of the later.

Table 12. Numerical results of HSS and modified self-adaptive conjugate gradient (MSCG) methods with the initial points randomly generated.

	HSS			**MSCG**		
S/No.	**MSE**	**ITER**	**CPU**	**MSE**	**ITER**	**CPU**
1	2.23×10^{-6}	79	178.11	4.61×10^{-6}	143	309.23
2	3.44×10^{-6}	82	198.38	6.46×10^{-6}	150	359.03
3	2.84×10^{-6}	71	175.06	7.43×10^{-6}	163	386.78
4	4.71×10^{-6}	72	160.22	1.49×10^{-5}	139	305.39
5	1.99×10^{-6}	76	184.17	4.07×10^{-6}	144	341.41
6	1.68×10^{-6}	83	197.36	3.12×10^{-6}	148	342.94
7	3.92×10^{-6}	65	158.09	6.43×10^{-6}	162	396.00
8	2.23×10^{-6}	79	173.63	4.43×10^{-6}	145	314.28
9	1.59×10^{-6}	71	170.02	3.94×10^{-6}	152	365.58
10	5.72×10^{-6}	65	124.77	9.63×10^{-6}	160	311.38
11	3.71×10^{-6}	67	152.91	4.44×10^{-6}	141	306.33
12	2.09×10^{-6}	70	148.05	3.58×10^{-6}	145	307.59
13	2.39×10^{-6}	72	155.38	4.93×10^{-6}	146	287.72
14	1.96×10^{-6}	72	148.69	5.41×10^{-6}	151	308.14
15	2.36×10^{-6}	79	177.47	4.61×10^{-6}	143	318.89
Average	2.86×10^{-6}	73.53	166.82	5.87×10^{-6}	148.80	330.71

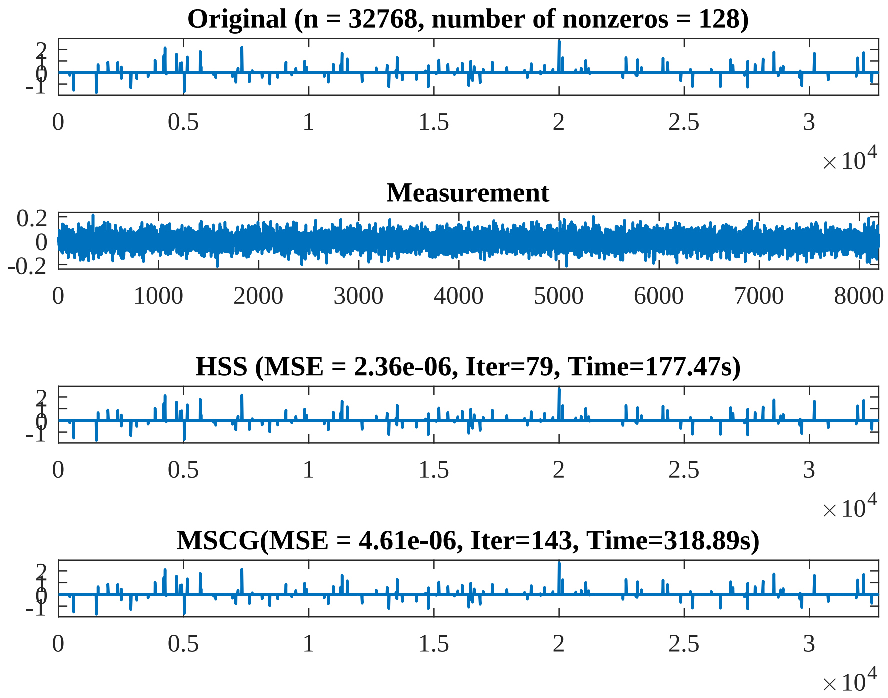

Figure 4. From top to bottom: the original signal, the measurement, the recovered signal by HSS and the recovered signal by MSCG.

