Iranian Journal of Numerical Analysis and Optimization
Vol. ??, No. ??, ??, pp ??
https://doi.org/??
https://ijnao.um.ac.ir/

Research Article

An efficient Dai-Kou-type method with image de-blurring application

K. Ahmed[1,*], M.Y. Waziri[1], S. Murtala[2], A.S. Halilu[3,4], H. Abdullahi[3] and Y.B. Musa[3]

Abstract

Well-conditioning of matrices has been shown to improve the numerical performance of algorithms by way of ensuring their numerical stability. In this paper, a modified Dai–Kou-type conjugate gradient method is developed for constrained nonlinear monotone systems by employing the well-conditioning approach. The new method ensures that the much required

*Corresponding author

Received ??? ; revised ??? ; accepted ???

Kabiru Ahmed, e-mail: kabiruhungu16@gmail.com
Mohammed Yususf Waziri, e-mail: mywaziri.mth@buk.edu.ng
Salisu Murtala, e-mail: salisumurtala@gmail.com
Abubakar Sani Halilu, e-mail: abubakars.halilu@slu.edu.ng
Habibu Abdullahi, e-mail: habibmth.slu@gmail.com
Ya'u Balarabe Musa, e-mail: yaumusa.jsu@gmail.com

[1]Department of Mathematical Sciences, Bayero University, Kano, Nigeria.
[2]Department of Mathematics, Federal University, Dutse, Nigeria.
[3]Department of Mathematics, Sule Lamido University, Kafin Hausa, Nigeria.
[4]Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Campus Besut, 22200 Terengganu, Malaysia

How to cite this article
Ahmed, K., Waziri, M.Y., Murtala, S., Halilu, A.S., Abdullahi, H. and Musa, Y.B., An efficient Dai-Kou-type method with image de-blurring application. *Iran. J. Numer. Anal. Optim.*, ??; ??(??): ??-??. ??

condition for global convergence of iterates generated is satisfied irrespective of the linesearch strategy employed. Another novelty of the scheme is its practical application in image de-blurring problems. The method performs well and converges globally under mild assumptions. Experiments in image de-blurring and convex constrained systems of equations show the scheme to be effective.

AMS subject classifications (2020): Primary 90C30; Secondary 90C26, 94A12.

Keywords: Nonlinear equations; Eigenvalues; Constrained equations; Convex set; Sparse signals.

1 Introduction

Generally, a system of nonlinear monotone equations is given by

$$F(x) = 0, \ x \in \mathbb{R}^n, \tag{1}$$

with F from $\mathbb{R}^n \to \mathbb{R}^n$, being a continuous and monotone mapping. Monotonicity of F means it satisfies the inequality

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

For the constrained version of (1), which is formulated as

$$F(\bar{x}) = 0; \quad \bar{x} \in \mathcal{C}, \tag{3}$$

$\bar{x}$ resides in a closed convex nonempty set $\mathcal{C} \subseteq \mathbb{R}^n$ for which (3) holds.

The Newton's and quasi-Newton's methods [14, 21, 48, 54] are the famous schemes employed for solving (1) and (3). However, storing the Jacobian or its approximation in every iteration, renders these methods unsuitable for high dimension problems.

The appropriate iterative scheme that conveniently addresses storage requirements is the conjugate gradient (CG) scheme. It is usually designed for the optimization problem

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

in which f denotes a smooth real-valued function. The CG method is often applied to solve (4) due to its minimal memory requirement. As with other line search methods, and starting with $x_0 \in \mathbb{R}^n$, the CG method's iterates are obtained via

$$x_{k+1} = x_k + s_k, \quad s_k = \vartheta_k d_k, \quad k \geq 0, \tag{5}$$

where x_k stands for previous iterate, $\vartheta_k > 0$ is the steplength that is usually obtained using a well-defined formula in the scheme's direction d_k, namely,

$$d_{k+1} = -g_{k+1} + \beta_k d_k, \quad d_0 = -g_0, \tag{6}$$

where $g_{k+1} = g(x_{k+1})$, $g_0 = g(x_0)$ represent gradients of f at x_{k+1} and x_k. In addition, β_k in (6) is a parameter that defines the CG scheme and its various formulation exists in the literature (see [31, 42]). The classical ones are proposed in [20, 25, 27, 33, 41, 50, 51] and are given by

$$\beta_k^{FR} = \frac{\|g_{k+1}\|^2}{\|g_k\|^2}, \quad \beta_k^{CD} = \frac{\|g_{k+1}\|^2}{-g_k^T d_k}, \quad \beta_k^{DY} = \frac{\|g_{k+1}\|^2}{d_k^T (g_{k+1} - g_k)}, \tag{7}$$

$$\beta_k^{HS} = \frac{g_{k+1}^T (g_{k+1} - g_k)}{d_k^T (g_{k+1} - g_k)}, \quad \beta_k^{PRP} = \frac{g_{k+1}^T (g_{k+1} - g_k)}{\|g_k\|^2}, \quad \beta_k^{LS} = \frac{g_{k+1}^T (g_{k+1} - g_k)}{-g_k^T d_k}, \tag{8}$$

with $\|\cdot\|$ being the $\ell_2 - norm$ of vectors.

A typical CG scheme implemented with (5) and (6), generates descent directions if the following inequality holds:

$$d_{k+1}^T g_{k+1} < 0. \tag{9}$$

However, for convergence analysis, the CG methods are required to satisfy the following sufficient descent condition:

$$d_{k+1}^T g_{k+1} \leq -c\|g_{k+1}\|^2, \quad c > 0. \tag{10}$$

By seeking a CG direction such that it will be closest to that of the scaled memoryless BFGS scheme [55], Dai and Kou [18] provided a class of CG schemes (DK) for solving (4) with the update parameter

$$\beta_k^{DK} = \frac{y_k^T g_{k+1}}{y_k^T d_k} - \left(\tau_k + \frac{\|y_k\|^2}{s_k^T y_k} - \frac{y_k^T s_k}{\|s_k\|^2}\right) \frac{g_{k+1}^T s_k}{y_k^T d_k}, \tag{11}$$

where $y_k = g_{k+1} - g_k$. The authors in [18] defined τ_k in (11) similar to the one given in [55]. Interestingly, other formulations have been provided over the years, which include the ones in [47] provided by Oren and Spedicato, namely,

$$\tau_k^{(1)} = \frac{s_k^T y_k}{y_k^T M_k y_k}, \quad \tau_k^{(2)} = \frac{\|y_k\|^2}{s_k^T y_k},$$

the ones proposed by Oren and Luenberger in [46], that is,

$$\tau_k^{(3)} = \frac{s_k^T M_k^{-1} s_k}{s_k^T y_k}, \quad \tau_k^{(4)} = \frac{s_k^T y_k}{s_k^T Q_k s_k},$$

as well as the choice provided in [6] by Al-Baali, namely,

$$\tau_k^{(5)} = \min\left\{1, \frac{\|y_k\|^2}{s_k^T y_k}\right\}, \quad \tau_k^{(6)} = \min\left\{1, \frac{s_k^T y_k}{\|s_k\|^2}\right\},$$

where M_k and Q_k are matrices. The approximation of τ_k given in [18], that is,

$$\tau_k = \frac{s_k^T y_k}{\|s_k\|^2},$$

has so far been taken to be the most effective for implementing the DK scheme. In their work in [18], the authors declared that other efficient approximations of τ_k can be obtained by employing different approaches.

Due to the appealing attributes of CG schemes for solving (4) with the knowledge that the optimality condition of (4) and (3) equates both concepts, that is, $\nabla f = F$, where F denotes the gradient of some objective functions, researchers have proposed their versions for solving (1) [34, 58, 59] and (3) [2, 3, 4, 32, 36, 40, 57, 63, 62]. Search directions of these schemes are defined as

$$d_0 = -F_0, \quad d_{k+1} = -F_{k+1} + \beta_k d_k, \quad F_{k+1} = F(x_{k+1}), \quad k = 0, 1, \ldots,$$

with β_k representing a modified version of any of the earlier CG parameters in (7) and (8) or their hybrid. To that end, researchers have combined the parameters in (7) and (8) with the projection technique in [54] to solve (1) and (3) (see [3, 34, 40, 58, 59, 62] for details). In response to the issue raised by the authors in [18] regarding other more effective approximations of the parameter τ_k in (11), some research aimed at addressing it have been made in recent years. For example, Ding et al. [22] provided a class of DK schemes for (3) with choices of τ_k given as

$$\tau_k^A = \frac{\|y_k\|^2}{s_k^T y_k}, \quad \tau_k^B = \frac{s_k^T y_k}{\|s_k\|^2}, \tag{12}$$

or the convex combination

$$\tau_k = \delta\tau_k^A + (1-\delta)\tau_k^B, \quad \delta \in [0,1], \tag{13}$$

in which

$$y_k = \tilde{y}_k - \lambda_k \sigma_k \|F_k\| d_k, \quad \sigma_k d_k = s_k, \quad \sigma_k > 0,$$

with

$$\lambda_k = 1 + \|F_k\|^{-1} \max\left\{0, \frac{-\sigma_k(\tilde{y}_k^T d_k)}{\|\sigma_k d_k\|^2}\right\}, \quad \tilde{y}_k = F_{k+1} - F_k, \quad F_k = F(x_k).$$

Following the work in [22] and by exploiting Newton's direction, Waziri et al. [56] presented another DK-type scheme for solving (3) with the choice of τ_k given as

$$\tau_k^{MDK} = 1 + \frac{s_k^T w_k}{\|s_k\|^2} - \frac{\|w_k\|^2}{s_k^T w_k}, \tag{14}$$

where

$$w_k = y_k + C_k s_k + D\|F_k\|^r s_k, \quad y_k = F(z_k) - F(x_k), \quad s_k = z_k - x_k = \sigma_k d_k,$$

$$C_k = \max\left\{-\frac{s_k^T y_k}{\|s_k\|}, 0\right\}, \quad D > 0, \quad r > 0.$$

In their recent work, Waziri et al. [2] proposed two other types of DK-type methods for (3) with approximations of τ_k defined as

$$\bar{\tau}_k = \max\left\{\tilde{\tau}_k, c_1\frac{\|\bar{y}_k\|^2}{\bar{s}_k^T \bar{y}_k}\right\}, \quad \hat{\tau}_k = \max\left\{\tilde{\tau}_k, c_2 + \frac{\|\bar{y}_k\|^2}{\bar{s}_k^T \bar{y}_k}\right\}, \tag{15}$$

in which

$$\tilde{\tau}_k = \frac{3\bar{s}_k^T \bar{y}_k}{\|\bar{s}_k\|^2} - \frac{\|\bar{y}_k\|^2}{\bar{s}_k^T \bar{y}_k}, \tag{16}$$

where

$\bar{y}_k = y_k + \psi_k \bar{s_k} + \Lambda \|F_k\|^r \bar{s_k}, \quad \bar{s_k} = w_k - x_k, \quad y_k = F(w_k) - F(x_k), \quad \Lambda > 0, \quad r > 0,$

with

$$\psi_k = \max\left\{\frac{-\bar{s}_k^T y_k}{\|\bar{s}_k\|^2}, 0\right\},$$

and $c_1 > 1$ and $c_2 > 0$.

Remark 1. It is worth stating here that only the schemes in [22] with choices of τ_k presented in (12) and (13) satisfy the condition (10) necessary for determining global convergence of algorithms for the problem (3) without any adjustments. For instance, the choice of τ_k given in (15) was obtained by adjusting the original choice in (16) since adopting the latter may not satisfy (10) automatically. Also, note that the choice in (14) may be negative or zero at some iterative point and may also not always satisfy (10). Lastly, the iteration matrices of the directions in [2, 22, 56] were not shown to be well-conditioned, which could improve the efficiency of the methods.

