A Modified Liu and Story Algorithm for Solving System of Convex-constrained Monotone Nonlinear Equations with Applications to Signal Recovery

Yogesh Kumar, A.K. Awasthi, *Member, IAENG,* and Habibu Abdullahi

***Abstract*—The Liu and Story (LS) conjugate gradient (CG) parameter β_k^{LS} (J. Optim, Theory and Appl., 69:129-137, (1991)) is among the classical CG parameters with remarkable numerical performance. However, its global convergence is not always guaranteed. This paper presents a modified LS scheme for solving constrained monotone nonlinear equations with application to compressive sensing. Using eigenvalue analysis, it has been shown that the symmetrized direction matrix of the modified scheme is positive-definite and descent. By incorporating the proposed algorithm with Solodov & Svaiter's projection algorithm (1999), some convex-constrained monotone systems of nonlinear equations were solved with impressive performance. Using appropriate assumptions, the global convergence of the new scheme was proved, and the numerical performance as compared with some recent algorithms in the literature indicates its robustness and effectiveness. Furthermore, the new algorithm performed better than the compared algorithm in restoring some signal problems in compressed sensing.**

Keywords: Liu & Story CG parameter, Eigenvalue analysis, Convex constraints, Compressed sensing, Search direction matrix, Positive definite matrix.

***Index Terms*—Liu & Story CG parameter, Eigenvalue analysis, Convex constraints, Compressed sensing, Search direction matrix, Positive definite matrix.**

I. Introduction

CONSIDER the system:

$$F(x) = 0, \quad x \in \psi. \tag{1}$$

Here, $F : \mathbb{R}^n \to \mathbb{R}^n$ is continuous and monotone mapping. $\psi \subseteq \mathbb{R}^n$ is a non-empty, closed, and convex set. F being monotone satisfies the following

$$(F(x) - F(y))^T (x - y) \geq 0 \quad \forall x, y \in \mathbb{R}^n. \tag{2}$$

In this paper, $\mathbb{R}^n$ and $\|.\|$ refer to the real n-dimensional space and Euclidean norm, respectively, while $F(x_k) = F_k$. The above system is present in sciences and engineering; many scientific formulations often ended up to systems of nonlinear equations. They appeared in mathematical research fields such as differential equations, optimization theory, and integral equations. They are useful in the generalized proximal algorithms with Bregman distances as subproblems [4]. They are applied in chemistry when determining chemical equilibrium points. Similarly, the equilibrium in an economic analysis is determined in the same way [5]. Some variational inequalities are converted to monotone systems before solving [6]. Furthermore, they are applicable in compressed sensing for signal reconstruction and image deblurring [9], [10], [5], [8]. They play an essential role in optimal power flow equations [7]. They are helpful in the radioactive transfer process by discretizing the Chandrasekhar integral equation [5]. They appeared in financial institutions for forecasting mechanizing [21].

Dozens of methods and/or techniques have been investigated, designed, and analyzed for finding the solutions of system (1). The most popular among them is Newton's methods and their improved versions [11], [13]. Steepest descent was investigated in the past, but it was neglected due to its low convergence rate [14]. Levenberg-Marquardt's approach is also a good alternative for solving system (1) [15], [16].

Due to the presence of a large-scale system, researchers' attention turned to CG methods. Being derivative-free, they have the capacity of solving large-scale systems with less CPU time [17].

The CG methods are iterative, i.e., given any starting point, x_k, it generates the next iterative point x_{k+1}, via

$$x_{k+1} = x_k + \alpha_k d_k, \quad k = 0, 1, 2, ..., \tag{3}$$

where α_k, is the step length and d_k is the search direction. The $d_k =$ for the CG methods is represented by the following expression:

$$d_k = \begin{cases} -F_k, & \text{if} \quad k = 0, \\ -F_k + \beta_k s_{k-1}, & \text{if} \quad k \geq 1, \end{cases} \tag{4}$$

$s_{k-1} = x_k - x_{k-1}$, and β_k is the CG parameter.

This CG parameter β_k, is of different expression and format based on the way they have been formed; for example,

$$\beta_k^{LS} = -\frac{F_k^T y_{k-1}}{F_{k-1}^T d_{k-1}}, \tag{5}$$

is called a conjugate descent (CD) CG parameter [43].

The parameter β_k is considered the most critical part of any CG direction, as different CG parameters bring about different CG directions with unique convergence properties and numerical performance. Although some CG parameters

Manuscript received November 14, 2024; revised March 22, 2025.

Yogesh Kumar is a Ph.D. student in the Department of Mathematics, School of Chemical Engineering and Physical Sciences, Lovely Professional University, Phagwara,India 144411. (e-mail:ror.happy@gmail.com)

A.K. Awasthi is a Professor in the Department of Mathematics, School of Chemical Engineering and Physical Sciences, Lovely Professional University, Phagwara, India, 144411. (corresponding author to provide phone: +91-9315517628; e-mail:dramitawasthi@gmail.com)

Habibu Abdullahi is an Assistant Professor in the Department of Mathematics, Sule Lamido University, Kafin Hausa, Jigawa, Nigeria. (e-mail:habibmth.slu@gmail.com)

are computationally excellent, some often do not satisfy the inequality below, an essential ingredient for global convergence [5].

$$F_k^T d_k \leq -\tau \|F_k\|^2, \quad \tau > 0, \tag{6}$$

The conjugate descent (CD) parameter β_k^{CD} belongs to the class of classical CGs with good convergence attributes but weak numerical performance. In an effort to address the above challenge, Yang et al., [18] hybridized Liu-story (LS) and CD parameters using Wolfe's line search strategy. The algorithm they produced was an efficient one. Kaelo et al. [19] hybridized Liu-Story LS, Hestenes-Steifel (HS), and Dai-Yuan (DY) CG parameters and produced an algorithm that satisfied (6), an ingredient for global convergence. Using strong Wolfe's line search strategy, the proposed algorithm converged globally. Similarly, Snezana & Djordjevic [20] proposed a new hybrid of CD and LS, with a hybrid parameter generated in a way that (6) is satisfied. Liu et al. [22] presented a scheme that satisfies condition (6) and presented an LS-type CG parameter for solving the system (1) by applying Powell's strategy on the LS-type scheme [23]. Motivated by the method presented in [24], Wang et al. [25] presented a three-term CG algorithm for the solution to the system (1). Similarly, Li wang, in [26], based on FR-type CG iterative scheme, proposed an algorithm for symmetric system by improving the method of Zhang et al. [27]. The method satisfies (6) and converges globally. Furthermore, in their effort to come up with a better algorithm that solve (1), Liu and Li [28] updated the scheme of Yu et al. [29] and presented a multivariate spectral algorithm for solving the system of the form (1). Motivated by the fact that the numerical performance of the classical CD method is weak, Zhang et al., [30] proposed a three-term CG direction that satisfies the (6) and trust-region property. The proposed method's convergence was achieved by a modified Weak Wolf-Powell (MWWP) line search strategy coupled with the projection algorithm. The numerical results outperformed most of the available algorithms.

Originally, classical conjugate gradient methods were designed for solving systems of the form

$$\min f(x), \quad x \in \mathbb{R}^n, \tag{7}$$

such that $f : \mathbb{R}^n \to \mathbb{R}$ is nonlinear and continuous mapping and the second derivative exists. Considering their nice properties and the fact that the first-order optimality condition for (7) i.e.,$\nabla f(x) = 0$ is the same as (1) whenever, $\nabla f = F_k$ is the first derivative of some nonlinear functions, all the methods applied to (7), can be extended to tackle the problem of the form (1).

II. Motivation and Derivation

To modify an existing or to produce a new successful CG parameter, two factors must be considered:

- The new modified CG parameter must satisfy (6), which will give way to global convergence.
- It must outperform the existing methods computationally by carrying out numerical comparisons with the relevant existing methods in the literature.

In [31], Yu and Guan presented a modified version of the DY CG parameter as follows:

$$\beta_k^{DDY} = \beta_k^{DY} - \frac{b\|F_k\|^2}{(d_{k-1}^T y_{k-1})^2} d_{k-1}^T F_k. \tag{8}$$

In the effort to prove the convergence of the method, they showed that (6) is satisfied for $\tau = 1 - \frac{1}{4b}$ and $b > \frac{1}{4}$. motivated by Guan's scheme; Yu et al. [29] proposed a spectral DY of the method in [31] and presented the following direction

$$d_k = \frac{1}{\delta_k} F_k + \beta_k^{DSDY} d_{k-1}, \tag{9}$$

where

$$\beta_k^{DSY} = \frac{\|F_k\|^2}{\delta_k (d_{k-1}^T y_{k-1})} - \frac{b\|F_k\|^2}{\delta_k (d_{k-1}^T y_{k-1})^2} d_{k-1}^T F_k. \tag{10}$$

Both the schemes in [31] and [29] satisfy (6) for $b > \frac{1}{4}$ i.e., $b \in (\frac{1}{4}, \infty)$. Moreover, Yu and Guan presented a modified Liu–Storey CG parameter in [31] for solving unconstrained optimization problems. They presented the following

$$\beta_k^{MLS} = -\frac{g_k^T y_{k-1}}{d_{k-1}^T g_{k-1}} - \frac{b\|y_{k-1}\|^2}{(d_{k-1}^T g_{k-1})^2} g_{k-1}^T d_{k-1}, \quad b > \frac{1}{4}. \tag{11}$$

They showed that the scheme with modified LS satisfied equation (6) for $b \in (\frac{1}{4}, \infty)$ only. However, the main challenge with the scheme (11) is that the parameter b is not mathematically derived but is randomly selected in the interval $(\frac{1}{4}, \infty)$,, and the scheme will not satisfy (6) for $b > 0$.

Motivated by the above and the fact that the modified LS scheme for the system of nonlinear equations is limited in the literature, the following modification is proposed:

$$\beta_k^{MLS} = -\xi \frac{F_k^T y_{k-1}}{F_{k-1}^T s_{k-1}} - \xi \frac{b_k \|y_{k-1}\|^2}{(F_{k-1}^T s_{k-1})^2} (F_k^T s_{k-1}), \quad \xi > 0, \tag{12}$$

and $b_k > 0$ is a parameter to be determined.

The notable features of the new scheme are

- The inequality (6) is satisfied throughout the interval $(0, \infty)$.
- The value of the positive parameter b_k, is not fixed, but it is mathematically derived.

Then by (4) and (12), we have

$$d_k = -F_k - \xi \frac{F_k^T y_{k-1}}{s_{k-1}^T F_{k-1}} s_{k-1} - \xi \frac{b_k \|y_{k-1}\|^2 F_k^T s_{k-1}}{(s_{k-1}^T F_{k-1})^2} s_{k-1}. \tag{13}$$

By Perry's idea in [33], (13) can be written as

$$d_k = -Q_k F_k. \tag{14}$$

Here Q_k is called the search direction matrix, which is expressed as

$$Q_k = I + \xi \frac{s_{k-1} y_{k-1}^T}{F_{k-1}^T s_{k-1}} + \xi \frac{b_k \|y_{k-1}\|^2 (s_{k-1} s_{k-1}^T)}{(F_{k-1}^T s_{k-1})^2} \tag{15}$$

Q_k non-symmetric; it shall be symmetrized as follows:

$$H_k = I + \xi \frac{s_{k-1} y_{k-1}^T + y_{k-1} s_{k-1}^T}{F_{k-1}^T s_{k-1}} + \xi \frac{b_k \|y_{k-1}\|^2 s_{k-1} s_{k-1}^T}{(F_{k-1}^T s_{k-1})^2}. \tag{16}$$

Then (14) becomes

$$d_k = -H_k F_k, \tag{17}$$

where H_k is the symmetric version of Q_k in (14).