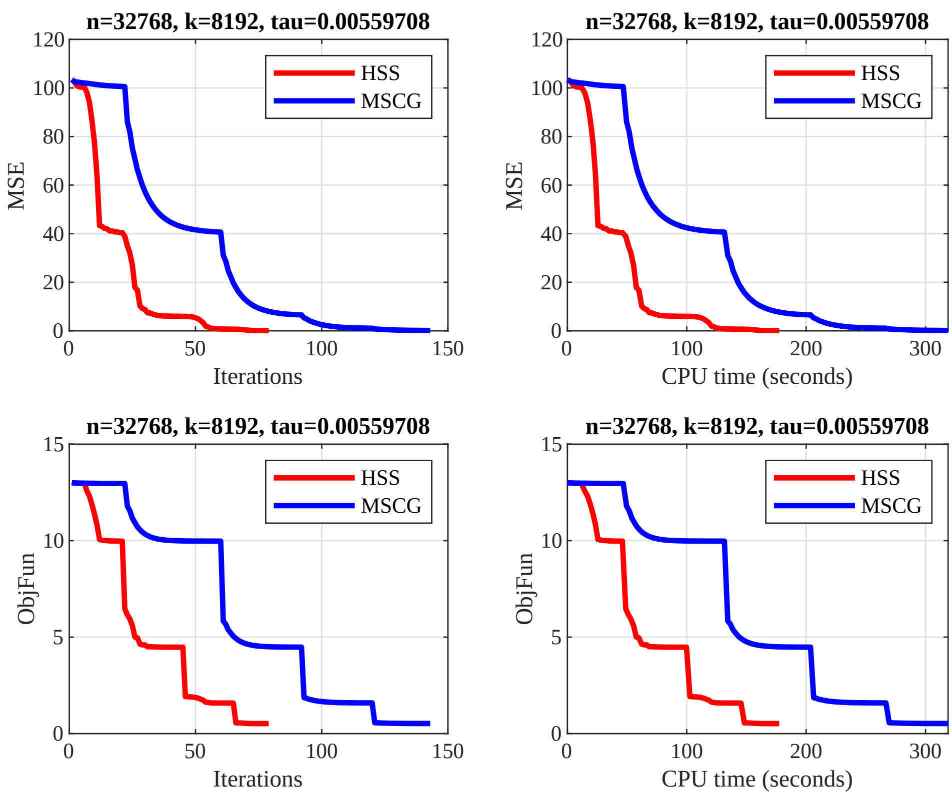

Figure 5. Comparison result of HSS and MSCG. The x-axis represents the number of Iterations top left and bottom left and CPU time in seconds top right and bottom right. The y-axis represents the mean of squared error (MSE) top left and top right and the objective function values bottom left and bottom right.

4. Conclusions

A Hestenes–Stiefel-like derivative-free method with spectral parameter for nonlinear monotone equations has been proposed based on a suitable line search strategy and projection technique. The method is a modification of the conjugate gradient algorithm proposed by Amini et al. [33]. Two types of experiments were presented to show the efficiency of the proposed method. Numerical results reported show that the proposed outperformed three existing methods [15,25,32] and was able to recover some disturbing signals in compressive sensing with better quality than the method in [50]. The convergence analysis of the proposed method was established under standard assumptions.

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

Funding: The authors acknowledge the financial support provided by the Petchra Pra Jom Klao Doctoral Scholarship for Ph.D. program of King Mongkut's University of Technology Thonburi (KMUTT) and Center of Excellence in Theoretical and Computational Science (TaCS-CoE), KMUTT. The first author was supported by the Petchra Pra Jom Klao Ph.D. Research Scholarship from King Mongkut's University of Technology Thonburi (Contract No. 42/2560).