The article's objectives are listed as follows:

- To derive an efficient DK-type scheme for the constrained problem (3) with an approximation of τ_k obtained without any adjustments.

- To present a DK-type scheme for which the inequality (10) necessary in obtaining convergence results of methods for the problem (3) holds.

- To derive a method in which the symmetric form of its direction matrix is well-conditioned.

- To present proof of the scheme's convergence under mild conditions.

- To apply the scheme to image deblurring problems.

The remaining sections of the paper are outlined as follows: Section 2 deals with motivation and derivation of the proposed algorithm. Section 3 discusses the results of the convergence of the scheme. In Section 4, results of experiments carried out for problem (3) and image deblurring are discussed, while conclusions are made in Section 5.

2 Inspiration and Algorithm

We first recall that the most prominent quasi-Newton scheme developed by the researchers Broyden [15], Fletcher [26], Goldfarb [30], and Shanno [53] popularly known as BFGS, where B_k is usually an $n \times n$ symmetric positive-definite matrix is formulated as

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T y_k}, \quad s_k \in \mathbb{R}^n, \quad y_k \in \mathbb{R}^n. \tag{17}$$

From the Woodbury formula presented in [55] for the inverse of the sum of an invertible matrix and a rank-k correction, the inverse of (17) is given as

$$H_{k+1} = H_k - \frac{s_k y_k^T H_k + H_k y_k s_k^T}{s_k^T y_k} + \left(1 + \frac{y_k H_k y_k}{s_k^T y_k}\right) \frac{s_k s_k^T}{s_k^T y_k}, \quad s_k \in \mathbb{R}^n, \quad y_k \in \mathbb{R}^n.$$

To avoid computing and storing the $n \times n$ matrix H_k at each iteration, it is replaced by the identity matrix I, and the so called memoryless update is obtained, that is,

$$H_{k+1} = I - \frac{s_k y_k^T + y_k s_k^T}{s_k^T y_k} + \left(1 + \frac{\|y_k\|^2}{s_k^T y_k}\right) \frac{s_k s_k^T}{s_k^T y_k}, \quad s_k \in \mathbb{R}^n, \quad y_k \in \mathbb{R}^n. \tag{18}$$

As mentioned earlier, the BFGS method implemented with (17) is the most popular and effective quasi-Newton scheme available. The method is guaranteed to satisfy the descent condition (9), since the update (17) satisfies the much required quasi-Newton condition. Other attributes of the BFGS scheme include its correction of eigenvalues mechanism [43]. However, the BFGS's efficiency depends strongly on the structure of eigenvalues of (17) [8]. Powell [52] and Byrd et al. [16] noted that the update (17) better corrects its small eigenvalues than large ones. Also, numerical experiments conducted by Gill and Leonard [29] showed that it is possible for the update (17) to require many iterations or gradient and function evaluations for some problems. The authors in [29] showed that these shortcomings of the BFGS method may result from poor initial Hessian approximations or its ill-conditioning along the iterations. To overcome these shortfalls of the scheme, a number of scaling techniques have been applied to the BFGS update matrix in (17). This includes the modification by Biggs [13], where the update's third term in (17) was scaled by a positive parameter γ_k to yield

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \gamma_k \frac{y_k y_k^T}{y_k^T s_k}, \quad s_k \in \mathbb{R}^n, \quad y_k \in \mathbb{R}^n.$$

In Oren and Luenberger [45], the first and second terms of the matrix in (17) were scaled and the resulting modification becomes

$$B_{k+1} = \delta_k \left[B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} \right] + \frac{y_k y_k^T}{y_k^T s_k}, \quad s_k \in \mathbb{R}^n, \quad y_k \in \mathbb{R}^n,$$

where $\delta_k > 0$. Motivated by the strategy of changing structure of eigenvalues [43], Andrei [10] provided a two-parameter scaling BFGS method, where B_{k+1} is given by

$$B_{k+1} = \delta_k \left[B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} \right] + \gamma_k \frac{y_k y_k^T}{y_k^T s_k}, \quad s_k \in \mathbb{R}^n, \quad y_k \in \mathbb{R}^n,$$

with $\gamma_k > 0$ and $\delta_k > 0$. In this update, δ_k is obtained such that eigenvalues of B_{k+1} are clustered, while γ_k is computed to have a shift of the eigenvalues to the left. The latter procedure produces a better distribution of the eigenvalues. In other developments, the update matrix defined by (18) has also been modified in order to better distribute the eigenvalues and improve performance of the scheme. To that end, the following self scaled memoryless approximation to the Hessian inverse (18) was presented in [44]

$$H_{k+1} = \theta_k I - \theta_k \frac{s_k y_k^T + y_k s_k^T}{s_k^T y_k} + \left(1 + \theta_k \frac{\|y_k\|^2}{s_k^T y_k} \right) \frac{s_k s_k^T}{s_k^T y_k}, \quad s_k \in \mathbb{R}^n, \quad y_k \in \mathbb{R}^n, \tag{19}$$

with θ_k known as scaling parameter. In line with (19), Babaie-Kafaki [12] proposed the following extension:

$$H_{k+1} = \theta_k I - \theta_k \frac{s_k y_k^T + y_k s_k^T}{s_k^T y_k} + \left(1 + \gamma_k \frac{\|y_k\|^2}{s_k^T y_k} \right) \frac{s_k s_k^T}{s_k^T y_k}, \quad s_k \in \mathbb{R}^n, \quad y_k \in \mathbb{R}^n, \tag{20}$$

where γ_k and θ_k represents positive parameters. Analysis of the scheme obtained with (20) proves that it satisfies (10) and its condition number remains in an improved condition. A modification of (18) was proposed in [11], namely,

$$H_{k+1} = \frac{1}{\delta_k}\left[H_k - \frac{H_k y_k s_k^T + s_k y_k^T H_k}{s_k^T y_k} + \left(\frac{\delta_k}{\gamma_k} + \frac{y_k^T H_k y_k}{s_k^T y_k}\right)\frac{s_k s_k^T}{s_k^T y_k}\right],$$

where δ_k and γ_k are parameters determined by employing Byrd and Nocedal's measure function in [17]. Now, as stated by Andrei [8], to achieve faster convergence of linear CG methods, the following approaches are employed:

- Clustering eigenvalues of a search direction matrix about a point [9, 60] or about several points [37] in its spectrum.

- Preconditioning of a search direction matrix [35].

Before we proceed to formulate our scheme, we first give the following additional assumptions on the mapping F:

Assumption 1. The solution set $\bar{\mathcal{C}}$ of (3) is not empty, that is, there exists $\bar{x} \in \mathcal{C}$ satisfying (3).

Assumption 2. F is Lipschitz continuous, that is,

$$\|F(x) - F(y)\| \leq L\|x - y\|, \quad \text{for all } x, y \in \mathcal{C}, \quad L \quad \text{a positive constant.} \tag{21}$$

Now, motivated by the shortcomings of the DK-type methods in [2, 22, 56], the scaled double parameter BFGS approximation to the inverse Hessian (20) as well as the need to explore other more effective approximations of the DK parameter, that ensures (10) holds without any adjustment, we propose the following DK-type search direction:

$$d_{k+1} = -\gamma F_{k+1} + \gamma\beta_k^{NHS} d_k - \left(\tau_k + \gamma\frac{\|\bar{y}_k\|^2}{s_k^T\bar{y}_k} - \gamma\frac{s_k^T\bar{y}_k}{\|s_k\|^2}\right)\frac{F_{k+1}^T s_k}{d_k^T\bar{y}_k} d_k, \quad d_0 = -F_0, \tag{22}$$

where

$$\beta_k^{NHS} = \frac{F_{k+1}^T\bar{y}_k}{d_k^T\bar{y}_k}, \quad k = 0, 1, \ldots, \tag{23}$$

with

$$\bar{y}_k = y_k + rs_k, \quad y_k = F(w_k) - F(x_k), \quad r > 0, \tag{24}$$

and

$$w_k = x_k + \vartheta_k d_k, \quad s_k = w_k - x_k.$$

From (24) and (2), we have

$$d_k^T \bar{y}_k = \frac{s_k^T y_k}{\vartheta_k} + \frac{r}{\vartheta_k}\|s_k\|^2 \geq \frac{r}{\vartheta_k}\|s_k\|^2 > 0,$$

from which we obtain

$$s_k^T \bar{y}_k = s_k^T y_k + r\|s_k\|^2 \geq r\|s_k\|^2 > 0. \tag{25}$$

Note that the search direction defined by (22) can be written in compact form as

$$d_{k+1} = -M_{k+1}F_{k+1},$$

where

$$M_{k+1} = \gamma I - \gamma\frac{s_k\bar{y}_k^T}{s_k^T\bar{y}_k} + \tau_k\frac{s_k s_k^T}{s_k^T\bar{y}_k} + \gamma\frac{\|\bar{y}_k\|^2 s_k s_k^T}{(s_k^T\bar{y}_k)^2} - \gamma\frac{s_k s_k^T}{\|s_k\|^2}. \tag{26}$$

To proceed, we add rank-one update to (26) to obtain its symmetric form as

$$\bar{M}_{k+1} = \gamma I - \gamma\frac{s_k\bar{y}_k^T}{s_k^T\bar{y}_k} - \gamma\frac{\bar{y}_k s_k^T}{s_k^T\bar{y}_k} + \tau_k\frac{s_k s_k^T}{s_k^T\bar{y}_k} + \gamma\frac{\|\bar{y}_k\|^2 s_k s_k^T}{(s_k^T\bar{y}_k)^2} - \gamma\frac{s_k s_k^T}{\|s_k\|^2}. \tag{27}$$

Better still, we can re-write (27) as

$$\bar{M}_{k+1} = \gamma I Q_{k+1}, \tag{28}$$

in which

$$Q_{k+1} = I - \frac{s_k\bar{y}_k^T}{s_k^T\bar{y}_k} - \frac{\bar{y}_k s_k^T}{s_k^T\bar{y}_k} + \tau_k\frac{s_k s_k^T}{\gamma s_k^T\bar{y}_k} + \frac{\|\bar{y}_k\|^2 s_k s_k^T}{(s_k^T\bar{y}_k)^2} - \frac{s_k s_k^T}{\|s_k\|^2}. \tag{29}$$

We can further express (29) as the rank-two update

$$\begin{aligned} Q_{k+1} =& I - \frac{s_k\bar{y}_k^T}{s_k^T\bar{y}_k} \\ &+ \frac{(\tau_k\|s_k\|^2(s_k^T\bar{y}_k)s_k - \gamma\|s_k\|^2(s_k^T\bar{y}_k)\bar{y}_k + \gamma\|s_k\|^2\|\bar{y}_k\|^2 s_k - \gamma(s_k^T\bar{y}_k)^2 s_k)s_k^T}{\gamma\|s_k\|^2(s_k^T\bar{y}_k)^2} \end{aligned} \tag{30}$$

Now, since from (25) $s_k^T\bar{y}_k > 0$, then $s_k \neq 0$ and $\bar{y}_k \neq 0$. Suppose $\mathcal{V} = span\{s_k, \bar{y}_k\}$. Then $\dim(\mathcal{V}) \leq 2$ and $\dim(\mathcal{V}^\perp) \geq n-2$, with $\mathcal{V}^\perp$ being orthogonal complement of $\mathcal{V}$. So, there exists a set of mutually orthogonal vectors $\{\xi_k^i\}_{i=1}^{n-2} \subset \mathcal{V}^\perp$ such that

$$s_k^T \xi_k^i = \bar{y}_k^T \xi_k^i = 0, \quad i = 1, \dots, n-2,$$

for which we obtain

$$\bar{M}_{k+1}\xi_k^i = \bar{M}_{k+1}^T \xi_k^i = \gamma \xi_k^i, \quad i = 1, \dots, n-2.$$

Therefore, $\bar{M}_{k+1}$ contains $n-2$ eigenvalues equal to γ each. We now find the remaining two eigenvalues, which we label as, λ_k^+ and λ_k^-.