Theorem 1. *H_k defined in* (16)*, has eigenvalues λ_k^+, λ_k^- and 1 $(n-2)$ times where,*

$$\lambda_k^+ = \frac{1}{2}\Big[(2+2\xi p_k+\xi b_k q_k) + \sqrt{(\xi b_k q_k+2\xi p_k)^2+4\xi^2(q_k-p_k^2)}\Big] \tag{18}$$

$$\lambda_k^- = \frac{1}{2}\Big[(2+2\xi p_k+\xi b_k q_k) - \sqrt{(\xi b_k q_k+2\xi p_k)^2+4\xi^2(q_k-p_k^2)}\Big] \tag{19}$$

with

$$p_k = \frac{y_{k-1}^T s_{k-1}}{F_{k-1}^T s_{k-1}}, \quad q_k = \frac{\|y_{k-1}\|^2\|s_{k-1}\|^2}{(F_{k-1}^T s_{k-1})^2}. \tag{20}$$

Moreover, the eigenvalues are real and positive.

Proof: Let's assume the function F_{k-1} and s_{k-1} are not zero unless the solution is reached; hence, $F_k^T s_{k-1} > 0$. Let V be a vector space spanned by $\{F_{k-1}, s_{k-1}\}$. Then the dimension of V and its orthogonal complement $V^\perp$, are $dim(V) \le 2$ and $dim(V^\perp) \ge n-2$, respectively. Let a set of mutually orthogonal vectors be $\{\iota_{k-1}^i\}_i^{n-2} \subset V^\perp$ satisfying

$$F_{k-1}^T \iota_{k-1}^i = s_{k-1}^T \iota_{k-1}^i = 0, \quad i = 1, ..., n-2. \tag{21}$$

Using (16) and (21) we get

$$H_k \iota_{k-1}^i = \iota_{k-1}^i, \tag{22}$$

The above equation is an eigenvector equation, and ι_{k-1}^i, for $i = 1, 2, ...n-2$ is an eigenvector with a corresponding eigenvalue of 1 up to $(n-2)$ times.
Let the remaining eigenvalue be λ_k^+ and λ_k^-.
Furthermore, (16) can also be written as

$$H_k = I + \xi\frac{y_{k-1}F_k^T}{F_{k-1}^T s_{k-1}} + \xi\left(\frac{b_k\|y_{k-1}\|^2 s_{k-1} + y_{k-1}^T s_{k-1} F_k}{(F_{k-1}^T s_{k-1})^2}\right) s_{k-1}^T. \tag{23}$$

It is clear that (23) is a rank-2 update matrix; therefore, from theorem 1.2.16 of [34], the relation

$$\det(I + a_1 a_2^T + a_3 a_4^T) = \big(1 + a_1^T a_2\big)\big(1 + a_3^T a_4\big) - \big(a_1^T a_4\big)\big(a_2^T a_3\big), \tag{24}$$

where the vectors a_1, a_2, a_3, and a_4 are defined as:

$$a_1 = \xi\frac{s_{k-1}}{F_{k-1}^T s_{k-1}}, \quad a_2 = y_{k-1},$$

$$a_3 = \xi\frac{b_k\|y_{k-1}\|^2 s_{k-1} + F_{k-1}^T s_{k-1} + y_{k-1}}{(F_{k-1}^T s_{k-1})^2}, \quad a_4 = s_{k-1}.$$

can be used to get the determinant matrix of A_k as

$$det(H_k) = \left(\xi\frac{(y_{k-1}^T s_{k-1})}{(F_{k-1}^T s_{k-1})} + 1\right)^2 + \xi\frac{b_k\|y_{k-1}\|^2\|s_{k-1}\|^2}{(F_{k-1}^T s_{k-1})^2} - \xi^2\frac{\|y_{k-1}\|^2\|s_{k-1}\|^2}{(F_{k-1}^T s_{k-1})^2}. \tag{25}$$

Furthermore, it is known that the trace of (16) is equivalent to the total number of its eigenvalues; hence we have

$$\begin{aligned} trace(H_k) &= \underbrace{1+1+\cdots+1}_{(n-2)times} + \lambda_k^+ + \lambda_k^- \\ &= n + 2\xi\frac{y_{k-1}^T s_{k-1}}{F_{k-1}^T s_{k-1}} + \xi\frac{b_k\|y_{k-1}\|^2\|s_{k-1}\|^2}{(F_{k-1}^T s_{k-1})^2} \end{aligned} \tag{26}$$

and consequently,

$$\lambda_k^+ + \lambda_k^- = 2 + 2\xi\frac{y_{k-1}^T s_{k-1}}{F_{k-1}^T s_{k-1}} + \xi\frac{b_k\|y_{k-1}\|^2\|s_{k-1}\|^2}{(F_{k-1}^T s_{k-1})^2}. \tag{27}$$

Then by (20), (25) and (27), we get

$$\lambda_k^2 - (2+2\xi p_k+\xi b_k q_k)\,\lambda_k + \big((\xi p_k+1)^2 + \xi(b_k-\xi)q_k\big) = 0. \tag{28}$$

Then using the quadratic equations formula we obtain

$$\lambda_k^\pm = \frac{1}{2}\left[(2+2\xi p_k+\xi b_k q_k) \pm \sqrt{(\xi b_k q_k+2\xi p_k)^2+4\xi^2(q_k-p_k^2)}\right]. \tag{29}$$

This gives (18) and (19).
Next is to show that λ_k^+ and λ_k^- are real and positive. The discriminant is positive, since $q_k > p_k^2$. Therefore we need to have

$$2 + 2\xi\frac{y_{k-1}^T s_{k-1}}{F_{k-1}^T s_{k-1}} + \xi\frac{b_k\|y_{k-1}\|^2\|s_{k-1}\|^2}{(F_{k-1}^T s_{k-1})^2} > 0. \tag{30}$$

λ_k^+ is real and positive when

$$b_k > -2\frac{(y_{k-1}^T s_{k-1})(F_{k-1}^T s_{k-1})}{\|y_{k-1}\|^2\|s_{k-1}\|^2} - 2\frac{(F_{k-1}^T s_{k-1})^2}{\xi\|y_{k-1}\|^2\|s_{k-1}\|^2}. \tag{31}$$

For λ_k^- to be real and positive, we require equation (28) to be greater than zero, i.e., (19) to be greater than zero.
After algebraic simplification, we get

$$b_k > \xi - \left(\frac{\sqrt{\xi}(y_{k-1}^T s_{k-1})}{\|y_{k-1}\|\|s_{k-1}\|} + \frac{F_{k-1}^T s_{k-1}}{\sqrt{\xi}\|y_{k-1}\|\|s_{k-1}\|}\right)^2. \tag{32}$$

Now (32) can be written as

$$b_k = \xi - \varphi\left(\frac{\sqrt{\xi}(y_{k-1}^T s_{k-1})}{\|y_{k-1}\|\|s_{k-1}\|} + \frac{F_{k-1}^T s_{k-1}}{\sqrt{\xi}\|y_{k-1}\|\|s_{k-1}\|}\right)^2. \tag{33}$$

with $\xi > 0$ and $\varphi \le 0$.
Defining $\lambda_k^- = \eta \in (0,1)$ and multiplying (17) with F_k, we have

$$F_k d_k = -F_k H_k F_k \le -\eta\|F_k\|^2 < 0. \tag{34}$$

Hence (6) is satisfied with $b_k > 0\ \forall k$ ■

To further improve the boundedness and convergence of the proposed algorithm, we employed Waziri et.al. idea [35] and present the following

$$d_k = -F_k + \beta_k^{UMLS} s_{k-1}, \tag{35}$$

where

$$\beta_k^{UMLS} = -\xi\frac{\|F_k\|^2}{\Phi_k} - \xi\frac{b_k\|y_{k-1}\|^2(F_k^T s_{k-1})}{\Phi_k^2}, \tag{36}$$

$$\Phi_k = \max\{F_{k-1}^T s_{k-1}, \xi\|y_{k-1}\|\|s_{k-1}\|\},$$

and

$$b_k = \xi - \varphi\left(\frac{\sqrt{\xi}(y_{k-1}^T s_{k-1})}{U_k} + \frac{F_{k-1}^T s_{k-1}}{V_k}\right)^2, \quad \xi > 0, \quad \varphi \leq 0. \tag{37}$$

$$U_k = \max\{\|y_{k-1}\|\|s_{k-1}\|, \xi\|F_{k-1}\|\|s_{k-1}\|\},$$

$$V_k = \max\{\|F_{k-1}\|\|s_{k-1}\|, \xi\|F_k\|\|s_{k-1}\|\}.$$

Let us define Solodov's projection mapping here.

Definition 1. *Let $\psi \subseteq \mathbb{R}^n$ not empty, be convex, and closed. For $x \in \mathbb{R}^n$, its projection onto ψ is given by*

$$P_\psi = \arg min\{\|x - y\| : y \in \psi\}. \tag{38}$$

The above mapping is a projection map that satisfies

$$\|P_\psi(x) - P_\psi(y)\| \leq \|x - y\|, \quad \text{called nonexpensive,} \tag{39}$$

and finally,

$$\|P_\psi(x) - y\| \leq \|x - y\|, \forall y \in \psi \tag{40}$$

Algorithm 1 Updated Modified (UMLS)

Given $x_0 \in \psi$, $d_0 = -F_0$, $\rho \in (0,1)$, $\sigma > 0$, $\epsilon = 10^{-6}$, and $\zeta > 0$, set $k = 0$.

Step i: find F_k. If $\|F_k\| \leq \epsilon$ stop, otherwise go to Step ii.

Step ii: Let $\alpha_k = \zeta\rho^{m_k}$, where m_k is the least positive integer m such that

$$-F(x_k + \alpha_k d_k)^T d_k \geq \sigma\alpha_k \|F(x_k + \alpha_k d_k)\|\|d_k\|^2. \tag{41}$$

Step iii: Set

$$z_k = x_k + \alpha_k d_k. \tag{42}$$

Step iv: If $z_k \in \psi$ and $\|F(z_k)\| \leq \epsilon$, stop; else, go to Step v.

Step v: Solve for x_{k+1} by

$$x_{k+1} = P_\psi[x_k - \lambda_k F(z_k)], \tag{43}$$

where, $\lambda_k = \dfrac{(x_k - z_k)^T F(z_k)}{\|F(z_k)\|^2}$.

Step vi: Get the new d_{k+1} using (35), (36) and (37).

Step vii: Update $k = k + 1$ and repeat Step ii.

III. Convergence Results

This part presents the analysis of the proposed algorithm **UMLS** and some certain preassumptions.