Acknowledgments: The authors would like to thank the anonymous referees for their useful comments. This project was supported by the Center of Excellence in Theoretical and Computational Science (TaCS-CoE) Center under Computational and Applied Science for Smart Innovation research Cluster (CLASSIC), Faculty of Science, KMUTT. This work was done while the first author visits Yunnan University of Finance and Economics, Kunming China.

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

Appendix A. List of Test Problems

The following are the list of test problems used in our experiments where $F = (f_1, f_2, ..., f_n)^T$.
Problem 1 [51]

$$f_1(x) = e^{x_1} - 1$$
$$f_i(x) = e^{x_i} + x_{i-1} - 1, \;\; i = 1, 2, \cdots, n-1,$$

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

Problem 2 [51]

$$f_i(x_i) = \log(x_i + 1) - \frac{x_i}{n}, \;\; i = 1, 2, \cdots, n,$$

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

Problem 3 [51]

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

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

Problem 4 [15]

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

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

Problem 5 [32]

$$
\begin{aligned}
f_1(x) &= x_1 - e^{\cos(h(x_1+x_2))}\\
f_i(x) &= x_i - e^{\cos(h(x_{i-1}+x_i+x_{i+1}))}, \ \ i = 2, \cdots, n-1,\\
f_n(x) &= x_n - e^{\cos(h(x_{n-1}+x_n))},
\end{aligned}
$$

where $h = \frac{1}{n+1}$ and $\Omega = \mathbb{R}^n_+$.

Problem 6 [15]

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

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

Problem 7 [52]

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

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

Problem 8 [51]

$$
f_i(x) = \min[\min(|x_i|, \ x_i^2), \ \max(|x_i|, \ x_i^3)], \ \ i = 1, 2, \cdots, n,
$$

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

Problem 9 [23]

$$
\begin{aligned}
f_1(x) &= 2x_1 - x_2 + e^{x_1} - 1,\\
f_i(x) &= -x_{i-1} + 2x_i - x_{i+1} + e^{x_i} - 1, \ \ i =, 2, \cdots, n-1,\\
f_n(x) &= -x_{n-1} + 2x_n + e^{x_n} - 1,
\end{aligned}
$$

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

Problem 10 [53]

$$
\begin{aligned}
f_1(x) &= \frac{5}{2}x_1 + x_2 - 1,\\
f_i(x) &= x_{i-1} + \frac{5}{2}x_i + x_{i+1} - 1, \ \ i =, 2, \cdots, n-1,\\
f_n(x) &= x_{n-1} + \frac{5}{2}x_n - 1,
\end{aligned}
$$

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

Problem 11 [51] $F : \mathbb{R}^4 \to \mathbb{R}^4$ defined by

$$
F(x) = \begin{pmatrix} 1 & 0 & 0 & 0\\ 0 & 1 & -1 & 0\\ 0 & 1 & 1 & 0\\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1\\ x_2\\ x_3\\ x_4 \end{pmatrix} + \begin{pmatrix} x_1^3\\ x_2^3\\ 2x_3^3\\ 2x_4^3 \end{pmatrix} + \begin{pmatrix} -10\\ 1\\ -3\\ 0 \end{pmatrix}
$$

where $\Omega = \{x \in \mathbb{R}^n : \sum_{i=1}^{4} x_i = 3, \ x_i \geq 0, \ i = 1, 2, 3, 4\}$.