By applying the fundamental formula of algebra (see [55, inequality (1.2.70)]) for determinant of a rank-two update, namely,

$$\det(I+v_1v_2^T+v_3v_4^T) = (1+v_1^Tv_2)(1+v_3^Tv_4)-(v_1^Tv_4)(v_2^Tv_3), \quad v_1, v_2, v_3, v_4 \in \mathbb{R}^n,$$

and setting $v_1 = -\frac{s_k}{s_k^T \bar{y}_k}$, $v_2 = \bar{y}_k$,
$v_3 = \frac{(\|s_k\|^2(s_k^T\bar{y}_k)^2\tau_k s_k - \gamma\|s_k\|^2(s_k^T\bar{y}_k)\bar{y}_k + \gamma\|s_k\|^2\|\bar{y}_k\|^2 s_k - \gamma(s_k^T\bar{y}_k)^2 s_k)}{\gamma\|s_k\|^2(s_k^T\bar{y}_k)^2}$, and $v_4 = s_k$, we get

$$\det(Q_{k+1}) = \tau_k \frac{\|s_k\|^2}{\gamma s_k^T \bar{y}_k} - 1. \tag{31}$$

Note that the matrix $\bar{M}_{k+1}$ as defined in (28) is the product of two matrices, and

$$\det(\gamma I) = \gamma^n.$$

Combining this result with (31), we obtain

$$\begin{aligned}\det(\bar{M}_{k+1}) &= \gamma^n \left(\tau_k \frac{\|s_k\|^2}{\gamma s_k^T \bar{y}_k} - 1\right)\\ &= \gamma^{n-2}.\lambda^+\lambda^-,\end{aligned}$$

which yields

$$\lambda^+\lambda^- = \gamma^2 \left(\tau_k \frac{\|s_k\|^2}{\gamma s_k^T \bar{y}_k} - 1\right) = \gamma\tau_k \frac{\|s_k\|^2}{s_k^T \bar{y}_k} - \gamma^2.$$

Since trace of the symmetric matrix $\bar{M}_{k+1}$ is the summation of all its eigenvalues, we have

$$\begin{aligned}\operatorname{tr}(\bar{M}_{k+1}) &= n\gamma - 2\gamma + \tau_k \frac{\|s_k\|^2}{s_k^T \bar{y}_k} + \gamma \frac{\|\bar{y}_k\|^2 \|s_k\|^2}{(s_k^T \bar{y}_k)^2)} - \gamma \\ &= \underbrace{\gamma + \cdots + \gamma}_{(n\text{-}2)\text{ times}} + \lambda_k^+ + \lambda_k^-,\end{aligned}$$

which further yields

$$\lambda_k^+ + \lambda_k^- = \tau_k \frac{\|s_k\|^2}{s_k^T \bar{y}_k} + \gamma \frac{\|\bar{y}_k\|^2 \|s_k\|^2}{(s_k^T \bar{y}_k)^2} - \gamma. \tag{32}$$

From (32) and (31), the remaining eigenvalues of $\bar{M}_{k+1}$ are obtained as solution of the following quadratic polynomial:

$$\lambda^2 - \left(\tau_k \frac{\|s_k\|^2}{s_k^T \bar{y}_k} + \gamma \frac{\|\bar{y}_k\|^2 \|s_k\|^2}{(s_k^T \bar{y}_k)^2} - \gamma\right)\lambda + \gamma\tau_k \frac{\|s_k\|^2}{s_k^T \bar{y}_k} - \gamma^2.$$

Consequently, by setting $\Phi_k = \frac{\|s_k\|^2}{s_k^T \bar{y}_k}$, $\mu_k = \frac{\|s_k\|^2 \|\bar{y}_k\|^2}{(s_k^T \bar{y}_k)^2}$, λ_k^+ and λ_k^- are determined by

$$\lambda_k^{\pm} = \frac{\tau_k \Phi_k + \gamma\mu_k - \gamma \pm \sqrt{(\tau_k \Phi_k + \gamma\mu_k - \gamma)^2 - 4(\gamma\tau_k \Phi_k - \gamma^2)}}{2},$$

or more precisely,

$$\lambda_k^{\pm} = \frac{\tau_k \Phi_k + \gamma\mu_k - \gamma \pm \sqrt{(\tau_k \Phi_k + \gamma\mu_k - 3\gamma)^2 + 4\gamma^2\mu_k - 4\gamma^2}}{2}. \tag{33}$$

Clearly, by the Cauchy–Schwarz inequality in (33), $\lambda_k^+ > 0$. Also, $\lambda_k^- > 0$ whenever

$$\tau_k > \frac{\gamma}{\Phi_k} = \frac{\gamma s_k^T \bar{y}_k}{\|s_k\|^2}. \tag{34}$$

Now, we proceed to obtain an approximation of τ_k such that (34) is satisfied making $\bar{M}_{k+1}$ a positive-definite matrix. To achieve this, we employ the clustering of eigenvalues technique. Suppose that λ_k^+ and λ_k^- have the same values as the first $(n-2)$ eigenvalues of $\bar{M}_{k+1}$, namely, $\lambda_k^+ = \lambda_k^- = \gamma$. Then from determinant of $\bar{M}_{k+1}$ obtained in (31), we have

$$\tau_k \frac{\|s_k\|^2}{\gamma s_k^T \bar{y}_k} - 1 = 1,$$

which implies that

$$\tau_k = 2\frac{\gamma s_k^T \bar{y}_k}{\|s_k\|^2}, \tag{35}$$

which clearly satisfies (34) and ensures that all the eigenvalues of $\bar{M}_{k+1}$ are clustered.

Lemma 1. The search direction sequence $\{d_k\}$ obtained by (22) with (23), (24) and $\gamma \in (0, 1]$ satisfy the inequality

$$d_{k+1}^T F_{k+1} \leq -c\|F_{k+1}\|^2, \tag{36}$$

where $c = \frac{3\gamma}{4}$.

Proof. From (22), (35), and by setting $\Gamma_k = s_k^T \bar{y}_k$ for convenience, we have

$$\begin{aligned}
d_{k+1}^T F_{k+1} &= -\gamma\|F_{k+1}\|^2 + \gamma \frac{F_{k+1}^T \bar{y}_k}{\Gamma_k} F_{k+1}^T s_k \\
&\quad - \left(\tau_k + \gamma \frac{\|\bar{y}_k\|^2}{\Gamma_k} - \gamma \frac{\Gamma_k}{\|s_k\|^2}\right) \frac{(F_{k+1}^T s_k)^2}{\Gamma_k} \\
&= -\gamma\|F_{k+1}\|^2 + \gamma \frac{F_{k+1}^T \bar{y}_k}{\Gamma_k} F_{k+1}^T s_k - \left(\gamma \frac{\Gamma_k}{\|s_k\|^2} + \gamma \frac{\|\bar{y}_k\|^2}{\Gamma_k}\right) \frac{(F_{k+1}^T s_k)^2}{\Gamma_k} \\
&\leq -\gamma\|F_{k+1}\|^2 + \gamma \frac{F_{k+1}^T \bar{y}_k}{\Gamma_k} F_{k+1}^T s_k - \gamma \frac{\|\bar{y}_k\|^2}{\Gamma_k^2} (F_{k+1}^T s_k)^2 \\
&= \frac{\gamma F_{k+1}^T \bar{y}_k \Gamma_k F_{k+1}^T s_k - \gamma \Gamma_k^2 \|F_{k+1}\|^2 - \gamma \|\bar{y}_k\|^2 (F_{k+1}^T s_k)^2}{\Gamma_k^2} \\
&\leq \frac{\gamma \frac{\Gamma_k^2 \|F_{k+1}\|^2}{4} + \gamma \|\bar{y}_k\|^2 (F_{k+1}^T s_k)^2 - \gamma \Gamma_k^2 \|F_{k+1}\|^2 - \gamma \|\bar{y}_k\|^2 (F_{k+1}^T s_k)^2}{\Gamma_k^2} \\
&= \gamma \frac{\|F_{k+1}\|^2}{4} - \gamma\|F_{k+1}\|^2 \\
&= -\gamma\left(1 - \frac{1}{4}\right)\|F_{k+1}\|^2 \\
&= -\frac{3\gamma}{4}\|F_{k+1}\|^2.
\end{aligned}$$

We arrived at the last inequality by employing the identity

$$2c_1^T c_2 \leq \|c_1\|^2 + \|c_2\|^2, \quad c_1, c_2 \in \mathbb{R}^n,$$

with $c_1 = \frac{\Gamma_k F_{k+1}}{\sqrt{2}}$, $c_2 = \sqrt{2}(F_{k+1}^T s_k)\bar{y}_k$. Hence, setting $c = \frac{3\gamma}{4}$, we see that (36) holds. □

Next, we introduce the projection operator defined by

$$\mathcal{P}_{\mathcal{C}}(x) = \arg\min \|x - y\| : y \in \mathcal{C}, \quad \text{for all } x \in \mathbb{R}^n,$$

with the properties:

$$\|\mathcal{P}_{\mathcal{C}}(x) - \mathcal{P}_{\mathcal{C}}(y)\| \leq \|x - y\|, \quad \text{for all } x, y \in \mathbb{R}^n,$$

and

$$\|\mathcal{P}_{\mathcal{C}}(x) - y\| \leq \|x - y\|, \quad \text{for all } y \in \mathcal{C}, \tag{37}$$

where $\mathcal{C}$ is as defined earlier.

Algorithm 1

Data: Select $\epsilon > 0$, $x_0 \in \mathcal{C}$, $\beta \in (0, 1)$, $\delta \in (0, 1)$, $0 < \phi < 2$, $r > 0$, $\gamma \in (0, 1]$.

Initialization: Set $k = 0$ and $d_0 = -F_0$.

1: Obtain $F(x_k)$ and confirm if $\|F(x_k)\| \leq \epsilon$. End if yes, otherwise goto 2.

2: Determine $w_k = x_k + \vartheta_k d_k$, where $\vartheta_k = \beta^{m_k}$, with m being the smallest nonnegative integer for which

$$-F(x_k + \beta^m d_k)^T d_k \geq \delta \beta^m \|d_k\|^2 \tag{38}$$

holds.

3: If $w_k \in \mathcal{C}$ and $\|F(w_k)\| \leq \epsilon$, end, otherwise, compute

$$x_{k+1} = \mathcal{P}_{\mathcal{C}}\left[x_k - \phi \rho_k F(w_k)\right], \quad \text{where} \tag{39}$$

$$\rho_k = \frac{F(w_k)^T (x_k - w_k)}{\|F(w_k)\|^2}. \tag{40}$$

4: Obtain d_{k+1} by (22) with (23), (24), and (35).

5: Set $k = k + 1$ and proceed to 1.

3 Convergence report

First, we show that τ_k obtained in (35) is bounded.

From (21), (25), (35) and the Cauchy Schwarz inequality, we have

$$
\begin{aligned}
|\tau_k| &\le \frac{2\|s_k\|\|\bar{y}_k\|}{\|s_k\|^2} \\
&\le \frac{2L\|s_k\|^2}{\|s_k\|^2} \\
&= 2L \stackrel{\text{def}}{=} \bar{m}.
\end{aligned} \tag{41}
$$

Lemma 2. The sequence $\{d_{k+1}\}$ of directions obtained by Algorithm 1 satisfy

$$
c\|F_{k+1}\| \le \|d_{k+1}\| \le \left(\gamma + \frac{2\gamma L}{r} + \frac{\bar{m}}{r} + \frac{\gamma L^2}{r^2}\right)\|F_{k+1}\|, \tag{42}
$$

where $\gamma \in (0,1]$, $r > 0$, and $L > 0$.