Assumption 1. *The map F is Lipschitz continuous; that is, there is a constant $L > 0$ such that*

$$\|F(x) - F(y)\| \leq L\|x - y\| \quad \textit{holds for any} \quad x, y \in \mathbb{R}^n. \tag{44}$$

Assumption 2. *F is strongly monotone, i.e., $\forall \quad x, y \in \mathbb{R}^n$ there exists $c > 0$ such that*

$$(x - y)^T (F(x) - F(y)) \geq c\|x - y\|^2, \text{ for all } x, y \in \mathbb{R}^n \tag{45}$$

Assumption 3. *Let $\bar{S}$, nonempty, be the solution set of* (1)

Lemma 1. *Let the assumptions hold, and $\{x_k\}$, $\{z_k\}$ are sequences defined by **UMLS** algorithm, then $\{x_k\}$, $\{z_k\}$ are bounded and*

$$\|d_k\| \leq M. \tag{46}$$

$$\lim_{k\to\infty} \|x_k - z_k\| = 0. \tag{47}$$

$$\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0. \tag{48}$$

Proof: Let $\bar{x} \in \bar{S}$ be a solution of (1), using (45) we have

$$(x_k - \bar{x})^T F(z_k) \geq (x_k - z_k)^T F(z_k). \tag{49}$$

By (41) and definition of z_k

$$(x_k - z_k)^T F(z_k) \geq \sigma\alpha_k^2 \|d_k\|^2 > 0. \tag{50}$$

And from (43) we have

$$\begin{aligned}
\|x_{k+1} - \bar{x}\|^2 &= \|x_k - \lambda_k F(z_k) - \bar{x}\|^2 \\
&= \|x_k - \bar{x}\|^2 - 2\lambda_k (x_k - \bar{x})^T F(z_k) \\
&\quad + \lambda_k^2 \|F(z_k)\|^2 \\
&\leq \|x_k - \bar{x}\|^2 - 2\lambda_k (x_k - z_k)^T F(z_k) \\
&\quad + \lambda_k^2 \|F(z_k)\|^2 \\
&= \|x_k - \bar{x}\|^2 - \frac{((x_k - z_k)^T F(z_k))^2}{\|F(z_k)\|^2} \\
&\leq \|x_k - \bar{x}\|^2.
\end{aligned} \tag{51}$$

Hence,

$$\|x_{k+1} - \bar{x}\| \leq \|x_k - \bar{x}\|. \tag{52}$$

Repeatedly, $\|x_k - \bar{x}\| \leq \|x_0 - \bar{x}\| \ \forall k$.

This means that $\{x_k - \bar{x}\}$ is decreasing and $\{x_k\}$ is bounded. Furthermore, by (1)-(3) and (52), we get

$$\|F_k\| = \|F_k - F_k(\bar{x})\| \leq L\|x_k - \bar{x}\| \leq \|x_0 - \bar{x}\|. \tag{53}$$

Hence

$$\|F_k\| \leq N_1 \quad \text{for} \quad N_1 = \|x_0 - \bar{x}\|. \tag{54}$$

Then from (37)

$$\begin{aligned}
|b_k| &= \left|\xi - \varphi\left(\frac{\sqrt{\xi}(y_{k-1}^T s_{k-1})}{U_k} + \frac{(F_{k-1}^T s_{k-1})}{V_k}\right)^2\right| \\
&\leq |\xi| + |\varphi|\left|\left(\frac{\sqrt{\xi}\|y_{k-1}\|\|s_{k-1}\|}{U_k} + \frac{\|F_{k-1}\|\|s_{k-1}\|}{V_k}\right)^2\right| \\
&\leq \xi + |\varphi|\left|\left(\frac{\sqrt{\xi}\|y_{k-1}\|\|s_{k-1}\|}{\|y_{k-1}\|\|s_{k-1}\|} + \frac{\|F_{k-1}\|\|s_{k-1}\|}{\|F_{k-1}\|\|s_{k-1}\|}\right)^2\right| \\
&= \xi + |\varphi|\left(\sqrt{\xi} + 1\right)^2.
\end{aligned} \tag{55}$$

This implies

$$|b_k| \leq \lambda \quad \text{for} \quad \lambda = \xi + |\varphi|\left(\sqrt{\xi} + 1\right)^2. \tag{56}$$

And from (35), (36), (37), (54) and (56), we arrive

$$\begin{aligned}
\|d_k\| &= \left\|-F_k + \beta_k^{UMLS} s_{k-1}\right\| \\
&= \left\| - F_k + \left(\frac{\xi(F_k^T y_{k-1})}{\Phi_k} - \frac{\xi b_k \|y_{k-1}\|^2 (F_k^T s_{k-1})}{\Phi_k^2}\right) s_{k-1}\right\| \\
&\leq \|F_k\| + \frac{|\xi|\|F_k\|\|s_{k-1}\|\|y_{k-1}\|}{\Phi_k} + \frac{|\xi||b_k|\|F_k\|\|y_{k-1}\|^2\|s_{k-1}\|^2}{\Phi_k^2} \\
&\leq \|F_k\| + \|F_k\| + \frac{1}{\xi}\lambda\|F_k\| \leq (2 + \frac{1}{\xi}\lambda)N_1.
\end{aligned} \tag{57}$$

Let $M = (2 + \frac{1}{\xi}\lambda)N_1$, (46) is proved.
Since $\{d_k\}$, $\{x_k\}$ are bounded sequences, from (42) we have that $\{z_k\}$ is bounded. Using the same argument as in (53) we have

$$\|F(z_k)\| \leq \ell, \quad \ell \quad \text{is a positive constant.} \tag{58}$$

From (51),

$$\left((x_k - z_k)^T F(z_k)\right)^2 \leq \|F_k\|^2 \left(\|x_k - \bar{x}\|^2 - \|x_{k+1} - \bar{x}\|^2\right). \tag{59}$$

From (41), we get

$$\sigma^2 \alpha_k^4 \|d_k\|^4 \leq \alpha_k^2 (F(z_k)^T d_k)^2. \tag{60}$$

By (51) and (58), we have

$$\sigma^2 \alpha_k^4 \|d_k\|^4 \leq \|F(z_k)\|^4 (\|x_k - \bar{x}\|^2 - \|x_{x+1} - \bar{x}\|^2), \tag{61}$$

$\{x_k - \bar{x}\}$ converged and $\{F(z_k)\}$ is bounded, we can apply limit on (59) to get

$$\sigma^2 \lim_{k\to\infty} \alpha_k^4 \|d_k\|^4 \leq 0. \tag{62}$$

and hence

$$\lim_{k\to\infty} \alpha_k d_k = 0. \tag{63}$$

(61) and (42) indicates that (47) is true. We can, however, deduce from the definition of λ_k that

$$\begin{aligned} \|x_{k+1} - x_k\| &= \|x_k - \lambda_k F(z_k) - x_k\| \\ &= \|\chi_k F(z_k)\| \\ &\leq \|x_k - z_k\|. \end{aligned} \tag{64}$$

This means (48) is proved. ■

Theorem 2. *Lets assumptions* (1)-(3) *be true and* $\{x_k\}$ *be a sequence defined by* ***UMLS*** *algorithm, then*

$$\liminf_{k\to\infty} \|F_k\| = 0. \tag{65}$$

Proof: Let us starts by contradicting the above assumptions i.e., (65) not true, then

$$\|F_k\| \geq \epsilon_0 \quad \text{is true} \forall k > 0. \tag{66}$$

Let $\|d_k\| \neq$ unless when $\|F_k\| = 0$, then there exists $\delta_1 > 0$ such that

$$\|d_k\| \geq \delta_1. \tag{67}$$

If $\alpha_k \neq \xi$, then α_k, $\rho^{-1}\alpha_k$ does not agrees (41) i.e.,

$$-F(x_k + \rho^{-1}\alpha_k d_k)^T d_k < \sigma\rho^{-1}\alpha_k \|F(x_k + \rho^{-1}\alpha_k d_k)\| \|d_k\|^2. \tag{68}$$

Using (68) and (6) yields

$$\begin{aligned} \tau\|F_k\|^2 &\leq -F_k^T d_k \\ &= \left(F(x_k + \rho^{-1}\alpha_k d_k) - F_k\right)^T d_k \\ &\quad - F_k\left(x_k + \rho^{-1}\alpha_k d_k\right)^T d_k \\ &\leq L\rho^{-1}\alpha_k \|d_k\|^2 + \sigma\rho^{-1}\alpha_k \|d_k\|^2 \\ &= \alpha_k \|d_k\| (L + \sigma)\rho^{-1} \|d_k\|. \end{aligned} \tag{69}$$

And from (69) we get

$$\alpha_k \|d_k\| > \frac{\rho}{L + \sigma} \frac{\tau \|F_k\|^2}{\|d_k\|} \geq \frac{\rho}{L + \sigma} \frac{\tau \delta_0}{M}. \tag{70}$$

Equation (70) contradicts (63). Hence (65) is true. This completes the proof. ■

IV. Numerical Experiments

This section is divided into two parts, the first and the second. In this part, numerical results of the proposed method is presented to highlight the development achieved by the proposed algorithm. The following algorithms were used for the comparisons:

- MDDYM (2021) Algorithm proposed in [35].
- DTCG1 (2020) Algorithm proposed in [9].

In the second part, UMLS Algorithm is applied to solve problems arising in compressive sensing. The Algorithm is compared with CHCG Algorithm for signal reconstruction and recovery. Below are the parameters used for the UMLS Algorithm: The codes are written in MaTlab 8.3.0 (R2014a) and run on a laptop computer with the following specifications: 1.80 GHz, processor (CPU), and 8 RAM (GB). However, the parameters of MDDYM and DTCG1 Algorithms, were chosen as in [35] and [9], while the parameters for UMLS were chosen to be $\xi = 1$, $\sigma = \varphi = 10^{-4}$ and $\rho = \zeta = 0.9$. The iterations of all the three algorithms were stop when $\|F_k\| \leq 10^{-6}$, $\|F(z_k)\| \leq 10^{-6}$, or the number of iterations exceed 2000 but stopping criteria is not satisfied. The numerical experiments of the three methods were carried out using the following ten test problems (T1-T6), and with the following initial points:
$x_0^1 = (0.01, 0.01, ..., 0.01)^T$, $x_0^2 = (0.25, 0.25, ..., 0.25)^T$, $x_0^3 = (0.4, 0.4, ..., 0.4)^T$, $x_0^4 = (0.5, 0.5, ..., 0.5)^T$, $x_0^5 = (1.25, 1.25, ..., 1.25)^T$, $x_0^6 = (0.3, 0.3, ..., 0.3)^T$, $x_0^7 = (1, 1, ..., 1)^T$, and $x_0^8 = (0.1, 0.1, ..., 0.1)^T$.

T1 [9]
$F_1(x) = e^{x_1} - 1$,
$F_i(x) = e^{x_i} + x_{i-1} - 1, \quad i = 2, 3, 4, \ldots, n-1$,
where $\psi = \mathbb{R}_+^n$.

T2 [10]
$F_i(x) = log(x_i + 1) - \frac{x_i}{n}, \quad i = 2, 3, \ldots, n-1$,
where $\psi = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i \leq n, x_i > -1,$
$i = 1, 2, 3, \ldots, n\}$.

T3 [35]
$F_i(x) = 2x_i - \sin|x_i|, \quad i = 1, 2, 3, \ldots, n$,
where $\psi = \mathbb{R}_+^n$.

T4[10]
$F_i(x) = \cos x_i + x_i - 1. \quad i = 1, 2, 3, \ldots, n$,
where $\psi = \mathbb{R}_+^n$.

T5 [9]
$F_i(x) = e^{x_i} - 1, \quad i = 1, 2, 3, \ldots, n$,
where $\psi = \mathbb{R}_+^n$.