References

1. Iusem, N.A.; Solodov, V.M. Newton-type methods with generalized distances for constrained optimization. *Optimization* **1997**, *41*, 257–278. [CrossRef]
2. Xiao, Y.; Wang, Q.; Hu, Q. Non-smooth equations based methods for l_1-norm problems with applications to compressed sensing. *Nonlinear Anal. Theory Methods Appl.* **2011**, *74*, 3570–3577. [CrossRef]
3. Ghaddar, B.; Marecek, J.; Mevissen, M. Optimal power flow as a polynomial optimization problem. *IEEE Trans. Power Syst.* **2016**, *31*, 539–546. [CrossRef]
4. Sun, W.; Yuan, Y.X. *Optimization Theory and Methods: Nonlinear Programming*; Springer Science & Business Media: New York, NY, USA, 2006.
5. Nocedal, J.; Wright, S.J. *Numerical Optimization*; Springer Science: New York, NY, USA, 2006.
6. 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]
7. Leong, W.J.; Hassan, M.A.; Yusuf, M.W. A matrix-free quasi-Newton method for solving large-scale nonlinear systems. *Comput. Math. Appl.* **2011**, *62*, 2354–2363. [CrossRef]
8. 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]
9. 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]
10. Mohammad, H. Barzilai–Borwein-Like Method for Solving Large-Scale Nonlinear Systems of Equations. *J. Niger. Math. Soc.* **2017**, *36*, 71–83.
11. Bellavia, S.; Macconi, M.; Morini, B. An affine scaling trust-region approach to bound-constrained nonlinear systems. *Appl. Numer. Math.* **2003**, *44*, 257–280. [CrossRef]
12. Zhu, D. An affine scaling trust-region algorithm with interior backtracking technique for solving bound-constrained nonlinear systems. *J. Comput. Appl. Math.* **2005**, *184*, 343–361. [CrossRef]
13. 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]
14. Yuan, Y.-X. Recent advances in trust region algorithms. *Math. Program.* **2015**, *151*, 249–281. [CrossRef]
15. 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]
16. 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]
17. 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.
18. Livieris, I.E.; Pintelas, P. Globally convergent modified Perry's conjugate gradient method. *Appl. Math. Comput.* **2012**, *218*, 9197–9207. [CrossRef]
19. Dai, Z.; Chen, X.; Wen, F. A modified Perry's conjugate gradient method-based derivative-free method for solving large-scale nonlinear monotone equations. *Appl. Math. Comput.* **2015**, *270*, 378–386. [CrossRef]
20. Li, Q.; Li, D.H. A class of derivative-free methods for large-scale nonlinear monotone equations. *IMA J. Numer. Anal.* **2011**, *31*, 1625–1635. [CrossRef]
21. 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]
22. 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]
23. 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**. Available online: http://thaijmath.in.cmu.ac.th/index.php/thaijmath/article/view/3376 (accessed on 21 April 2020).
24. 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]