Proof. The first inequality follows from the Cauchy–Schwarz inequality and (22). For $k = 0$ in (22), we have that $d_0 = -F_0$, which indicates that $\|d_0\| = \|F_0\|$. Now, we show that the inequality holds for $k \ge 1$. From the Cauchy–Schwarz inequality, (21), (22), (25), and (41), we obtain

$$
\begin{aligned}
\|d_{k+1}\| &= \left\| -\gamma F_{k+1} + \gamma \frac{F_{k+1}^T \bar{y}_k}{s_k^T \bar{y}_k} s_k - \left(\tau_k + \gamma \frac{\|\bar{y}_k\|^2}{s_k^T \bar{y}_k} - \gamma \frac{s_k^T \bar{y}_k}{\|s_k\|^2}\right) \frac{F_{k+1}^T s_k}{s_k^T \bar{y}_k} s_k \right\| \\
&\le \gamma\|F_{k+1}\| + \gamma \frac{\|F_{k+1}\|\|\bar{y}_k\|\|s_k\|}{s_k^T \bar{y}_k} + |\tau_k| \frac{\|F_{k+1}\|\|s_k\|^2}{s_k^T \bar{y}_k} \\
&\quad + \gamma \frac{\|F_{k+1}\|\|\bar{y}_k\|^2\|s_k\|^2}{(s_k^T \bar{y}_k)^2} + \gamma \frac{\|F_{k+1}\|\|s_k\|^3\|\bar{y}_k\|}{\|s_k\|^2 s_k^T \bar{y}_k} \\
&\le \gamma\|F_{k+1}\| + \gamma \frac{L\|F_{k+1}\|\|s_k\|^2}{r\|s_k\|^2} + \bar{m} \frac{\|F_{k+1}\|\|s_k\|^2}{r\|s_k\|^2} + \gamma \frac{L^2\|F_{k+1}\|\|s_k\|^4}{r^2\|s_k\|^4} \\
&\quad + \gamma \frac{L\|F_{k+1}\|\|s_k\|^4}{r\|s_k\|^4} \qquad (43) \\
&= \gamma\|F_{k+1}\| + \gamma \frac{L\|F_{k+1}\|}{r} + \bar{m} \frac{\|F_{k+1}\|}{r} + \gamma \frac{L^2\|F_{k+1}\|}{r^2} + \gamma \frac{L\|F_{k+1}\|}{r} \\
&= \gamma\|F_{k+1}\| + 2\gamma \frac{L\|F_{k+1}\|}{r} + \bar{m} \frac{\|F_{k+1}\|}{r} + \gamma \frac{L^2\|F_{k+1}\|}{r^2} \\
&= \left(\gamma + \frac{2\gamma L}{r} + \frac{\bar{m}}{r} + \frac{\gamma L^2}{r^2}\right)\|F_{k+1}\|,
\end{aligned}
$$

which proves the second inequality of (42). □

Next, we prove that the line search (38) is well defined and also terminates after finite iterations:

Lemma 3. Let Assumption 2 hold, and suppose that Algorithm 1 is not terminated in step 1. Then there exists a nonnegative integer m_k such that (38) is satisfied. In addition, the step-size ϑ_k obtained in (38) satisfies

$$\vartheta_k \geq \vartheta := \min \left\{ 1, \frac{3\gamma\beta}{4(L+\delta)\left(\gamma + \frac{2\gamma L}{r} + \frac{\bar{m}}{r} + \frac{\gamma L^2}{r^2}\right)^2} \right\}. \tag{44}$$

Proof. To show the first part, we assume that there exists $k_0 \geq 0$ such that (38) is not true in the k_0^{th} iterate for each value of m. So, for all $m \geq 0$, we have

$$-F(x_{k_0} + \beta^m d_{k_0})^T d_{k_0} < \delta \beta^m \|d_{k_0}\|^2. \tag{45}$$

Since F is continuous on $\mathbb{R}^n$, applying limit to (45) as m grows to infinity, yields

$$F(x_{k_0})^T d_{k_0} > 0,$$

which is contradicted by (36), namely,

$$F(x_{k_0})^T d_{k_0} \leq -\frac{3\gamma}{4} \|F(x_{k_0})\|^2.$$

Thus, we proved the first part.

Now, suppose that the algorithm is terminated at x_k, then $F(x_k) = 0$ or $F(w_k) = 0$. This indicates the solution to be x_k, otherwise x_k is not a solution. Then, from (36) $d_k \neq 0$. Now, from (38) we see that if $\vartheta_k \neq 1$, then $\bar{\vartheta}_k = \beta^{-1}\vartheta_k$ will not satisfy (38), that is,

$$-F(\bar{w}_k)^T d_k < \delta \bar{\vartheta}_k \|d_k\|^2,$$

where, $\bar{w}_k = x_k + \bar{\vartheta}_k d_k$. By Assumption 2 and (36), we have

$$\begin{aligned}
\frac{3\gamma}{4}\|F_k\|^2 &\leq -F_k^T d_k \\
&= (F(\bar{w}_k) - F_k)^T d_k - F(\bar{w}_k)^T d_k \\
&\leq L\bar{\vartheta}_k \|d_k\|^2 + \delta \bar{\vartheta}_k \|d_k\|^2 \\
&= \beta^{-1}\vartheta_k (L+\delta)\|d_k\|^2.
\end{aligned}$$

Hence, we obtain

$$\begin{aligned}
\vartheta_k &\geq \frac{3\gamma\beta}{4(L+\delta)}\frac{\|F_k\|^2}{\|d_k\|^2} \\
&\geq \frac{3\gamma\beta}{4(L+\delta)}\frac{\|F_k\|^2}{\left(\gamma+\frac{2\gamma L}{r}+\frac{\bar{m}}{r}+\frac{\gamma L^2}{r^2}\right)^2\|F_k\|^2} \\
&= \frac{3\gamma\beta}{4(L+\delta)\left(\gamma+\frac{2\gamma L}{r}+\frac{\bar{m}}{r}+\frac{\gamma L^2}{r^2}\right)^2},
\end{aligned}$$

where (43) was used to obtain the second inequality. □

Lemma 4. Let Assumptions 1, and 2 hold. Then for a solution $\bar{x}$ of (3) in $\bar{\mathcal{C}}$, the sequence $\{\|x_k-\bar{x}\|\}$ is convergent implying that $\{x_k\}$ is bounded. Also

$$\lim_{k\to\infty}\vartheta_k\|d_k\|=0. \tag{46}$$

Proof. From (38) and definition of w_k, we have

$$(x_k-w_k)^TF(w_k)\geq\delta\vartheta_k^2\|d_k\|^2. \tag{47}$$

By (2) and for all $\bar{x}\in\bar{\mathcal{C}}$, we have

$$\begin{aligned}
(x_k-\bar{x})^TF(w_k) &= (x_k-w_k)^TF(w_k)+(w_k-\bar{x})^TF(w_k) \\
&\geq (x_k-w_k)^TF(w_k)+(w_k-\bar{x})^TF(\bar{x}) \\
&= (x_k-w_k)^TF(w_k).
\end{aligned} \tag{48}$$

From (37), (39), (40), (47) and (48), we have

$$\begin{aligned}
\|x_{k+1}-\bar{x}\|^2 &= \|\mathcal{P}_{\mathcal{C}}[x_k-\phi\rho_kF(w_k)]-\bar{x}\|^2 \\
&\leq \|x_k-\phi\rho_kF(w_k)-\bar{x}\|^2 \\
&= \|(x_k-\bar{x})-\phi\rho_kF(w_k)\|^2 \\
&= \|x_k-\bar{x}\|^2-2\phi\rho_kF(w_k)^T(x_k-\bar{x})+\phi^2\rho_k^2\|F(w_k)\|^2 \\
&\leq \|x_k-\bar{x}\|^2-2\phi\rho_kF(w_k)^T(x_k-w_k)+\phi^2\rho_k^2\|F(w_k)\|^2 \\
&= \|x_k-\bar{x}\|^2-\phi(2-\phi)\frac{(F(w_k)^T(x_k-w_k))^2}{\|F(w_k)\|^2} \\
&\leq \|x_k-\bar{x}\|^2-\phi(2-\phi)\frac{\delta^2\|x_k-w_k\|^4}{\|F(w_k)\|^2},
\end{aligned} \tag{49}$$

which yields

$$0 \leq \|x_{k+1} - \bar{x}\| \leq \|x_k - \bar{x}\| \leq \|x_{k-1} - \bar{x}\| \leq \cdots \leq \|x_0 - \bar{x}\|.$$

So, $\{\|x_k - \bar{x}\|\}$ is non-increasing and bounded, which indicates that $\{x_k\}$ is bounded also. This with the fact that F is Lipschitz continuous implies that a constant m_1 exists for all $k \geq 0$ such that,

$$\|x_k\| \leq m_1, \quad \|F(x_k)\| \leq m_1. \tag{50}$$

Also, by (43) and (50) a constant m_2 exists for which

$$\|d_k\| \leq \left(\gamma + \frac{2\gamma L}{r} + \frac{\bar{m}}{r} + \frac{\gamma L^2}{r^2}\right) m_1.$$

Setting $m_2 = \left(\gamma + \frac{2\gamma L}{r} + \frac{\bar{m}}{r} + \frac{\gamma L^2}{r^2}\right) m_1$, we obtain that d_k is bounded. Furthermore, from (50), monotonicity of F, the Cauchy–Schwarz inequality, and (47), we have

$$m_1 \geq \|F_k\| \geq \frac{F_k^T(x_k - w_k)}{\|x_k - w_k\|} \geq \frac{F(w_k)^T(x_k - w_k)}{\|x_k - w_k\|} \geq \delta\|x_k - w_k\| \geq \delta\|w_k\| - \delta m_1,$$

which consequently implies that

$$\|w_k\| \leq \frac{m_1 + \delta m_1}{\delta}.$$

By setting $m_3 := \frac{m_1 + \delta m_1}{\delta}$, we establish boundedness of $\{w_k\}$. Hence, from continuity of F, a constant $\bar{m}$ exists such that

$$\|F(w_k)\| \leq \bar{m}, \quad \text{for all } k \geq 0.$$

Combining this with (49), we obtain

$$\delta^2\|x_k - w_k\|^4 \leq \frac{\bar{m}^2}{\phi(2-\phi)}(\|x_k - \bar{x}\|^2 - \|x_{k+1} - \bar{x}\|^2). \tag{51}$$

Now, following the convergence of $\{\|x_k - \bar{x}\|\}$ and boundedness of $\{F(w_k)\}$, we take limit as k approaches infinity in (51) to obtain

$$\delta^2 \lim_{k \to \infty} \vartheta_k^4 \|d_k\|^4 \leq 0,$$

which indicates that

$$\lim_{k \to \infty} \vartheta_k \|d_k\| = 0.$$

□

Theorem 1. Suppose that Assumptions 1 and 2 hold and that $\{x_k\}$ is obtained by Algorithm 2.1. Then, $\{x_k\}$ converges to a solution of (3).