T6 [35]
$F_1(x) = -2x_1 - x_2 + e^{x_1} - 1$,
$F_i(x) = -x_{i-1} + 2x_i - x_{i+1} + e^{x_i} - 1$,
$F_n(x) = -x_{n-1} + 2x_n + e^{x_n} - 1, \quad i = 2, 3, 4, \ldots, n-1$,
where $\psi = \{x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i \leq n, x_i \geq 0, \quad i = 1, 2, 3, \ldots, n\}$.

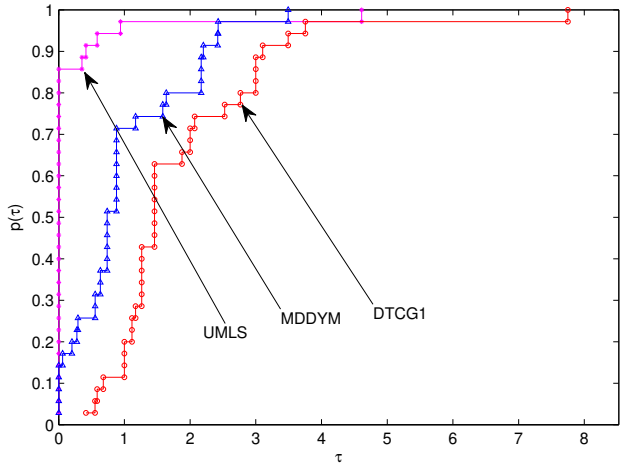


Fig. 1: Performance profile for the number of iterations.

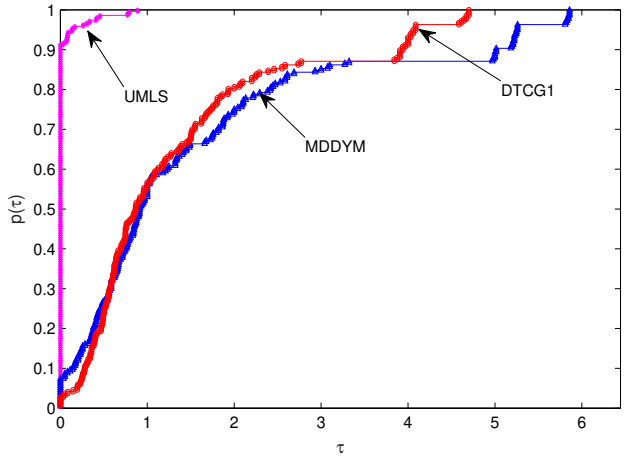


Fig. 3: Performance profile for the CPU Time (in seconds).

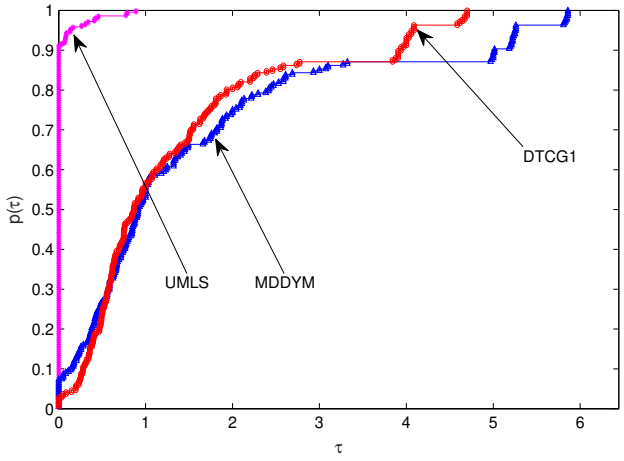


Fig. 2: Performance profile for the function evaluations.

Tables II-IV display the performance of the proposed algorithm (UMLS) in comparison with recent CG algorithms; MDDYM [35] and DTCG1 [9]. In the tables, the terms IP, Iter, and Fval refer to the initial point, number of iterations, and function evaluations respectively. Also, Time(s) and $\|F_k\|$ refers to CPU time and norm of residual functions respectively.

To be just a comparison, the two algorithms were run with the same initial starting point. The large dimension, i.e., (100,000), indicates the proposed algorithm's applicability to practical problems.

As it is observable from the tables, the UMLS algorithm solves all the test functions with the least number of iterations, least CPU time, and least number of function evaluations as compared to the DTCG1 and MDDYM algorithms. Based on the above, the UMLS algorithm outperformed the compared algorithms and can be used for practical purposes where large-scale systems are involved.

Furthermore, using Dolan and Moré [36] performance profile, data of tables II-IV were used to plot Figures 1-3. Each Figure compares the proposed method with that of MYYDM and DTCG1 algorithms based on the selected metric.

Figure 1 displays the comparison based on the number of iterations. In this Figure, it is clear that the UMLS algorithm wins over MYYDM algorithm in about 80% of the entire experiment. The same thing happened with Figure 2 where UMLS algorithm wins over MYYDM algorithm in about 53% of the entire experiment based on function evaluations. The same trend with Figure 3. Therefore, based on the above, the proposed algorithm (UMLS) could be relied on for solving system (1) and extending to compressive sensing. Furthermore, using Dolan and Moré performance profile, the data in tables II-IV were used to plot figures 1-3. Each figure displays the performance based on the criteria used i.e., number of iterations, CPU and number of function evaluations.

A. Application of UMLS Algorithm to signal recovery

Among the applications of constrained monotone nonlinear systems of equations is signal and image recovery in compressive sensing (CS). Compressing sensing or compressed sensing, concerns with the reconstruction of a sparse signal from incoherent measurements. It asserts that some signals in real applications have a better representation in a specific domain after transformation while the remaining ones are insignificant or zeros. It replaces the traditional sampling technique, thereby reducing the number of samples for better signal sampling and reconstruction. The technique depends mainly on mathematical algorithms, especially derivative-free based algorithms that need less storage facility and a high chance of solving large-scale systems. Therefore, optimization procedures that aim at the sparsest solution in a suitable representation will be a good candidate for the job. The CS theorem is present in biological engineering, medical sciences, sciences and engineering [37], [38]. It uses the following linear equation to recover the sparse signal

$$\phi = Nx, \tag{71}$$

$x \in \mathbb{R}^n$, $\phi \in \mathbb{R}^m$, $N \in \mathbb{R}^{mxn}$ $(m << n)$ is a linear operator. The popular method for solving the sparse solution is to

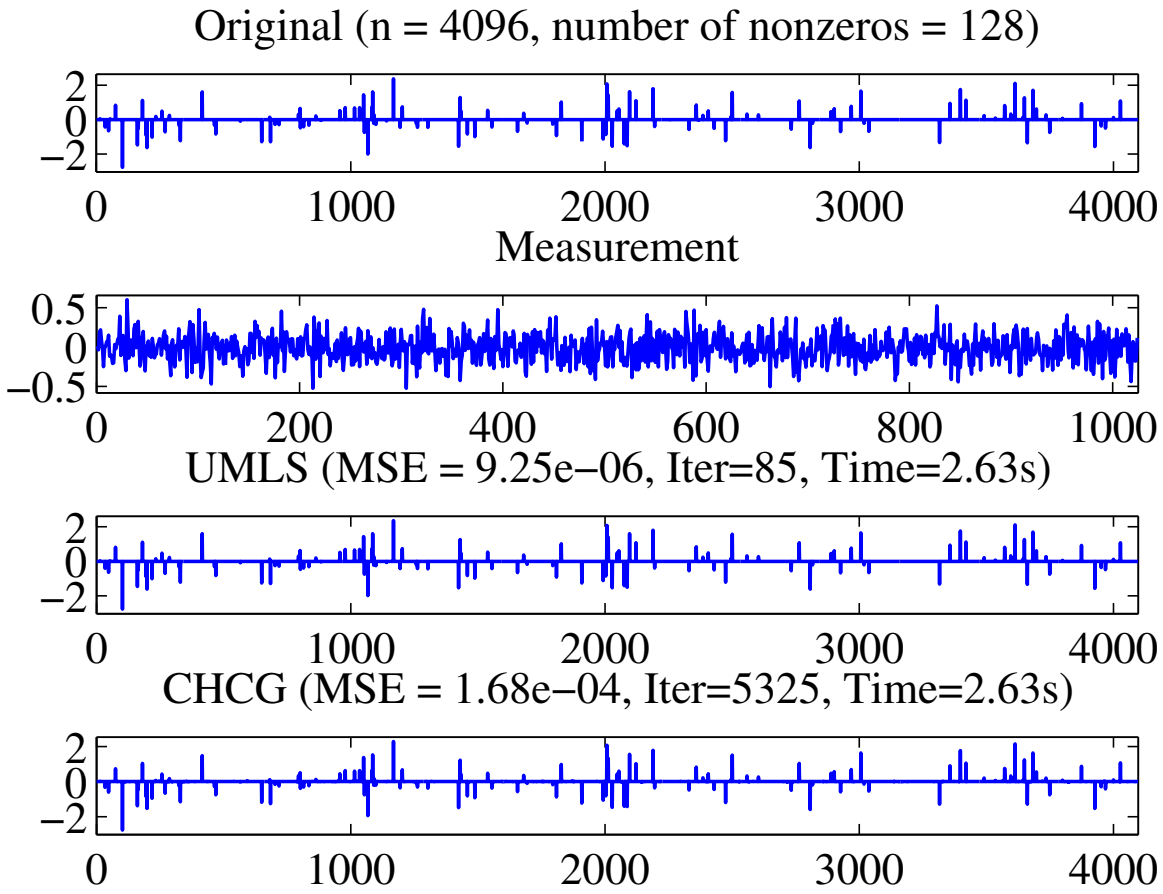


Fig. 4: Displays, from top to bottom, the original signal, the measurement, and the recovery signals recovered by the UMLS and CHCG algorithms.

minimize the relation below

$$\min_x \frac{1}{2}\|\phi - Nx\|_2^2 + \tau\|x\|_1, \tag{72}$$

τ is a positive parameter (regularized).
Equation (72) is a non-smooth due to the presence of ℓ_1-norm therefore, derivative-free algorithms can not be applied to solve it. As such, gradient projection for sparse reconstruction (GPSR) proposed by Figueredo et al. [39] is among the best approach for solving (72). In [39], problem (72) is written as

$$x = u - v, \quad u \geq 0, \quad v \geq 0, \quad u, v \in \mathbb{R}^n, \tag{73}$$

$u_i = (x_i)_+$, and $v_i = (-x_i)_+$ for $i = 1, 2, ..., n$, where $(.)_+$ denotes positive-part operator, which is defined as $(x)_+ = \max\{0, x\}$. Using ℓ_1-norm definition, it implies $\|x\|_1 = e_n^T u + e_n^T v$, with $e_n = (1, 1, ..., 1)^T \in \mathbb{R}^n$ and problem (72) become

$$\min_{u,v} \frac{1}{2}\|\phi - N(u - v)\|_2^2 + \tau e_n^T u + \tau e_n^T v, \quad u, v \geq 0. \tag{74}$$

Figueredo et al., [39] demonstrates how to express problem (74) in a more formal bound quadratic programming problem as

$$\min_z \frac{1}{2} z^T H z + c^T z, \quad \text{s.t} \quad z \geq 0, \tag{75}$$

where $z = \begin{pmatrix} u \\ v \end{pmatrix}$, $c = \tau e_{2n} + \begin{pmatrix} -b \\ b \end{pmatrix}$, $b = N^T\phi$,