25. Abubakar, A.B.; Kumam, P.; Mohammad, H.; Awwal, A.M.; Sitthithakerngkiet, K. A Modified Fletcher–Reeves Conjugate Gradient Method for Monotone Nonlinear Equations with Some Applications. *Mathematics* **2019**, *7*, 745. [CrossRef]
26. 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]
27. Abubakar, A.B.; Kumam, P.; Awwal, A.M. An inexact conjugate gradient method for symmetric nonlinear equations. *Comput. Math. Methods* **2019**, *6*, e1065. [CrossRef]
28. Mohammad, H.; Abubakar, A.B. A descent derivative-free algorithm for nonlinear monotone equations with convex constraints. *RAIRO Oper. Res.* **2020**, *54*, 489–505. [CrossRef]
29. Dai, Y-H.; Huang, Y.; Liu, X-W. A family of spectral gradient methods for optimization. *Comput. Optim. Appl.* **2019**, *74*, 43–65. [CrossRef]
30. Huang, S.; Wan, Z. A new nonmonotone spectral residual method for nonsmooth nonlinear equations. *J. Comput. Appl. Math.* **2017**, *313*, 82–101. [CrossRef]
31. Zheng, L.; Yang, L.; Liang, Y. A conjugate gradient projection method for solving equations with convex constraints. *J. Comput. Appl. Math.* **2020**, *375*, 112781. [CrossRef]
32. Liu, J.; Feng, Y. A derivative-free iterative method for nonlinear monotone equations with convex constraints. *Numer. Algorithms* **2019**, *82*, 245–262. [CrossRef]
33. Amini, K.; Faramarzi, P.; Pirfalah, N. A modified Hestenes–Stiefel conjugate gradient method with an optimal property. *Optim. Methods Softw.* **2019**, *34*, 770–782. [CrossRef]
34. Narushima, Y.; Yabe, H.; Ford, J.A. A three-term conjugate gradient method with sufficient descent property for unconstrained optimization. *SIAM J. Optim.* **2011**, *21*, 212–230. [CrossRef]
35. Dai, Y.-H.; Kou, C.-X. A nonlinear conjugate gradient algorithm with an optimal property and an improved Wolfe line search. *SIAM J. Optim.* **2013**, *23*, 296–320. [CrossRef]
36. Cheng, W. A PRP type method for systems of monotone equations. *Math. Comput. Model.* **2009**, *50*, 15–20. [CrossRef]
37. Zhang, L.; Zhou, W. Spectral gradient projection method for solving nonlinear monotone equations. *J. Comput. Appl. Math.* **2006**, *196*, 478–484. [CrossRef]
38. Dolan, E.D.; Moré, J.J. Benchmarking optimization software with performance profiles. *Math. Program. Ser.* **2002**, *91*, 201–213. [CrossRef]
39. Daubechies, I.; Defrise, M.; Mol, C.D. An efficient thresholding algorithm for linear inverse problems with a sparsity constraint. *Commun. Pure Appl. Math. J. Issued Courant Inst. Math. Sci.* **2004**, *57*, 1413–1457. [CrossRef]
40. Figueiredo, M.A.T.; Nowak, R. An EM algorithm for wavelet-based image restoration. *IEEE Trans. Image Process.* **2003**, *12*, 906–916. [CrossRef]
41. Mol, C.D.; Defrise, M. A note on wavelet-based inversion algorithms. *Contemp. Math.* **2002**, *313*, 85–96.
42. Hale, E.T.; Yin, W.; Zhang, Y. *A Fixed-Point Continuation Method for ell$_1$-Regularized Minimization with Applications to Compressed Sensing*; CAAM Technical Report TR07-07 for Rice University: Houston, TX, USA, 2007.
43. 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]
44. Yang, J.; Zhang, Y.; Yin, W. An efficient TVL1 algorithm for deblurring multichannel images corrupted by impulsive noise. *SIAM J. Sci. Comput.* **2009**, *31*, 2842–2865. [CrossRef]
45. Xiao, Y.-H.; Jin, Z.-F. An alternating direction method for linear-constrained matrix nuclear norm minimization. *Numer. Linear Algebra Appl.* **2012**, *19*, 541–554. [CrossRef]
46. Figueiredo, M.A.T.; 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]
47. Yun, S.; Toh, K.-C. A coordinate gradient descent method for ℓ_1-regularized convex minimization. *Comput. Optim. Appl.* **2011**, *48*, 273–307. [CrossRef]
48. Loris, I.; Bertero, M.; Mol, D.C.; Zanella, R.; Zanni, L. Accelerating gradient projection methods for ℓ_1-constrained signal recovery by steplength selection rules. *Appl. Comput. Harmon. Anal.* **2009**, *27*, 247–254. [CrossRef]

49. Pang, J.S. Inexact Newton methods for the nonlinear complementarity problem. *Math. Program.* **1986**, *36*, 54–71. [CrossRef]
50. 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]
51. Awwal, A.M.; Kumam, P.; Abubakar, A.B. A modified conjugate gradient method for monotone nonlinear equations with convex constraints. *Appl. Numer. Math.* **2019**, *145*, 507–520. [CrossRef]
52. Gao, P.; He, C. An efficient three-term conjugate gradient method for nonlinear monotone equations with convex constraints. *Calcolo* **2018**, *55*, 53. [CrossRef]
53. 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**. Available online: http://thaijmath.in.cmu.ac.th/index.php/thaijmath/article/view/3372 (accessed on 21 April 2020).