Proof. Firstly, from (44) and (46), we have that $0 \leq \vartheta\|d_k\| \leq \vartheta_k\|d_k\| \to 0$, which consequently indicates that $\lim_{k\to\infty}\|d_k\| = 0$. This together with (42) yields

$$0 \leq \frac{3}{4\gamma}\|F_k\| \leq \|d_k\| \to 0,$$

which indicates that $\lim_{k\to\infty}\|F_k\| = 0$. Now, inequality (46) and the boundedness of the sequence $\{x_k\}$ indicates the existence of a cluster point of $\{x_k\}$ say $\tilde{x} \subset \bar{\mathcal{C}}$, where $\bar{\mathcal{C}}$ denotes solution set of F. Let $\mathcal{K} \subseteq \{0, 1, 2, \ldots\}$ be an infinite index set for which

$$\lim_{k\to\infty, k\in\mathcal{K}} x_k = \tilde{x} \in \bar{\mathcal{C}}.$$

Since F is continuous, we have that

$$0 = \lim_{k\to\infty}\|F_k\| = \lim_{k\to\infty, k\in\mathcal{K}}\|F_k\| = \|F(\tilde{x})\|,$$

which indicates that $\tilde{x}$ is a solution of (3). Also, since $\{\|x_k - \bar{x}\|\}$ is convergent, setting $\bar{x} = \tilde{x}$ yields

$$\lim_{k\to\infty}\|x_k - \bar{x}\| = \lim_{k\to\infty, k\in\mathcal{K}}\|x_k - \bar{x}\| = 0.$$

which, therefore, indicates that $\{x_k\}$ converges to $\bar{x} \in \bar{\mathcal{C}}$. □

4 Results of numerical experiments

To test effectiveness of Algorithm 1, two experiments are conducted and discussed in the next two subsections.

4.1 First experiment: Convex constrained nonlinear monotone systems

For these experiments, the performance of Algorithm 1 is tested against four recent methods for solving the constrained problem (3), namely, ACGD [22], MDKM [56], SCRME [28], and SDYCG [7]. Codes for the algorithms, which are available at https://github.com/hungugida/hungugida/blob/main/MATLABcodeforconstrainedsystem.zip was written in MATLAB $R2014a$ and executed using a system configured as (2.30ghz cpu, 4gb RAM). The stoppage criteria for all runs are $\|F(x_k)\| \leq 10^{-10}$ or $\|F(w_k)\| \leq 10^{-10}$ or iterations exceed 1000. We set parameters of (38) for Algorithm 1 as $\beta = 0.6$, $\delta = 0.0001$, $\gamma = 0.27$, $\phi = 1.8$, $r = 0.0001$. The exact values of the parameters used in the articles for each of the four schemes were also applied here.

The underlisted test examples with dimensions 5000, 10000, and 50000 were used to test Algorithm 1, ACGD, MDKM, SCRME and SDYCG, where F is given as: $F = (f_1(x), f_2(x), \ldots, f_n(x))^T$.

Example 1. [38] with $\mathcal{C} = \mathbb{R}^n_+$ added to yield
$f_i(x) = 2x_i - \sin x_i, \quad i = 1, 2, \ldots, n.$

Example 2. [40].
$f_1(x) = x_1 - \exp\left(\cos\left(\frac{x_1+x_2}{n+1}\right)\right),$
$f_i(x) = x_i - \exp\left(\cos\left(\frac{x_{i-1}+x_i+x_{i+1}}{n+1}\right)\right), \quad i = 2, 3, \ldots, n-1,$
$f_n(x) = x_n - \exp\left(\cos\left(\frac{x_{n-1}+x_n}{n+1}\right)\right),$
with $\mathcal{C} = \mathbb{R}^n_+$.

Example 3. [38]
$f_i(x) = 2x_i - \sin|x_i|, \quad i = 1, 2, \ldots, n,$
where $\mathcal{C} = \mathbb{R}^n_+$.

Example 4. This is a modified version of the example in [39] with $\mathcal{C} = \mathbb{R}^n_+$ added to yield
$f_1(x) = e^{\sin x_1} - 1,$
$f_i(x) = e^{\sin x_i} + x_i - 1, \quad i = 2, \ldots, n.$

Example 5. [64] with $\mathcal{C} = \mathbb{R}^n_+$ added to yield
$f_1(x) = 2x_1 + \sin x_1 - 1,$
$f_i(x) = 2x_{i-1} + 2x_i + 2\sin x_i - 1,$
$f_n(x) = 2x_n + \sin x_n - 1, \quad i = 2, \ldots, n-1.$

Example 6. This is a modification of test example 4
$f_1(x) = 3x_1 + e^{\sin x_1} - 1,$
$f_i(x) = 3x_i + e^{\sin x_i} - 1, \quad i = 2, \ldots, n,$
with $\mathcal{C} = \mathbb{R}^n_+$.

Example 7. This is a modification of test example 5
$f_1(x) = 3x_1 + \cos x_1 - 1,$
$f_i(x) = 3x_{i-1} + 3x_i + \cos x_i - 1,$
$f_n(x) = 3x_n + \cos x_n - 1, \quad i = 2, \ldots, n-1,$
with $\mathcal{C} = \mathbb{R}^n_+$.

Example 8. Modification of test example 2
$f_1(x) = x_1 - e^{\left(\cos \frac{x_1 + x_2}{2}\right)},$
$f_i(x) = x_i - e^{\left(\cos \frac{x_{i-1} + x_i + x_{i+1}}{i}\right)}, \quad i = 2, 3, \ldots, n-1,$
$f_n(x) = x_n - e^{\left(\cos \frac{x_{n-1} + x_n}{n}\right)}.$
where $\mathcal{C} = \mathbb{R}^n_+$.

The following initial guesses were used:
$x_0^1 = \left(1, \frac{1}{2}, \ldots, \frac{1}{n}\right)^T$, $x_0^2 = \left(\frac{1}{2}, \frac{3}{2}, \ldots, -\frac{[(-1)^n - 2]}{2}\right)^T$, $x_0^3 = \left(1, 3, \ldots, -\frac{-2[(-1)^n - 2]}{2}\right)^T$,
$x_0^4 = \left(\frac{n-1}{n}, \frac{n-2}{n}, \ldots, 0\right)^T$, $x_0^5 = \left(\frac{1}{4}, \frac{3}{4}, \ldots, \frac{-[(-1)^n - 2]}{4}\right)^T$, $x_0^6 = \left(\frac{1}{n}, \frac{2}{n}, \ldots, 1\right)^T$.

Table 1: Test results for Examples 1-2.

PN	VAR	SP	Algorithm 1				ACGD				MDKM				SRCME				SDYCG			
			NIT	FE	PT	Norm	NIT	FE	PT	Norm	NIT	FE	PT	Norm	NIT	FE	PT	Norm	NIT	FE	PT	Norm
1	5000	x_0^1	1	3	0.0054	0	19	41	0.5232	7.42E-11	14	17	0.3490	9.47E-11	38	41	0.1396	9.66E-11	281	1277	1.4496	9.53E-11
	5000	x_0^2	1	3	0.0070	0	19	41	0.0632	7.42E-11	15	18	0.0519	1.01E-11	38	41	0.1088	9.65E-11	292	1175	1.2170	1.79E-11
	5000	x_0^3	1	4	0.0082	0	20	44	0.0755	7.45E-11	13	18	0.0464	3.37E-11	40	43	0.0970	7.81E-11	347	1593	1.6216	1.92E-11
	5000	x_0^4	1	3	0.0092	0	19	41	0.0647	7.42E-11	15	18	0.0444	1.01E-11	38	41	0.0926	9.65E-11	331	1275	1.3538	6.81E-12
	5000	x_0^5	1	3	0.0088	0	19	41	0.0668	6.50E-11	12	14	0.0485	1.58E-11	38	41	0.0905	8.95E-11	319	683	0.9334	9.97E-11
	5000	x_0^6	1	3	0.0067	0	19	43	0.0717	1.00E-10	13	20	0.0450	1.03E-11	42	44	0.0992	8.05E-11	242	511	0.6694	3.62E-11
	10000	x_0^1	1	3	0.0374	0	20	43	0.0929	2.73E-11	15	18	0.0703	1.37E-11	39	42	0.1401	6.84E-11	351	1528	2.3655	1.81E-11
	10000	x_0^2	1	3	0.0089	0	20	43	0.0998	2.73E-11	15	18	0.0614	1.42E-11	39	42	0.1427	6.84E-11	285	1187	1.9231	6.45E-11
	10000	x_0^3	1	4	0.0101	0	21	46	0.1042	2.75E-11	13	18	0.0598	4.77E-11	41	44	0.1461	5.54E-11	288	1113	1.8137	5.72E-11
	10000	x_0^4	1	3	0.0084	0	20	43	0.0973	2.73E-11	15	18	0.0794	1.42E-11	39	42	0.1578	6.84E-11	289	1269	1.9909	3.27E-11
	10000	x_0^5	1	3	0.0093	0	19	41	0.1470	9.20E-11	12	14	0.0577	2.23E-11	39	42	0.1445	6.33E-11	283	604	1.3013	1.46E-11
	10000	x_0^6	1	3	0.0068	0	20	45	0.1043	3.68E-11	13	20	0.0687	1.46E-11	42	45	0.1496	5.71E-11	240	507	1.0708	6.79E-11
	50000	x_0^1	1	3	0.3420	0	20	43	0.3388	6.12E-11	15	18	0.2442	3.12E-11	40	43	0.5318	7.66E-11	296	1369	9.3452	9.80E-11
	50000	x_0^2	1	3	0.0260	0	20	43	0.3510	6.12E-11	15	18	0.2356	3.14E-11	40	43	0.5466	7.66E-11	325	1634	10.8721	2.33E-11
	50000	x_0^3	1	4	0.0329	0	21	46	0.3673	6.16E-11	14	19	0.2720	1.07E-11	42	45	0.5699	6.22E-11	398	1565	11.1525	9.91E-11
	50000	x_0^4	1	3	0.0241	0	20	43	0.3459	6.12E-11	15	18	0.2455	3.14E-11	40	43	0.5385	7.66E-11	411	1712	12.1336	9.77E-11
	50000	x_0^5	1	3	0.0261	0	20	43	0.3859	5.35E-11	12	14	0.2001	4.99E-11	40	43	0.5689	7.08E-11	257	552	5.1273	6.16E-11
	50000	x_0^6	1	3	0.0233	0	20	45	0.3542	8.23E-11	13	20	0.2499	3.27E-11	44	46	0.6126	6.43E-11	246	519	4.8538	9.24E-11
2	5000	x_0^1	11	13	0.1418	1.14E-11	20	43	0.0904	8.19E-11	12	14	0.0497	1.58E-11	40	43	0.1520	8.63E-11	95	193	0.4535	9.42E-11
	5000	x_0^2	11	14	0.0487	4.79E-11	20	43	0.1007	8.19E-11	12	14	0.0491	1.58E-11	40	43	0.1289	8.63E-11	95	193	0.4406	9.42E-11
	5000	x_0^3	11	14	0.0439	7.30E-11	20	43	0.0921	4.59E-11	11	13	0.0448	8.85E-11	39	42	0.1374	9.66E-11	91	185	0.4374	8.79E-11
	5000	x_0^4	12	14	0.0417	2.69E-11	20	43	0.0920	8.19E-11	12	14	0.0515	1.58E-11	40	43	0.1390	8.63E-11	95	193	0.4177	9.42E-11
	5000	x_0^5	11	14	0.0474	4.91E-11	20	43	0.0891	8.18E-11	12	14	0.0531	1.58E-11	40	43	0.1374	8.61E-11	99	201	0.4301	8.55E-11
	5000	x_0^6	12	14	0.0486	2.69E-11	20	43	0.0959	4.51E-11	11	13	0.0607	8.71E-11	39	42	0.1393	9.50E-11	104	211	0.4846	8.90E-11
	10000	x_0^1	10	13	0.0954	9.21E-11	21	45	0.1570	3.01E-11	12	14	0.0859	2.24E-11	41	44	0.2455	6.10E-11	103	209	0.7820	8.88E-11
	10000	x_0^2	12	14	0.1105	7.48E-11	21	45	0.1631	3.01E-11	12	14	0.0780	2.24E-11	41	44	0.2176	6.10E-11	103	209	0.7706	8.88E-11
	10000	x_0^3	12	14	0.0731	1.03E-11	20	43	0.1622	6.49E-11	12	14	0.0953	1.25E-11	40	43	0.2300	6.83E-11	93	189	0.7331	9.06E-11
	10000	x_0^4	11	14	0.0672	6.70E-11	21	45	0.1729	3.01E-11	12	14	0.0799	2.24E-11	41	44	0.2297	6.10E-11	103	209	0.8000	8.88E-11
	10000	x_0^5	11	13	0.0703	3.31E-11	21	45	0.1597	3.01E-11	12	14	0.0797	2.23E-11	41	44	0.2452	6.09E-11	109	221	0.8264	9.79E-11
	10000	x_0^6	11	14	0.0856	6.70E-11	20	43	0.1617	6.38E-11	12	14	0.0761	1.23E-11	40	43	0.2246	6.72E-11	123	249	0.9785	7.83E-11
	50000	x_0^1	9	12	0.3199	8.35E-11	21	45	0.6183	6.74E-11	12	14	0.3359	5.00E-11	42	45	0.9522	6.82E-11	72	147	2.6030	8.18E-11
	50000	x_0^2	11	13	0.2748	8.54E-11	21	45	0.6229	6.74E-11	12	14	0.3279	5.00E-11	42	45	0.9546	6.82E-11	72	147	2.6052	8.18E-11
	50000	x_0^3	10	13	0.2791	6.32E-11	21	45	0.6132	3.77E-11	12	14	0.3226	2.80E-11	41	44	0.9294	7.64E-11	68	139	2.4719	7.67E-11
	50000	x_0^4	10	13	0.2708	6.50E-11	21	45	0.6050	6.74E-11	12	14	0.3376	5.00E-11	42	45	0.9661	6.82E-11	72	147	2.5842	8.18E-11
	50000	x_0^5	10	12	0.2800	4.03E-11	21	45	0.6152	6.73E-11	12	14	0.3191	4.99E-11	42	45	0.9721	6.81E-11	74	151	2.6862	9.09E-11
	50000	x_0^6	10	13	0.2787	6.50E-11	21	45	0.6079	3.71E-11	12	14	0.3260	2.75E-11	41	44	0.9699	7.51E-11	81	165	2.9192	7.70E-11