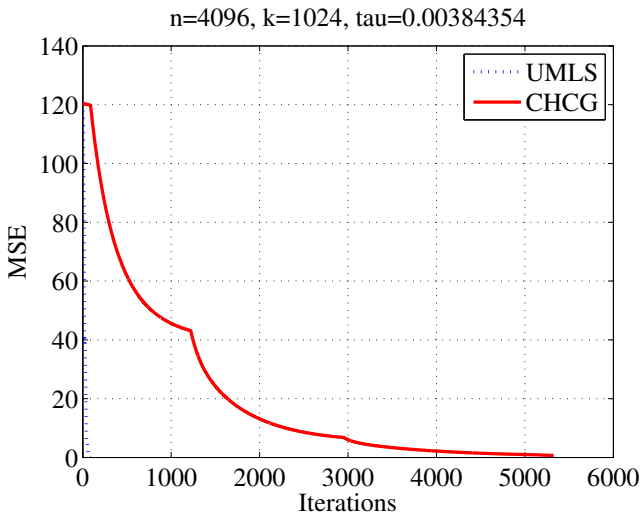
$H = \begin{pmatrix} B^T B & -B^T B \\ -B^T B & B^T B \end{pmatrix}$.
The matrix H is a positive semi-definite. Equation (75) can be transformed to a linear variable inequality (LVT), which is the same thing to

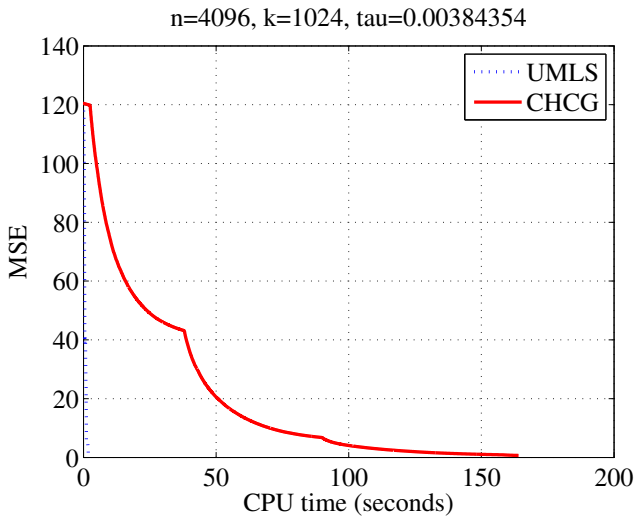
$$F(z) = \min\{z, Hz + c\} = 0. \tag{76}$$

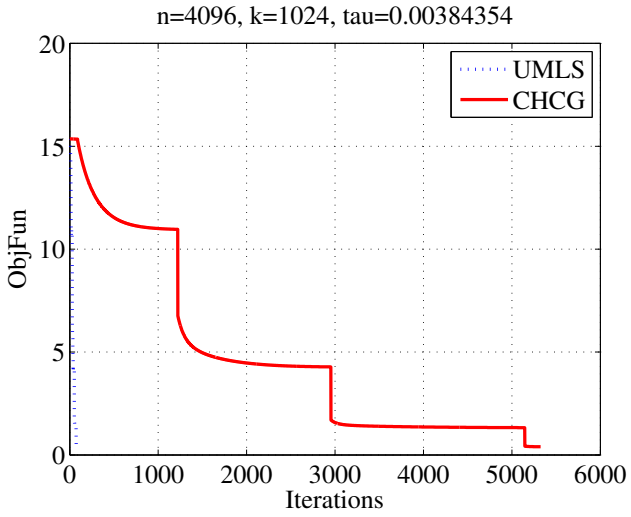
The function F_k in (75) is a vector-valued function, it was shown to be Lipschitz continuous and monotone in Lemma 3 of [40] and Lemma 2.2 of [41]. Hence, our proposed algorithm, UMLS algorithm, could be a good choice for solving it.
Numerical comparison is carried out in comparison with CHCG algorithm in signal reconstruction and recovery. The size of the signal is considered to be $n = 2^{12}$ with $k = 2^{10}$. The Error, termed as means square error (MSE) is used to assess the quality of the restoration process, it is measured by

$$MSE = \frac{1}{n}\|\hat{x} - \tilde{x}\|^2, \tag{77}$$

where $\hat{x}$ refers to the original signal, and $\tilde{x}$ is the recovered signal. The experiment starts by initializing $x_0 = N^T\phi$,







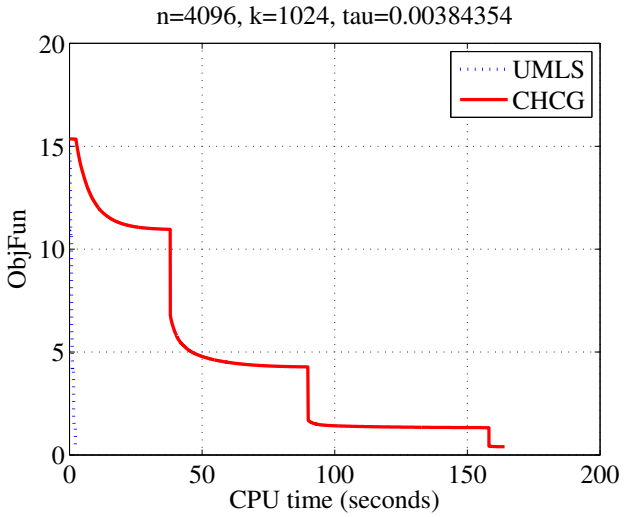


Fig. 5: Compares UMLS and CHCG Algorithm in signal recovery experiment. In above figure, x-axes represent the number of iterations, y-axes represent MSE, and the number of function evaluations.

Fig. 6: Compares UMLS and CHCG Algorithm in signal recovery experiment. In above figure, x-axes represent CPU time (seconds), and y-axes represent MSE and the number of function evaluations.

conducting ten (10) trials recording the data of interest for each trial. It set to stop when $\frac{|f_k - f_{k-1}|}{|f_{k-1}|} < 10^{-5}$, where $f(x) = \frac{1}{2}\|Nx - \phi\|_2^2 + \tau\|x\|_1$ is a merit function. The following information is recorded:

As seen from the Figures and the table, both algorithms performed well in recovering the signals with a reasonable degree of accuracy. But considering the number of iterations (ITER) and the TIMES (CPU times), the proposed algorithm is faster and more efficient than the compared algorithm. Then it is evident that UMLS algorithm is an excellent algorithm to be applied to ℓ_1-norm minimization and signal recovery.

TABLE I: Ten trials recorded of UMLS and CHCG signal's recovery with ℓ_1-norm regularization

	UMLS			CHCG		
	MSE	ITER	TIME(s)	MSE	ITER	TIME(s)
	2.13E-04	91	2.76	2.30E-05	137	2.06
	8.16E-06	92	2.52	1.96E-05	141	2.06
	8.11E-06	89	2.6	2.38E-05	121	3.55
	2.09E-04	91	2.52	2.38E-05	135	2.98
	4.62E-05	84	2.34	1.62E-05	140	3.09
	7.70E-06	96	2.88	1.85E-05	125	3.77
	2.72E-05	87	2.72	4.31E-05	135	3.16
	1.09E-05	84	2.53	1.94E-05	147	3.17
	0.74E-05	75	2.25	2.10E-04	108	0.41
	1.19E-05	72	2.53	2.71E-05	122	4.53
AVERAGE	3.62E-05	92.8	2.556	4.24E-05	132	2.878

V. Conclusion

This paper presented a modified Liu and Storey algorithm for the constrained monotone nonlinear equation. The algorithm satisfied the descent condition by conducting an eigenvalue analysis of the search direction matrix. Unlike methods presented by Yu and Guan [31], and YU et al. [32], which assured global convergence only for $C \in (\frac{1}{4}, \infty)$ interval, the proposed method converges globally in the set $C \in (0, \infty)$. The impact of this development is clearer in the comparison section. Being fast and robust, the proposed algorithm is extended and outperforms the CHCG algorithm in signal recovery in compressive sensing. In the future, more classical CGs' should be utilized, analyzed in similar fashion and prove the global convergence.

TABLE II: Numerical of UMLS, MDDYM & DTCG1 methods for problems 1 and 2

			UMLS				MDDYM				DTCG1			
Problem	Dimension	IP	Iter	Fval	Time(s)	$\|F_k\|$	Iter	Fval	Time(s)	$\|F_k\|$	Iter	Fval	Time(s)	$\|F_k\|$
1	1000	x_0^1	4	6	0.02721	$1.65\text{x}10^{-07}$	15	47	0.05013	$1.85\text{x}10^{-07}$	–	–	–	–
		x_0^2	5	7	0.01554	$1.85\text{x}10^{-07}$	12	37	0.02560	$1.65\text{x}10^{-07}$	–	–	–	–
		x_0^3	5	7	0.01107	$2.37\text{x}10^{-07}$	14	44	0.01818	$1.65\text{x}10^{-07}$	37	181	0.05749	$6.96\text{x}10^{-07}$
		x_0^4	6	8	0.01787	$9.07\text{x}10^{-07}$	14	46	0.01873	$1.65\text{x}10^{-07}$	18	133	0.03628	0
		x_0^5	5	7	0.01478	$1.65\text{x}10^{-07}$	16	51	0.02003	$1.65\text{x}10^{-07}$	29	321	0.07102	0
		x_0^6	5	7	0.01571	$1.65\text{x}10^{-07}$	13	43	0.01832	$1.65\text{x}10^{-07}$	29	318	0.04908	0
		x_0^7	5	7	0.01673	$1.65\text{x}10^{-07}$	11	33	0.01508	$1.65\text{x}10^{-07}$	–	–	–	–
		x_0^8	6	7	0.01700	$1.65\text{x}10^{-07}$	18	58	0.02026	$6.96\text{x}10^{-07}$	8	46	0.01811	0
	10,000	x_0^1	4	6	0.06954	$1.65\text{x}10^{-07}$	14	43	0.08175	$8.65\text{x}10^{-07}$	103	1155	1.09874	0
		x_0^2	6	7	0.06813	$1.65\text{x}10^{-07}$	13	42	0.08544	$1.72\text{x}10^{-07}$	32	284	0.27656	0
		x_0^3	6	7	0.07684	$4.72\text{x}10^{-07}$	6	15	0.03590	$4.90\text{x}10^{-07}$	16	122	0.14851	0
		x_0^4	6	7	0.07881	$7.14\text{x}10^{-07}$	6	15	0.03468	$1.65\text{x}10^{-07}$	92	1015	1.17238	0
		x_0^5	6	8	0.09135	$5.55\text{x}10^{-08}$	11	33	0.07394	$2.99\text{x}10^{-07}$	25	45	0.10508	$9.34\text{x}10^{-07}$
		x_0^6	6	7	0.06874	$5.45\text{x}10^{-}07$	10	28	0.05651	$2.13\text{x}10^{-07}$	92	1030	1.09964	0
		x_0^7	6	8	0.08200	$4.72\text{x}10^{-07}$	8	23	0.05284	$8.54\text{x}10^{-07}$	9	60	0.09466	0
		x_0^8	6	7	0.07325	$1.65\text{x}10^{-07}$	17	61	0.10574	$9.75\text{x}10^{-07}$	–	–	–	–
	100,000	x_0^1	4	6	0.40163	$9.07\text{x}10^{-07}$	16	46	0.60481	$4.68\text{x}10^{-07}$	30	199	1.86071	0
		x_0^2	6	8	0.55833	$7.15\text{x}10^{-07}$	10	27	0.42738	$1.28\text{x}10^{-07}$	28	286	2.22426	0
		x_0^3	6	8	0.61761	$6.96\text{x}10^{-07}$	7	17	0.27876	$1.13\text{x}10^{-07}$	101	1163	8.61800	0
		x_0^4	6	7	0.51188	$7.15\text{x}10^{-07}$	6	15	0.25381	$9.96\text{x}10^{-07}$	26	43	0.64096	$8.31\text{x}10^{-07}$
		x_0^5	6	8	0.59577	$1.65\text{x}10^{-07}$	10	31	0.42040	$5.01\text{x}10^{-07}$	29	72	0.86860	$7.15\text{x}10^{-07}$
		x_0^6	6	8	0.54648	$7.15\text{x}10^{-07}$	8	21	0.34679	$4.92\text{x}10^{-07}$	27	68	0.81840	$5.14\text{x}10^{-07}$
		x_0^7	6	8	0.57418	$1.49\text{x}10^{-07}$	8	23	0.35909	$9.48\text{x}10^{-07}$	29	71	0.84687	$5.14\text{x}10^{-07}$
		x_0^8	6	7	0.48058	$1.91\text{x}10^{-07}$	11	34	0.45743	$7.74\text{x}10^{-07}$	9	58	0.51423	0
2	1000	x_0^1	2	4	0.00658	$1.54\text{x}10^{-07}$	5	11	0.01064	$8.95\text{x}10^{-07}$	18	20	0.01990	$6.96\text{x}10^{-07}$
		x_0^2	3	5	0.00831	$1.16\text{x}10^{-07}$	7	15	0.01153	$1.74\text{x}10^{-07}$	23	25	0.03049	$6.96\text{x}10^{-07}$
		x_0^3	3	5	0.00732	$1.07\text{x}10^{-07}$	6	12	0.01029	$9.99\text{x}10^{-07}$	24	26	0.02492	$5.14\text{x}10^{-07}$
		x_0^4	3	5	0.00802	$3.71\text{x}10^{-07}$	14	38	0.01838	$9.56\text{x}10^{-07}$	24	26	0.02431	$7.58\text{x}10^{-07}$
		x_0^5	4	6	0.00754	$1.80\text{x}10^{-07}$	9	18	0.01394	$1.10\text{x}10^{-07}$	26	28	0.03217	$8.44\text{x}10^{-07}$
		x_0^6	3	5	0.00730	$2.58\text{x}10^{-07}$	10	26	0.01591	$8.72\text{x}10^{-07}$	23	25	0.02636	$7.58\text{x}10^{-07}$
		x_0^7	4	6	0.00921	$2.90\text{x}10^{-07}$	14	30	0.01793	$7.75\text{x}10^{-07}$	26	28	0.02475	$5.14\text{x}10^{-07}$
		x_0^8	2	4	0.00655	$4.43\text{x}10^{-07}$	9	25	0.01563	$5.40\text{x}10^{-07}$	21	23	0.02674	$8.44\text{x}10^{-07}$
	10,000	x_0^1	1	3	0.01447	$6.61\text{x}10^{-07}$	5	11	0.03786	$2.71\text{x}10^{-07}$	19	21	0.10785	$9.45\text{x}10^{-07}$
		x_0^2	3	5	0.02728	$5.76\text{x}10^{-07}$	7	15	0.04710	$5.14\text{x}10^{-07}$	24	26	0.14158	$9.45\text{x}10^{-07}$
		x_0^3	5	7	0.04079	$1.85\text{x}10^{-07}$	7	13	0.04656	$8.67\text{x}10^{-07}$	25	27	0.12629	$8.44\text{x}10^{-07}$
		x_0^4	5	7	0.03502	$8.71\text{x}10^{-07}$	14	38	0.10976	$1.68\text{x}10^{-07}$	26	28	0.14233	$5.86\text{x}10^{-07}$
		x_0^5	6	8	0.05201	$3.53\text{x}10^{-07}$	9	18	0.05696	$4.65\text{x}10^{-07}$	28	30	0.14684	$6.96\text{x}10^{-07}$
		x_0^6	4	6	0.03174	$2.27\text{x}10^{-07}$	11	27	0.07181	$7.88\text{x}10^{-07}$	25	27	0.12188	$5.86\text{x}10^{-07}$
		x_0^7	5	7	0.03385	$2.68\text{x}10^{-07}$	15	32	0.09280	$3.92\text{x}10^{-07}$	27	29	0.14370	$8.68\text{x}10^{-07}$
		x_0^8	5	7	0.04305	$1.09\text{x}10^{-07}$	12	32	0.08737	$5.56\text{x}10^{-07}$	23	25	0.12935	$6.96\text{x}10^{-07}$
	100,000	x_0^1	5	7	0.27125	$1.56\text{x}10^{-07}$	5	11	0.26042	$8.52\text{x}10^{-07}$	21	23	0.71941	$7.62\text{x}10^{-07}$
		x_0^2	3	5	0.19059	$7.89\text{x}10^{-07}$	8	16	0.38236	$2.72\text{x}10^{-07}$	26	28	0.86779	$7.62\text{x}10^{-07}$
		x_0^3	5	7	0.24291	$5.19\text{x}10^{-07}$	7	13	0.31723	$2.70\text{x}10^{-07}$	27	29	0.89661	$6.96\text{x}10^{-07}$
		x_0^4	6	8	0.28945	$2.61\text{x}10^{-07}$	17	45	0.80638	$5.08\text{x}10^{-07}$	27	29	0.90785	$9.36\text{x}10^{-07}$
		x_0^5	7	9	0.32860	$1.03\text{x}10^{-07}$	10	19	0.43548	$3.93\text{x}10^{-07}$	30	32	1.01914	$5.86\text{x}10^{-07}$
		x_0^6	4	6	0.21091	$5.14\text{x}10^{-07}$	11	27	0.54315	$2.47\text{x}10^{-07}$	26	28	0.87760	$9.36\text{x}10^{-07}$
		x_0^7	5	7	0.23726	$6.96\text{x}10^{-07}$	15	32	0.65560	$2.94\text{x}10^{-07}$	29	31	0.97489	$6.96\text{x}10^{-07}$
		x_0^8	5	7	0.28842	$3.29\text{x}10^{-07}$	12	32	0.62117	$1.73\text{x}10^{-07}$	25	27	0.84362	$5.86\text{x}10^{-07}$