Table 2: Test results for Examples 3–4.

PN	VAR	SP	Algorithm 1				ACGD				MDKM				SRCME				SDYCG			
			NIT	FE	PT	Norm	NIT	FE	PT	Norm	NIT	FE	PT	Norm	NIT	FE	PT	Norm	NIT	FE	PT	Norm
3	5000	x_0^1	1	3	0.0506	0	19	41	0.0575	7.42E-11	12	15	0.0406	1.16E-11	38	41	0.0979	9.66E-11	172	1027	0.9486	5.42E-11
	5000	x_0^2	1	3	0.0073	0	19	41	0.0630	7.42E-11	12	15	0.0359	1.16E-11	38	41	0.0998	9.65E-11	153	1041	0.9608	5.83E-11
	5000	x_0^3	1	4	0.0082	0	20	44	0.0650	7.45E-11	13	18	0.0387	3.42E-11	40	43	0.1023	7.81E-11	137	1009	0.8833	4.83E-11
	5000	x_0^4	1	3	0.0054	0	19	41	0.0718	7.42E-11	12	15	0.0463	1.16E-11	38	41	0.1005	9.65E-11	123	1017	0.8978	2.90E-11
	5000	x_0^5	1	3	0.0070	0	19	41	0.0641	6.50E-11	12	14	0.0433	1.60E-11	38	41	0.0914	8.95E-11	106	241	0.3715	8.02E-11
	5000	x_0^6	1	3	0.0068	0	19	43	0.0657	1.00E-10	13	20	0.0450	1.04E-11	43	46	0.1046	5.69E-11	33	153	0.1958	7.05E-11
	10000	x_0^1	1	3	0.0092	0	20	43	0.0971	2.73E-11	12	15	0.0568	1.65E-11	39	42	0.1443	6.84E-11	179	1026	1.5294	9.73E-11
	10000	x_0^2	1	3	0.0085	0	20	43	0.0946	2.73E-11	12	15	0.0553	1.65E-11	39	42	0.1432	6.84E-11	166	928	1.4229	6.49E-11
	10000	x_0^3	1	4	0.0108	0	21	46	0.1059	2.75E-11	13	18	0.0625	4.84E-11	41	44	0.1475	5.54E-11	118	918	1.2932	1.66E-11
	10000	x_0^4	1	3	0.0093	0	20	43	0.0979	2.73E-11	12	15	0.0640	1.65E-11	39	42	0.1534	6.84E-11	178	989	1.4766	8.44E-11
	10000	x_0^5	1	3	0.0091	0	19	41	0.0880	9.20E-11	12	14	0.0640	2.26E-11	39	42	0.1836	6.33E-11	110	249	0.5705	6.12E-11
	10000	x_0^6	1	3	0.0078	0	20	45	0.0987	3.68E-11	13	20	0.0682	1.46E-11	43	46	0.1671	8.04E-11	33	153	0.2766	9.37E-11
	50000	x_0^1	1	3	0.0249	0	20	43	0.3522	6.12E-11	12	15	0.2029	3.68E-11	40	43	0.5142	7.66E-11	216	1296	8.0604	6.36E-11
	50000	x_0^2	1	3	0.0259	0	20	43	0.3571	6.12E-11	12	15	0.2056	3.68E-11	40	43	0.5524	7.66E-11	* * *	* * *	* * *	* * *
	50000	x_0^3	1	4	0.0326	0	21	46	0.3795	6.16E-11	14	19	0.2440	1.08E-11	42	45	0.5538	6.22E-11	149	1005	6.3431	4.96E-11
	50000	x_0^4	1	3	0.0259	0	20	43	0.3509	6.12E-11	12	15	0.2145	3.68E-11	40	43	0.5434	7.66E-11	210	1545	9.647	7.74E-11
	50000	x_0^5	1	3	0.0280	0	20	43	0.3539	5.35E-11	12	14	0.2240	5.05E-11	40	43	0.5291	7.08E-11	143	315	2.9196	8.83E-11
	50000	x_0^6	1	3	0.0257	0	20	45	0.3622	8.23E-11	13	20	0.2395	3.27E-11	44	47	0.6443	9.05E-11	37	162	1.2419	2.33E-11
4	5000	x_0^1	1	4	0.0944	0	17	69	0.0637	3.94E-11	12	50	0.0492	2.69E-11	48	51	0.1253	4.73E-11	1	13	0.0193	0
	5000	x_0^2	1	4	0.0081	0	17	69	0.0765	3.94E-11	12	50	0.0497	2.69E-11	48	51	0.1203	4.73E-11	1	13	0.0191	0
	5000	x_0^3	5	10	0.0322	0	18	73	0.0884	3.23E-11	12	55	0.0602	9.59E-11	50	53	0.1223	9.30E-11	1	13	0.0206	0
	5000	x_0^4	1	4	0.0071	0	17	69	0.0724	3.94E-11	12	50	0.0599	2.69E-11	48	51	0.1152	4.73E-11	1	13	0.0219	0
	5000	x_0^5	1	4	0.0082	0	16	69	0.0906	3.85E-11	10	46	0.0465	7.37E-11	48	50	0.1100	8.64E-11	1	13	0.0208	0
	5000	x_0^6	1	4	0.0079	0	19	78	0.0952	4.11E-11	13	56	0.0639	5.52E-11	53	57	0.1416	8.86E-11	3	26	0.0328	0
	10000	x_0^1	1	4	0.0099	0	17	69	0.1228	5.59E-11	12	50	0.0975	3.81E-11	48	51	0.1850	6.69E-11	1	13	0.0296	0
	10000	x_0^2	1	4	0.0115	0	17	69	0.1115	5.59E-11	12	50	0.0840	3.81E-11	48	51	0.1988	6.69E-11	1	13	0.0303	0
	10000	x_0^3	5	10	0.0277	0	18	73	0.1336	4.59E-11	14	59	0.1005	1.06E-11	52	54	0.2025	9.30E-11	1	13	0.0283	0
	10000	x_0^4	1	4	0.0092	0	17	69	0.1294	5.59E-11	12	50	0.0830	3.81E-11	48	51	0.1887	6.69E-11	1	13	0.0302	0
	10000	x_0^5	1	4	0.0084	0	16	69	0.1142	5.44E-11	11	50	0.1068	8.17E-12	48	51	0.1862	5.76E-11	1	13	0.0258	0
	10000	x_0^6	1	4	0.0106	0	19	78	0.1383	5.85E-11	13	56	0.0997	7.79E-11	55	58	0.2113	8.86E-11	3	26	0.0686	0
	50000	x_0^1	1	4	0.0347	0	18	73	0.4537	2.39E-11	12	50	0.3198	8.51E-11	50	53	0.7322	4.99E-11	1	13	0.1211	0
	50000	x_0^2	1	4	0.0322	0	18	73	0.4312	2.39E-11	12	50	0.3259	8.51E-11	50	53	0.7229	4.99E-11	1	13	0.1313	0
	50000	x_0^3	5	10	0.1218	0	19	77	0.5047	1.96E-11	14	59	0.3879	2.38E-11	52	55	0.7367	9.80E-11	1	13	0.1366	0
	50000	x_0^4	1	4	0.0425	0	18	73	0.4601	2.39E-11	12	50	0.3517	8.51E-11	50	53	0.7321	4.99E-11	1	13	0.1464	0
	50000	x_0^5	1	4	0.0337	0	17	73	0.4407	2.31E-11	11	50	0.3181	1.83E-11	50	52	0.7817	9.11E-11	1	13	0.1235	0
	50000	x_0^6	1	4	0.0366	0	20	82	0.5146	2.50E-11	14	60	0.4002	1.36E-11	55	59	0.8389	9.34E-11	3	26	0.2487	0

Table 3: Test results for Examples 5–6.