TABLE III: Numerical of UMLS, MDDYM & DTCG1 methods for problems 3 and 4

			UMLS				MDDYM				DTCG1			
Problem	Dimension	IP	Iter	Fval	Time(s)	$\|F_k\|$	Iter	Fval	Time(s)	$\|F_k\|$	Iter	Fval	Time(s)	$\|F_k\|$
3	1000	x_0^1	5	7	0.00853	$3.14\text{x}10^{-07}$	5	11	0.00963	$7.68\text{x}10^{-07}$	18	20	0.01455	$6.03\text{x}10^{-07}$
		x_0^2	6	8	0.00937	$7.10\text{x}10^{-07}$	4	6	0.00599	$5.22\text{x}10^{-07}$	22	24	0.01911	$9.29\text{x}10^{-07}$
		x_0^3	6	8	0.01021	$9.55\text{x}10^{-07}$	4	6	0.00669	$1.27\text{x}10^{-07}$	23	25	0.01698	$7.28\text{x}10^{-07}$
		x_0^4	6	8	0.00938	$9.86\text{x}10^{-07}$	4	6	0.00663	$2.71\text{x}10^{-07}$	23	25	0.01919	$8.92\text{x}10^{-07}$
		x_0^5	7	9	0.00968	$3.66\text{x}10^{-07}$	9	21	0.01272	$3.02\text{x}10^{-07}$	24	26	0.02035	$8.51\text{x}10^{-07}$
		x_0^6	6	8	0.00941	$8.14\text{x}10^{-07}$	4	6	0.00621	$9.44\text{x}10^{-07}$	23	25	0.01856	$5.54\text{x}10^{-07}$
		x_0^7	7	9	0.00957	$1.92\text{x}10^{-07}$	9	23	0.01211	$4.88\text{x}10^{-07}$	24	26	0.01970	$7.63\text{x}10^{-07}$
		x_0^8	6	8	0.00856	$3.09\text{x}10^{-07}$	3	5	0.00596	$3.80\text{x}10^{-07}$	21	23	0.02028	$7.52\text{x}10^{-07}$
	10,000	x_0^1	5	7	0.03517	$9.93\text{x}10^{-07}$	5	11	0.02923	$2.43\text{x}10^{-07}$	19	21	0.06630	$9.54\text{x}10^{-07}$
		x_0^2	7	9	0.04194	$2.24\text{x}10^{-07}$	4	6	0.02085	$1.65\text{x}10^{-07}$	24	26	0.09151	$7.35\text{x}10^{-07}$
		x_0^3	7	9	0.03727	$3.01\text{x}10^{-07}$	4	6	0.02221	$4.02\text{x}10^{-07}$	25	27	0.08422	$5.75\text{x}10^{-07}$
		x_0^4	7	9	0.03848	$3.11\text{x}10^{-07}$	4	6	0.02084	$8.56\text{x}10^{-07}$	25	27	0.08742	$7.05\text{x}10^{-07}$
		x_0^5	8	10	0.04967	$1.15\text{x}10^{-07}$	9	21	0.04244	$9.55\text{x}10^{-07}$	26	28	0.08726	$6.73\text{x}10^{-07}$
		x_0^6	7	9	0.03930	$2.57\text{x}10^{-07}$	4	6	0.02077	$2.99\text{x}10^{-07}$	24	26	0.07492	$8.76\text{x}10^{-07}$
		x_0^7	7	9	0.04167	$6.07\text{x}10^{-07}$	10	24	0.05214	$5.62\text{x}10^{-07}$	26	28	0.08787	$6.03\text{x}10^{-07}$
		x_0^8	6	8	0.04243	$9.77\text{x}10^{-07}$	3	5	0.01828	$1.20\text{x}10^{-07}$	23	25	0.07662	$5.95\text{x}10^{-07}$
	100,000	x_0^1	6	8	0.21705	$3.14\text{x}10^{-07}$	5	11	0.18513	$7.68\text{x}10^{-07}$	21	23	0.47274	$7.54\text{x}10^{-07}$
		x_0^2	7	9	0.29739	$7.09\text{x}10^{-07}$	4	6	0.12993	$5.22\text{x}10^{-07}$	26	28	0.53435	$5.81\text{x}10^{-07}$
		x_0^3	7	9	0.26404	$9.53\text{x}10^{-07}$	5	7	0.14596	$3.34\text{x}10^{-07}$	26	28	0.55658	$9.10\text{x}10^{-07}$
		x_0^4	7	9	0.28047	$9.85\text{x}10^{-07}$	4	6	0.12700	$2.71\text{x}10^{-07}$	27	29	0.55989	$5.58\text{x}10^{-07}$
		x_0^5	8	10	0.29058	$3.65\text{x}10^{-07}$	10	22	0.35633	$9.62\text{x}10^{-07}$	28	30	0.56943	$5.32\text{x}10^{-07}$
		x_0^6	7	9	0.30547	$8.13\text{x}10^{-07}$	4	6	0.13119	$9.44\text{x}10^{-07}$	26	28	0.56150	$6.93\text{x}10^{-07}$
		x_0^7	8	10	0.27400	$1.92\text{x}10^{-07}$	10	24	0.35296	$1.78\text{x}10^{-07}$	27	29	0.55055	$9.53\text{x}10^{-07}$
		x_0^8	7	9	0.27023	$3.08\text{x}10^{-07}$	3	5	0.11902	$3.80\text{x}10^{-07}$	24	26	0.52348	$9.40\text{x}10^{-07}$
4	1000	x_0^1	4	6	0.00751	$1.57\text{x}10^{-07}$	5	11	0.00890	$8.51\text{x}10^{-07}$	18	20	0.01677	$6.09\text{x}10^{-07}$
		x_0^2	3	5	0.00582	$1.88\text{x}10^{-07}$	10	26	0.01336	$3.96\text{x}10^{-07}$	23	25	0.01723	$6.12\text{x}10^{-07}$
		x_0^3	5	7	0.00778	$1.40\text{x}10^{-07}$	8	16	0.01060	$1.54\text{x}10^{-07}$	24	26	0.02026	$5.79\text{x}10^{-07}$
		x_0^4	5	7	0.00854	$8.47\text{x}10^{-07}$	14	32	0.01575	$9.25\text{x}10^{-07}$	24	26	0.02279	$8.15\text{x}10^{-07}$
		x_0^5	4	6	0.00695	$4.27\text{x}10^{-07}$	13	29	0.01548	$3.14\text{x}10^{-07}$	27	29	0.02235	$7.33\text{x}10^{-07}$
		x_0^6	4	6	0.00750	$1.18\text{x}10^{-07}$	6	12	0.00807	$1.16\text{x}10^{-07}$	23	25	0.01842	$7.76\text{x}10^{-07}$
		x_0^7	6	8	0.00896	$4.31\text{x}10^{-07}$	13	30	0.01455	$5.36\text{x}10^{-07}$	26	28	0.02169	$7.95\text{x}10^{-07}$
		x_0^8	4	6	0.00739	$3.74\text{x}10^{-07}$	10	26	0.01308	$6.07\text{x}10^{-07}$	21	23	0.02000	$8.35\text{x}10^{-07}$
	10,000	x_0^1	4	6	0.02468	$4.98\text{x}10^{-07}$	5	11	0.02726	$2.69\text{x}10^{-07}$	19	21	0.07851	$9.63\text{x}10^{-07}$
		x_0^2	3	5	0.02131	$5.96\text{x}10^{-07}$	11	27	0.06135	$9.78\text{x}10^{-07}$	24	26	0.08885	$9.67\text{x}10^{-07}$
		x_0^3	5	7	0.02761	$4.42\text{x}10^{-07}$	8	16	0.04121	$4.87\text{x}10^{-07}$	25	27	0.08508	$9.16\text{x}10^{-07}$
		x_0^4	6	8	0.03326	$2.68\text{x}10^{-07}$	14	32	0.06539	$2.93\text{x}10^{-07}$	26	28	0.09025	$6.44\text{x}10^{-07}$
		x_0^5	5	7	0.02637	$1.35\text{x}10^{-07}$	13	29	0.06477	$9.93\text{x}10^{-07}$	29	31	0.10635	$5.79\text{x}10^{-07}$
		x_0^6	4	6	0.02229	$3.74\text{x}10^{-07}$	6	12	0.03515	$3.67\text{x}10^{-07}$	25	27	0.07623	$6.13\text{x}10^{-07}$
		x_0^7	7	9	0.04239	$1.36\text{x}10^{-07}$	13	30	0.07940	$1.70\text{x}10^{-07}$	28	30	0.09822	$6.28\text{x}10^{-07}$
		x_0^8	5	7	0.02730	$1.18\text{x}10^{-07}$	10	26	0.05617	$1.92\text{x}10^{-07}$	23	25	0.10240	$6.60\text{x}10^{-07}$
	100,000	x_0^1	5	7	0.19552	$1.57\text{x}10^{-07}$	5	11	0.18691	$8.51\text{x}10^{-07}$	21	23	0.44488	$7.62\text{x}10^{-07}$
		x_0^2	4	6	0.15867	$1.88\text{x}10^{-07}$	11	27	0.39970	$3.09\text{x}10^{-07}$	26	28	0.57424	$7.65\text{x}10^{-07}$
		x_0^3	6	8	0.20061	$1.39\text{x}10^{-07}$	9	17	0.31194	$2.59\text{x}10^{-07}$	27	29	0.56037	$7.24\text{x}10^{-07}$
		x_0^4	6	8	0.19384	$8.46\text{x}10^{-07}$	14	32	0.48130	$9.25\text{x}10^{-07}$	28	30	0.60149	$5.09\text{x}10^{-07}$
		x_0^5	5	7	0.16203	$4.26\text{x}10^{-07}$	13	29	0.47761	$3.14\text{x}10^{-07}$	30	32	0.61837	$9.16\text{x}10^{-07}$
		x_0^6	5	7	0.19897	$1.18\text{x}10^{-07}$	7	13	0.25444	$5.03\text{x}10^{-07}$	26	28	0.57032	$9.70\text{x}10^{-07}$
		x_0^7	7	9	0.25124	$4.30\text{x}10^{-07}$	13	30	0.42969	$5.36\text{x}10^{-07}$	29	31	0.61943	$9.94\text{x}10^{-07}$
		x_0^8	5	7	0.18047	$3.73\text{x}10^{-07}$	10	26	0.37194	$6.07\text{x}10^{-07}$	25	27	0.52880	$5.22\text{x}10^{-07}$

TABLE IV: Numerical of UMLS, MDDYM & DTCG1 methods for problems 5 and 6

			UMLS				MDDYM				DTCG1			
Problem	Dimension	IP	Iter	Fval	Time(s)	$\|F_k\|$	Iter	Fval	Time(s)	$\|F_k\|$	Iter	Fval	Time(s)	$\|F_k\|$
5	1000	x_0^1	5	7	0.00836	$2.98\text{x}10^{-07}$	3	5	0.00616	$1.13\text{x}10^{-08}$	18	20	0.01372	$5.97\text{x}10^{-07}$
		x_0^2	6	8	0.00978	$5.64\text{x}10^{-07}$	5	9	0.00638	$1.50\text{x}10^{-08}$	22	24	0.01689	$7.30\text{x}10^{-07}$
		x_0^3	5	7	0.00925	$5.05\text{x}10^{-07}$	8	18	0.00929	$3.65\text{x}10^{-08}$	22	24	0.01649	$9.94\text{x}10^{-07}$
		x_0^4	6	8	0.00992	$7.36\text{x}10^{-07}$	21	62	0.02099	$5.63\text{x}10^{-08}$	23	25	0.01779	$5.54\text{x}10^{-07}$
		x_0^5	7	9	0.00951	$1.31\text{x}10^{-07}$	9	22	0.01093	$1.03\text{x}10^{-08}$	18	20	0.01682	$5.80\text{x}10^{-07}$
		x_0^6	6	8	0.00898	$4.78\text{x}10^{-07}$	7	15	0.00815	$1.46\text{x}10^{-08}$	22	24	0.01617	$8.31\text{x}10^{-07}$
		x_0^7	7	9	0.00961	$1.58\text{x}10^{-07}$	10	21	0.01099	$4.14\text{x}10^{-08}$	22	24	0.01651	$9.21\text{x}10^{-07}$
		x_0^8	6	8	0.00917	$1.63\text{x}10^{-07}$	2	4	0.00593	$2.99\text{x}10^{-08}$	21	23	0.01836	$6.82\text{x}10^{-07}$
	10,000	x_0^1	5	7	0.02735	$9.44\text{x}10^{-07}$	3	5	0.01613	$3.57\text{x}10^{-08}$	19	21	0.06026	$9.44\text{x}10^{-07}$
		x_0^2	7	9	0.03485	$1.78\text{x}10^{-07}$	5	9	0.02478	$4.76\text{x}10^{-08}$	24	26	0.06837	$5.77\text{x}10^{-07}$
		x_0^3	6	8	0.03216	$1.60\text{x}10^{-07}$	8	18	0.04011	$1.16\text{x}10^{-07}$	24	26	0.07163	$7.86\text{x}10^{-07}$
		x_0^4	7	9	0.03838	$2.32\text{x}10^{-07}$	22	65	0.09848	$9.08\text{x}10^{-07}$	24	26	0.08265	$8.76\text{x}10^{-07}$
		x_0^5	7	9	0.04040	$4.16\text{x}10^{-07}$	9	22	0.04017	$3.27\text{x}10^{-08}$	19	21	0.05703	$9.17\text{x}10^{-07}$
		x_0^6	7	9	0.03701	$1.51\text{x}10^{-07}$	7	15	0.03766	$4.62\text{x}10^{-08}$	24	26	0.07164	$6.57\text{x}10^{-07}$
		x_0^7	7	9	0.03967	$5.00\text{x}10^{-07}$	10	21	0.04598	$1.31\text{x}10^{-07}$	24	26	0.07879	$7.28\text{x}10^{-07}$
		x_0^8	6	8	0.03002	$5.16\text{x}10^{-07}$	2	4	0.01291	$9.45\text{x}10^{-08}$	23	25	0.07202	$5.39\text{x}10^{-07}$
	100,000	x_0^1	6	8	0.21126	$2.98\text{x}10^{-07}$	3	5	0.08105	$1.13\text{x}10^{-07}$	21	23	0.35696	$7.46\text{x}10^{-07}$
		x_0^2	7	9	0.23062	$5.63\text{x}10^{-07}$	5	9	0.12459	$1.50\text{x}10^{-07}$	25	27	0.42356	$9.13\text{x}10^{-07}$
		x_0^3	6	8	0.20330	$5.05\text{x}10^{-07}$	8	18	0.24486	$3.65\text{x}10^{-07}$	26	28	0.43346	$6.21\text{x}10^{-07}$
		x_0^4	7	9	0.24566	$7.35\text{x}10^{-07}$	23	67	0.69782	$2.67\text{x}10^{-08}$	26	28	0.42764	$6.93\text{x}10^{-07}$
		x_0^5	8	10	0.26866	$1.31\text{x}10^{-07}$	9	22	0.36473	$1.03\text{x}10^{-07}$	21	23	0.35811	$7.25\text{x}10^{-07}$
		x_0^6	7	9	0.23379	$4.78\text{x}10^{-07}$	7	15	0.20453	$1.46\text{x}10^{-07}$	26	28	0.54513	$5.20\text{x}10^{-07}$
		x_0^7	8	10	0.27548	$1.58\text{x}10^{-07}$	10	21	0.30392	$4.14\text{x}10^{-07}$	26	28	0.43737	$5.76\text{x}10^{-07}$
		x_0^8	7	9	0.22293	$1.63\text{x}10^{-07}$	2	4	0.06697	$2.99\text{x}10^{-07}$	24	26	0.41946	$8.52\text{x}10^{-07}$
6	1000	x_0^1	4	5	0.01323	0	46	338	0.11266	0	5	17	0.01536	0
		x_0^2	6	7	0.02358	0	–	–	–	–	4	27	0.01580	0
		x_0^3	3	4	0.01376	0	–	–	–	–	4	28	0.01618	0
		x_0^4	3	4	0.01430	0	152	980	0.21840	$9.03\text{x}10^{-07}$	4	30	0.01213	0
		x_0^5	2	3	0.01126	0	7	56	0.02595	0	2	15	0.01031	0
		x_0^6	5	6	0.01809	0	34	239	0.08493	0	4	27	0.01598	0
		x_0^7	4	5	0.01829	0	140	908	0.25018	$9.46\text{x}10^{-07}$	3	26	0.01242	0
		x_0^8	6	7	0.02376	0	33	245	0.11966	0	4	27	0.01382	0
	10,000	x_0^1	3	4	0.04679	0	10	73	0.14656	0	4	27	0.06608	0
		x_0^2	3	4	0.05887	0	8	57	0.13372	0	3	15	0.04217	0
		x_0^3	4	5	0.06621	0	9	65	0.12308	0	3	15	0.03669	0
		x_0^4	2	3	0.03632	0	11	76	0.15159	0	4	27	0.07693	0
		x_0^5	2	3	0.04617	0	4	35	0.07066	0	2	14	0.03100	0
		x_0^6	3	4	0.05115	0	8	62	0.13624	0	3	15	0.03967	0
		x_0^7	2	3	0.03956	0	8	66	0.16883	0	2	14	0.03868	0
		x_0^8	4	5	0.06691	0	14	94	0.21110	0	3	15	0.03861	0
	100,000	x_0^1	3	4	0.37248	0	11	76	1.24016	0	3	15	0.30455	0
		x_0^2	3	4	0.43502	0	5	39	0.63866	0	3	15	0.27900	0
		x_0^3	2	3	0.26978	0	4	29	0.47312	0	3	15	0.27563	0
		x_0^4	2	3	0.28419	0	10	80	1.26866	0	3	15	0.26904	0
		x_0^5	2	3	0.31900	0	3	24	0.39598	0	2	14	0.27089	0
		x_0^6	4	5	0.61536	0	4	29	0.47028	0	3	15	0.26858	0
		x_0^7	2	3	0.31410	0	6	48	0.77552	0	2	14	0.24513	0
		x_0^8	3	4	0.38979	0	9	73	1.16449	0	3	15	0.27406	0

REFERENCES

[1] W. Han, "A gradient descent method for solving a system of nonlinear equations," *Applied Mathematics Letters*, vol. 112, p. 106739, 2021. DOI: 10.1016/j.aml.2020.106739.

[2] J. Liu and S. J. Li, "A projection method for convex constrained monotone nonlinear equations with applications," *Computers and Mathematics with Applications*, vol. 70, pp. 2442-2453, 2015.

[3] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, vol. 16. Philadelphia, PA: SIAM, 1996.

[4] A. N. Iusem and M. V. Solodov, "Newton-type methods with generalized distances for constrained optimization," *Optimization*, vol. 41, pp. 257-278, 1997.

[5] A. S. Halilu, A. Majumder, M. Y. Waziri, and K. Ahmed, "Signal recovery with convex constrained nonlinear monotone equations through conjugate gradient hybrid approach," *Mathematics and Computers in Simulation*, 2021. DOI: 10.1016/j.matcom.2021.03.020.

[6] M. A. Aliyu, P. Kumama, and A. B. Abubakar, "A modified conjugate gradient method for monotone nonlinear equations with convex constraints," *Applied Numerical Mathematics*, vol. 145, pp. 507-520, 2019.

[7] B. Khan and P. Singh, "Optimal power flow techniques under characterization of conventional and renewable energy sources: A comprehensive analysis," *Journal of Engineering*, vol. 2017, pp. 1-16, 2017. DOI: 10.1155/2017/9539506.

[8] H. Mohammad and A. B. Abubakar, "A descent derivative-free algorithm for nonlinear monotone equations with convex constraints," *RAIRO Operations Research*, vol. 54, pp. 489–505, 2020.

[9] H. Abdullahi, A. K. Awasthi, M. Y. Waziri, and A. S. Halilu, "Descent three-term DY-type conjugate gradient methods for constrained monotone equations with application," *Computational and Applied Mathematics*, vol. 41, p. 32, 2022.

[10] H. Abdullahi, A. K. Awasthi, M. Y. Waziri, and A. S. Halilu, "A scaled three-term conjugate gradient method for convex-constrained monotone nonlinear equations with applications," *Journal of Physics: Conference Series*, vol. 2267, p. 012066, 2022. DOI: 10.1088/1742-6596/2267/1/012066.

[11] M. Y. Waziri, W. J. Leong, and M. A. Hassan, "Jacobian-Free Diagonal Newton's Method for Solving Nonlinear Systems with Singular Jacobian," *Malaysian Journal of Mathematical Science*, vol. 5, pp. 241–255, 2011.

[12] A. M. Awwal, P. Kumam, H. Mohammad, W. Watthayu, and A. B. Abubakar, "A Perry-type derivative-free algorithm for solving nonlinear system of equations and minimizing ℓ_1 regularized problem," *Optimization*, 2020. DOI: 10.1080/02331934.2020.1808647.

[13] M. Al-Baali, E. Spedicato, and F. Maggioni, "Broyden's quasi-Newton methods for a nonlinear system of equations and unconstrained optimization: a review and open problems," *Optimization Methods and Software*, vol. 29, no. 5, pp. 937–954, 2014.

[14] J. C. Manteuffel, "Steepest descent," *Lawrence Berkeley National Laboratory*, Berkeley, California, p. 94720, 2010. DOI: 10.2172/983240.

[15] K. Amini and F. Rostami, "A modified two-step Levenberg-Marquardt method for nonlinear equations," *Journal of Computational and Applied Mathematics*, vol. 288, pp. 341–350, 2015.

[16] Y. B. Musa, M. Y. Waziri, and A. S. Halilu, "On computing the regularization parameter for the Levenberg-Marquardt method via the spectral radius approach to solving systems of nonlinear equations," *Journal of Numerical Mathematics and Stochastics*, vol. 9, no. 1, pp. 80–94, 2017.

[17] Z. Papp and S. Rapajic, "FR type methods for systems of large-scale nonlinear monotone equations," *Applied Mathematics and Computation*, vol. 269, pp. 816–823, 2015.

[18] Y. Xiangfei, L. Zhijun, and D. Xiaoyu, "A global convergence of LS-CD hybrid conjugate gradient method," *Hindawi Publishing Corporation Advances in Numerical Analysis*, vol. 2013, Article ID 517452, 5 pages, 2013. DOI: 10.1155/2013/517452.

[19] P. Kaelo, P. Mtagulwa, and M. V. Thuto, "A globally convergent hybrid conjugate gradient method with strong Wolfe conditions for unconstrained optimization," *Mathematical Sciences*, 2019. DOI: 10.1007/s40096-019-00310-y.

[20] S. Djordjevic, "New hybrid conjugate gradient method as a convex combination of LS and CD methods," *Filomat*, vol. 31, no. 6, pp. 1813–1825, 2017. DOI: 10.2298/FIL1706813D.

[21] S. Aji, K. Poom, S. Punnarai, B. A. Auwal, and M. Y. Mahmoud, "A modified conjugate descent projection method for monotone nonlinear equations and image restoration," *IEEE Access*, 2020. DOI: 10.1109/ACCESS.2020.3020334.

[22] J. K. Liu, J. L. Xu, and L. Q. Zhang, "Partially symmetrical derivative-free Liu-Storey projection method for convex constrained equations," *International Journal of Computer Mathematics*, vol. 10, no. 1080/00207160, p. 1533122, 2018.

[23] Y. Liu and C. Storey, "Efficient generalized conjugate gradient algorithms. Part 1: theory," *Journal of Optimization Theory and Applications*, vol. 69, pp. 129–137, 1991.

[24] B. S. He, H. Yang, and S. L. Wang, "Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities," *Journal of Optimization Theory and Applications*, vol. 106, pp. 337–356, 2000.

[25] X. Y. Wang, X. J. Li, and X. P. Kou, "A self-adaptive three-term conjugate gradient method for monotone nonlinear equations with convex constraints," *Calcolo*, 2016. DOI: 10.1007/s10092-015-0140-5.

[26] D. H. Li and X. L. Wang, "A modified Fletcher-Reeves-type derivative-free method for symmetric nonlinear equations," *Numerical Algebra, Control and Optimization*, vol. 1, no. 1, pp. 71–82, 2011.

[27] L. Zhang, W. Zhou, and D. H. Li, "Global convergence of a modified Fletcher-Reeves conjugate gradient method with Armijo-type line search," *Numerische Mathematik*, vol. 104, pp. 561–572, 2006.

[28] J. Liu and S. Li, "Multivariate spectral projection method for convex constrained nonlinear monotone equations," *Journal of Industrial Management and Optimization*, vol. 13, no. 1, pp. 283–297, 2017.

[29] G. H. Yu, S. Z. Niu, and J. H. Ma, "Multivariate spectral gradient projection method for nonlinear monotone equations with convex constraints," *Journal of Industrial Management and Optimization*, vol. 9, pp. 117–129, 2013.

[30] W. Zhan, P. Li, X. Li, and H. Pham, "A modified three-term type CD conjugate gradient algorithm for unconstrained optimization problems," *Hindawi Mathematical Problems in Engineering*, vol. 2020, Article ID 4381515, 14 pages, 2020. DOI: 10.1155/2020/4381515.

[31] G. H. Yu and L. T. Guan, "New descent nonlinear conjugate gradient methods for large-scale unconstrained optimization," *Technical Report, Department of Scientific Computation and Computer Applications, Sun Yat-Sen University, Guangzhou, P.R. China*, 2005.

[32] G. Yu, L. Guan, and W. Chen, "Spectral conjugate gradient methods with sufficient descent property for large-scale unconstrained optimization," *Optimization Methods and Software*, vol. 23, pp. 275–293, 2008.

[33] A. Perry, "A modified conjugate gradient algorithm," *Operations Research Technical Notes*, vol. 26, no. 6, pp. 1073–1078, 1978.

[34] W. Sun and X. X. Yuan, *Optimization Theory and Methods: Nonlinear Programming*, Springer, New York, 2006.

[35] M. Y. Waziri, K. Ahmad, A. S. Halilu, and A. M. Awwal, "A modified Dai-Yuan iterative scheme for nonlinear systems and its application," *Numerical Algebra, Control and Optimization*, 2021. DOI: 10.3934/naco.2021044.

[36] E. D. Dolan and J. J. More, "Benchmarking optimization software with performance profiles," *Mathematical Programming*, Ser. A, vol. 91, pp. 201–213, 2002.

[37] M. R. Banham and A. K. Katsaggelos, "Digital image restoration," *IEEE Signal Processing Magazine*, vol. 14, no. 2, pp. 24–41, 1997.

[38] C. L. Chan, A. K. Katsaggelos, and A. V. Sahakian, "Image sequence filtering in quantum-limited noise with applications to low-dose fluoroscopy," *IEEE Transactions on Medical Imaging*, vol. 12, no. 3, pp. 610–621, 1993.

[39] M. Figueiredo, R. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 586–597, 2007.

[40] J. S. Pang, "Inexact Newton methods for the nonlinear complementarity problem," *Mathematical Programming*, vol. 1, pp. 54–71, 1986.

[41] Y. Xiao, Q. Wang, and Q. Hu, "Non-smooth equations-based method for ℓ_1-norm problems with applications to compressed sensing," *Nonlinear Analysis: Theory, Methods and Applications*, vol. 74, no. 11, pp. 3570–3577, 2011.

[42] R. Fletcher, *Practical Methods of Optimization: Volume 1: Unconstrained Optimization*, 2nd ed., Wiley, New York, 1997.

[43] Y. Liu and C. Storey, "Efficient generalized conjugate gradient algorithms, part 1: theory," *Journal of Optimization Theory and Applications*, vol. 69, pp. 129–137, 1991.