PN	VAR	SP	Algorithm 1				ACGD				MDKM				SRCME				SDYCG			
			NIT	FE	PT	Norm	NIT	FE	PT	Norm	NIT	FE	PT	Norm	NIT	FE	PT	Norm	NIT	FE	PT	Norm
5	5000	x_0^1	39	125	0.1944	7.91E-11	***	***	***	***	47	433	0.3396	9.53E-11	58	62	0.2015	7.56E-11	***	***	***	***
	5000	x_0^2	46	167	0.1822	8.03E-11	***	***	***	***	47	433	0.3043	9.93E-11	58	62	0.1707	6.55E-11	***	***	***	***
	5000	x_0^3	35	118	0.1484	3.66E-11	167	1532	0.9541	8.43E-11	49	450	0.3283	8.24E-11	62	66	0.1717	6.26E-11	***	***	***	***
	5000	x_0^4	43	127	0.1625	9.12E-11	62	558	0.4088	5.30E-11	47	433	0.3098	9.93E-11	58	62	0.1645	6.55E-11	***	***	***	***
	5000	x_0^5	38	108	0.1451	4.16E-11	***	***	***	***	49	451	0.3166	6.70E-11	67	72	0.1840	8.07E-11	***	***	***	***
	5000	x_0^6	38	129	0.1564	6.50E-11	***	***	***	***	50	458	0.3394	7.72E-11	68	74	0.2063	8.85E-11	***	***	***	***
	10000	x_0^1	32	111	0.2280	8.73E-11	***	***	***	***	48	442	0.5151	6.19E-11	58	62	0.2597	9.37E-11	***	***	***	***
	10000	x_0^2	46	167	0.2847	9.05E-11	***	***	***	***	48	442	0.5069	6.48E-11	58	62	0.2468	8.44E-11	***	***	***	***
	10000	x_0^3	38	117	0.2288	6.22E-11	***	***	***	***	49	450	0.5597	8.85E-11	62	66	0.2643	8.92E-11	***	***	***	***
	10000	x_0^4	43	130	0.2462	8.99E-11	134	1221	1.3007	9.29E-11	48	442	0.4975	6.48E-11	58	62	0.2412	8.44E-11	***	***	***	***
	10000	x_0^5	34	109	0.2048	8.33E-11	***	***	***	***	49	451	0.5514	8.23E-11	68	73	0.2792	8.30E-11	***	***	***	***
	10000	x_0^6	41	134	0.2719	5.37E-11	***	***	***	***	50	458	0.5229	8.52E-11	70	76	0.2798	7.16E-11	***	***	***	***
	50000	x_0^1	36	128	0.9632	6.36E-11	***	***	***	***	48	442	2.2058	9.31E-11	60	64	0.9733	7.14E-11	***	***	***	***
	50000	x_0^2	47	171	1.2254	8.08E-11	***	***	***	***	48	442	2.1443	9.50E-11	59	63	0.9761	1.00E-10	***	***	***	***
	50000	x_0^3	38	124	0.9703	7.24E-11	***	***	***	***	50	459	2.2698	6.54E-11	64	68	1.0506	6.86E-11	***	***	***	***
	50000	x_0^4	46	144	1.0945	9.30E-11	***	***	***	***	48	442	2.1490	9.50E-11	59	63	0.9941	1.00E-10	***	***	***	***
	50000	x_0^5	43	155	1.1195	5.40E-11	***	***	***	***	50	460	2.2638	8.07E-11	71	76	1.1856	6.75E-11	***	***	***	***
	50000	x_0^6	49	149	1.1485	3.40E-11	***	***	***	***	51	467	2.2987	7.15E-11	73	79	1.2177	8.84E-11	***	***	***	***
6	5000	x_0^1	1	5	0.0461	0	13	79	0.0566	1.00E-10	11	79	0.0680	1.96E-11	56	60	0.1672	9.23E-11	1	13	0.0167	0
	5000	x_0^2	1	5	0.0078	0	13	79	0.0761	1.00E-10	11	79	0.0717	1.96E-11	56	60	0.1466	9.23E-11	1	13	0.0225	0
	5000	x_0^3	7	19	0.0257	0	13	85	0.0836	6.70E-11	11	86	0.0811	2.32E-11	61	65	0.1602	6.35E-11	1	13	0.0224	0
	5000	x_0^4	1	5	0.0082	0	13	79	0.0868	1.00E-10	11	79	0.0763	1.96E-11	56	60	0.1507	9.23E-11	1	13	0.0210	0
	5000	x_0^5	1	5	0.0086	0	12	79	0.0759	9.24E-11	10	79	0.0723	7.83E-12	56	60	0.1641	9.70E-11	1	13	0.0211	0
	5000	x_0^6	1	5	0.0093	0	15	96	0.0854	1.14E-11	12	92	0.0850	6.23E-12	62	67	0.1759	6.60E-11	1	13	0.0282	0
	10000	x_0^1	1	5	0.0088	0	14	85	0.1316	1.60E-11	11	79	0.1254	2.77E-11	58	62	0.2513	5.22E-11	1	13	0.0357	0
	10000	x_0^2	1	5	0.0151	0	14	85	0.1262	1.60E-11	11	79	0.1250	2.77E-11	58	62	0.2652	5.22E-11	1	13	0.0326	0
	10000	x_0^3	7	19	0.0485	0	13	85	0.1399	9.53E-11	11	86	0.1047	3.28E-11	61	65	0.2242	8.98E-11	1	13	0.0328	0
	10000	x_0^4	1	5	0.0125	0	14	85	0.1159	1.60E-11	11	79	0.1281	2.77E-11	58	62	0.2317	5.22E-11	1	13	0.0348	0
	10000	x_0^5	1	5	0.0116	0	13	85	0.1214	1.47E-11	10	79	0.1353	1.11E-11	58	62	0.2324	5.49E-11	1	13	0.0348	0
	10000	x_0^6	1	5	0.0090	0	15	96	0.1562	1.64E-11	12	92	0.1333	8.81E-12	62	67	0.2273	9.33E-11	1	13	0.0346	0
	50000	x_0^1	1	5	0.0366	0	14	85	0.4488	3.62E-11	11	79	0.3852	6.20E-11	60	64	0.8419	4.67E-11	1	13	0.1447	0
	50000	x_0^2	1	5	0.0425	0	14	85	0.4320	3.62E-11	11	79	0.3877	6.20E-11	60	64	0.8647	4.67E-11	1	13	0.1333	0
	50000	x_0^3	7	19	0.1931	0	15	91	0.4840	2.42E-11	11	86	0.4735	7.33E-11	63	67	0.9133	8.03E-11	1	13	0.1443	0
	50000	x_0^4	1	5	0.0406	0	14	85	0.4272	3.62E-11	11	79	0.4529	6.20E-11	60	64	0.8653	4.67E-11	1	13	0.1435	0
	50000	x_0^5	1	5	0.0422	0	13	85	0.4573	3.29E-11	10	79	0.4343	2.48E-11	60	64	0.8796	4.92E-11	1	13	0.1320	0
	50000	x_0^6	1	5	0.0384	0	15	96	0.5068	3.72E-11	12	92	0.5133	1.97E-11	64	69	0.9350	8.34E-11	1	13	0.1334	0

Table 4: Test results for Examples 7–8.

PN	VAR	SP	Algorithm 1				ACGD				MDKM				SRCME				SDYCG			
			NIT	FE	PT	Norm	NIT	FE	PT	Norm	NIT	FE	PT	Norm	NIT	FE	PT	Norm	NIT	FE	PT	Norm
7	5000	x_0^1	53	200	0.2101	7.56E-11	21	192	0.1532	0	79	712	0.5303	9.73E-11	76	79	0.1909	8.28E-11	2	25	0.0338	0
	5000	x_0^2	46	186	0.2265	6.80E-11	17	157	0.1397	0	104	937	0.7143	9.86E-11	91	94	0.2226	8.75E-11	10	121	0.1459	7.98E-12
	5000	x_0^3	47	190	0.1880	9.99E-11	15	124	0.1260	0	108	973	0.6780	9.80E-11	106	109	0.2492	7.94E-11	10	121	0.1465	2.43E-11
	5000	x_0^4	13	44	0.0587	5.33E-11	17	157	0.1419	0	104	937	0.6570	9.86E-11	91	94	0.2167	8.75E-11	10	121	0.1479	7.98E-12
	5000	x_0^5	44	178	0.1873	7.26E-11	45	408	0.2961	2.83E-13	133	1198	0.8348	8.79E-11	174	177	0.4006	9.98E-11	9	109	0.1424	3.46E-11
	5000	x_0^6	1	6	0.0096	0	73	534	0.3972	0	142	1279	0.8802	9.53E-11	192	195	0.4206	8.61E-11	10	121	0.1454	6.76E-11
	10000	x_0^1	53	200	0.3128	7.56E-11	31	300	0.3714	5.28E-11	79	712	0.8057	9.62E-11	71	74	0.2977	8.85E-11	2	25	0.0625	0
	10000	x_0^2	46	186	0.2824	7.34E-11	26	247	0.2910	5.13E-12	104	937	1.0475	9.46E-11	90	93	0.3259	9.35E-11	10	121	0.2220	7.98E-12
	10000	x_0^3	48	194	0.3304	6.20E-11	12	103	0.1481	0	110	991	1.1046	8.99E-11	105	108	0.4397	9.28E-11	10	121	0.2328	2.43E-11
	10000	x_0^4	7	22	0.0506	8.84E-11	26	247	0.2894	4.18E-12	104	937	1.0184	9.46E-11	90	93	0.3275	9.35E-11	10	121	0.2525	7.98E-12
	10000	x_0^5	44	178	0.3100	7.75E-11	28	231	0.3074	0	134	1207	1.3429	9.34E-11	177	180	0.6494	8.68E-11	9	109	0.2011	3.46E-11
	10000	x_0^6	1	6	0.0157	0	74	544	0.6694	0	144	1297	1.3942	8.44E-11	194	197	0.7516	8.87E-11	10	121	0.2279	6.76E-11
	50000	x_0^1	53	200	1.6891	7.57E-11	17	147	0.7691	2.54E-11	73	658	3.0758	9.06E-11	69	72	1.1170	8.83E-11	2	25	0.2342	0
	50000	x_0^2	46	186	1.3757	7.46E-11	18	165	0.8798	0	107	964	4.5572	9.28E-11	89	92	1.4188	8.98E-11	10	121	0.9147	7.98E-12
	50000	x_0^3	48	194	1.2152	6.57E-11	30	284	1.4076	0	112	1009	4.6808	9.40E-11	104	107	1.5571	9.51E-11	10	121	0.9020	2.43E-11
	50000	x_0^4	3	10	0.0886	1.11E-12	18	165	0.8443	0	107	964	4.4980	9.28E-11	89	92	1.3599	8.98E-11	10	121	0.9147	7.98E-12
	50000	x_0^5	44	178	1.1118	8.02E-11	28	232	1.1933	0	137	1234	5.8379	9.36E-11	181	184	2.7239	9.61E-11	9	109	0.8139	3.46E-11
	50000	x_0^6	1	6	0.0441	0	94	685	3.5413	0	147	1324	6.2662	8.46E-11	199	202	3.0006	8.62E-11	10	121	0.9277	6.76E-11
8	5000	x_0^1	23	25	0.1556	7.52E-11	70	284	0.3487	8.11E-11	24	62	0.1110	2.69E-11	43	46	0.1386	5.92E-11	***	***	***	***
	5000	x_0^2	21	23	0.1221	5.58E-11	28	104	0.1570	3.45E-11	23	61	0.1212	2.62E-11	44	47	0.1399	6.58E-11	***	***	***	***
	5000	x_0^3	21	24	0.0794	9.32E-11	20	59	0.1114	6.93E-11	20	54	0.0959	4.48E-11	42	45	0.1438	5.80E-11	***	***	***	***
	5000	x_0^4	20	23	0.0671	5.55E-11	28	104	0.1447	3.45E-11	23	61	0.1168	2.62E-11	44	47	0.1388	6.58E-11	***	***	***	***
	5000	x_0^5	21	24	0.0736	9.41E-11	29	110	0.1378	7.07E-11	24	62	0.1171	2.67E-11	44	47	0.1518	7.64E-11	***	***	***	***
	5000	x_0^6	21	23	0.1076	9.59E-11	20	59	0.1096	8.60E-11	20	54	0.0993	4.89E-11	42	45	0.1308	5.74E-11	***	***	***	***
	10000	x_0^1	22	25	0.1168	8.12E-11	33	144	0.2716	5.37E-11	24	62	0.1874	2.64E-11	43	46	0.2143	7.26E-11	***	***	***	***
	10000	x_0^2	22	25	0.1308	6.35E-11	27	95	0.2227	3.72E-11	23	61	0.1746	2.58E-11	44	47	0.2515	8.08E-11	***	***	***	***
	10000	x_0^3	21	24	0.1258	5.30E-11	22	66	0.1621	3.90E-11	19	52	0.1560	2.32E-11	42	45	0.2120	7.12E-11	***	***	***	***
	10000	x_0^4	18	21	0.1217	9.01E-11	27	95	0.2210	3.72E-11	23	61	0.1935	2.58E-11	44	47	0.2332	8.08E-11	***	***	***	***
	10000	x_0^5	21	24	0.1117	4.31E-11	73	336	0.6304	9.88E-11	24	62	0.1723	2.63E-11	44	47	0.2345	9.37E-11	***	***	***	***
	10000	x_0^6	21	23	0.1119	5.68E-11	22	66	0.1639	4.96E-11	19	52	0.1725	2.59E-11	42	45	0.2244	7.05E-11	***	***	***	***
	50000	x_0^1	24	26	0.5426	3.53E-11	28	97	0.9419	2.72E-11	24	62	0.7122	2.56E-11	44	47	0.9469	6.43E-11	***	***	***	***
	50000	x_0^2	21	24	0.4761	8.87E-11	24	76	0.7661	2.67E-11	24	62	0.7165	2.53E-11	45	48	0.9792	7.16E-11	***	***	***	***
	50000	x_0^3	22	25	0.5304	4.68E-11	21	63	0.6520	4.01E-11	20	54	0.6841	4.24E-11	43	46	0.9370	6.31E-11	***	***	***	***
	50000	x_0^4	19	22	0.4547	6.54E-11	24	76	0.7589	2.67E-11	24	62	0.7456	2.53E-11	45	48	0.9650	7.16E-11	***	***	***	***
	50000	x_0^5	23	25	0.5032	3.46E-11	25	80	0.8003	3.38E-11	24	62	0.7193	2.55E-11	45	48	0.9796	8.30E-11	***	***	***	***
	50000	x_0^6	23	26	0.5042	5.09E-11	46	183	1.6076	3.29E-11	20	54	0.6270	3.17E-11	43	46	0.9348	6.25E-11	***	***	***	***

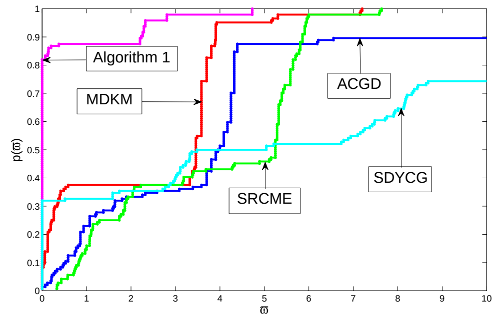

Figure 1: Dolan and More profile for number of iterations

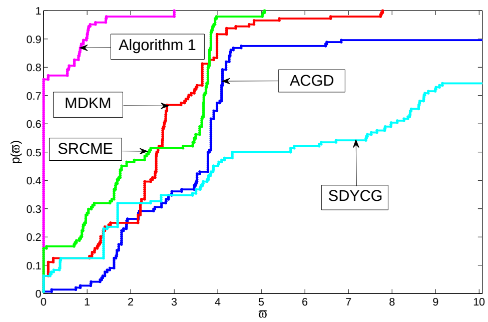

Figure 2: Dolan and More profile for function evaluations

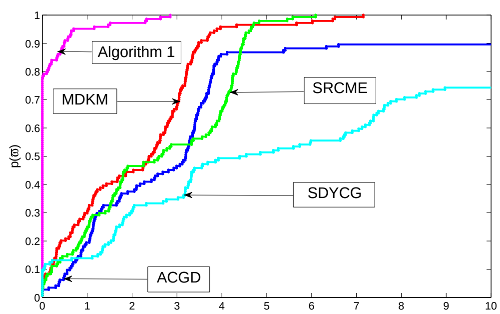

Figure 3: Dolan and More profile for CPU time

We presented results of the first experiment in Tables 1, 2, 3 and 4, where the labels PN, VAR, SP, NIT, FE, PT, and Norm represent number of test example, Dimension, Initial guess, number of iterations, function evaluations, CPU time, and norm achieved at approximate solution. Also, $***$ indicates no solution of (3) was obtained in 1000 iterations. It is clear from the four tables that Algorithm 1 outperformed the other methods in all three metrics considered. These results are further analyzed in Figures 1, 2, and 3, which are plotted by utilizing Dolan and More [23] performance profile. In Figure 1, we see that about 83% of the test examples were solved by Algorithm 1 with less iterations, while ACGD, MDKM, SRCME and SDYCG solved 2%, 8%, 0% and 32%. Furthermore, these values include instances where some of the algorithms solved 24% of the text examples with the same minimum number of iterations. Also, from Figure 2, we see that Algorithm 1 solved 77% of the test examples with minimum function evaluations compared to ACGD, MDKM, SRCME and SDYCG that recorded 1%, 5%, 16%, and 6%. Here also, some of the algorithms solved 13% of the test examples with the same minimum function evaluations. Next, we observed from Figure 3 that Algorithm 1 solved 77.78% of the test examples with the least CPU time compared to ACGD, MDKM, SRCME and SDYCG that recorded 2.78%, 6.25%, 3.47%, and 9.72%. In addition, the top curve in all three figures corresponds to that of Algorithm 1, which clearly shows that the scheme is the most effective. Moreover, the average residual for the five algorithms as computed from Tables 1, 2, 3, and 4 are given as follows: Algorithm 1 (3.09×10^{-11}), ACGD (3.36×10^{-11}), MDKM (4.56×10^{-11}), SRCME (7.58×10^{-11}), and SDYCG (3.90×10^{-10}). This, together with the other metrics analyzed, indicates that Algorithm 1 is more efficient for solving (3) than the other schemes.

4.2 Experiment 2: Image De-blurring

We use this subsection to demonstrate the application of Algorithm 1 in deblurring images contaminated by noise. To achieve the desired goal, we compare our scheme with two effective schemes in the literature, namely, HTTCGP [63] and MFRM [1].

As a background for image de-blurring, we briefly discuss sparse signal recovery, which deals with obtaining sparse solutions for the under-determined linear system $\mathcal{H}x = h$, where $\mathcal{H} \in \mathbb{R}^{k\times n}(k \ll n)$ is a sampled matrix, x a sparse signal and $h \in \mathbb{R}^k$ denotes an observed value. In recovering x from $\mathcal{H}x - h$, the following ℓ_1 *norm* regularization problem is solved:

$$\min_x f(x) := \frac{1}{2}\|h - \mathcal{H}x\|_2^2 + \zeta\|x\|_1, \tag{52}$$

with $\zeta > 0$. Careful observation reveals (52) to be a form of the problem represented in (4).

In [24], it was shown that to solve (52), it is first expressed as a convex quadratic model, where $x \in \mathbb{R}^n$ is written as

$$x = \upsilon - \nu, \quad \upsilon \geq 0, \quad \nu \geq 0, \quad \upsilon, \nu \in \mathbb{R}^n,$$

with $\upsilon_i = (x_i)_+, \nu_i = (-x_i)_+,$ for all $i = 1, 2, \ldots, n$ and $(\cdot)_+ = \max\{0, x\}$. Using this expression, we have $\|x\|_1 = E_n^T\upsilon + E_n^T\nu$ where $E_n = (1, 1, \ldots, 1)^T \in \mathbb{R}^n$. Thus, (52) becomes

$$\min\left\{\frac{1}{2}\|\mathcal{H}(\upsilon - \nu) - h\|_2^2 + \zeta(E_n^T\upsilon + E_n^T\nu)|\upsilon \geq 0, \nu \geq 0\right\}. \tag{53}$$

Now, if we define

$$w = \begin{pmatrix}\upsilon\\ \nu\end{pmatrix}, \quad \chi = \zeta E_{2n} + \begin{pmatrix}-\omega\\ \omega\end{pmatrix}, \quad \omega = \mathcal{H}^T h, \quad G = \begin{pmatrix}\mathcal{H}^T\mathcal{H} & -\mathcal{H}^T\mathcal{H}\\ -\mathcal{H}^T\mathcal{H} & \mathcal{H}^T\mathcal{H}\end{pmatrix},$$

then (53) becomes

$$\min\left\{\frac{1}{2}w^TGw + \chi^Tw| \quad w \geq 0\right\}. \tag{54}$$

Moreover, since G is a positive semi-definite matrix, (54) is a convex quadratic problem [61]. Also, based on the optimality condition mentioned earlier, w in (54) is a minimizer of (54) if it solves the system of equations

$$F(w) = \min\{w, Gw + \chi\} = 0.$$

Finally, Xiao [61] and Pang [49], showed that F satisfies (2) and (21). Hence, (52) can be represented as the problem (3), and solved using Algorithm 1.

Next, we apply Algorithm 1 to de-blur three images, which includes Einstein.tif (M1) (512×512), Cameraman.png (M2) (512×512) and Barbara.png (M3) (512×512). In the experiments, the signal-to-noise ratio (SNR)

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

and the peak to signal ratio (PSNR)

$$PSNR = 10 \times \log_{10}\frac{V^2}{MSE},$$

were used to calculate restoration quality, with V being the maximum absolute value of recovery and (MSE) is defined by

$$MSE = \frac{1}{n}\|\tilde{x} - \bar{x}\|^2, \tag{55}$$

where x is the signal recovered and $\tilde{x}$ the actual sparse one. In addition, we use MSE as defined in (55) and structured similarity index (SSIM), which describes the similarity between the original and reconstructed or recovered images to measure numerical efficiency of the algorithms. Performance of Algorithm 1 is compared with that of HTTCGP [63] and MFRM [1], which are also effective for de-blurring images, using the same parameter values in the respective papers. Parameters for Algorithm 1 are set as $\beta = 0.9$, $\delta = 0.001$, $r = 0.01$ and $\gamma = 0.25$, while ϕ retains the value in the first experiment.

Table 5: Image de-blurring results for Algorithm 1, HTTCGP, and MFRM under different Gaussian blur kernels

Image	Algorithm 1				HTTCGP				MFRM			
	MSE	SNR	PSNR	SSIM	MSE	SNR	PSNR	SSIM	MSE	SNR	PSNR	SSIM
M1(0.5)	$8.6452e+01$	20.52	29.11	0.84	$9.8875e+01$	19.94	28.25	0.82	$9.0269e+01$	20.33	28.81	0.83
M2(0.5)	$1.6736e+02$	20.31	26.10	0.83	$1.7560e+02$	20.10	25.83	0.83	$1.7312e+02$	20.16	25.97	0.83
M3(0.5)	$2.2125e+02$	18.79	24.51	0.75	$2.2442e+02$	18.73	24.43	0.74	$2.2872e+02$	18.65	24.28	0.73
M1(0.75)	$8.8465e+01$	20.42	28.93	0.83	$9.9917e+01$	19.89	28.23	0.82	$9.2995e+01$	20.21	28.63	0.82
M2(0.75)	$1.6960e+02$	20.25	26.11	0.82	$1.7060e+02$	20.23	26.06	0.82	$1.7149e+02$	20.20	26.06	0.82
M3(0.75)	$2.2348e+02$	18.75	24.49	0.74	$2.2572e+02$	18.71	24.31	0.73	$2.2573e+02$	18.71	24.30	0.73
M1(1.25)	$9.5444e+01$	20.09	28.62	0.81	$1.0258e+02$	19.78	28.10	0.81	$9.6906e+01$	20.03	28.51	0.81
M2(1.25)	$1.7558e+02$	20.10	25.86	0.78	$1.9352e+02$	19.68	25.31	0.79	$1.7748e+02$	20.06	25.81	0.79
M3(1.25)	$2.3043e+02$	18.62	24.35	0.71	$2.3006e+02$	18.63	24.34	0.71	$2.3312e+02$	18.57	24.22	0.71

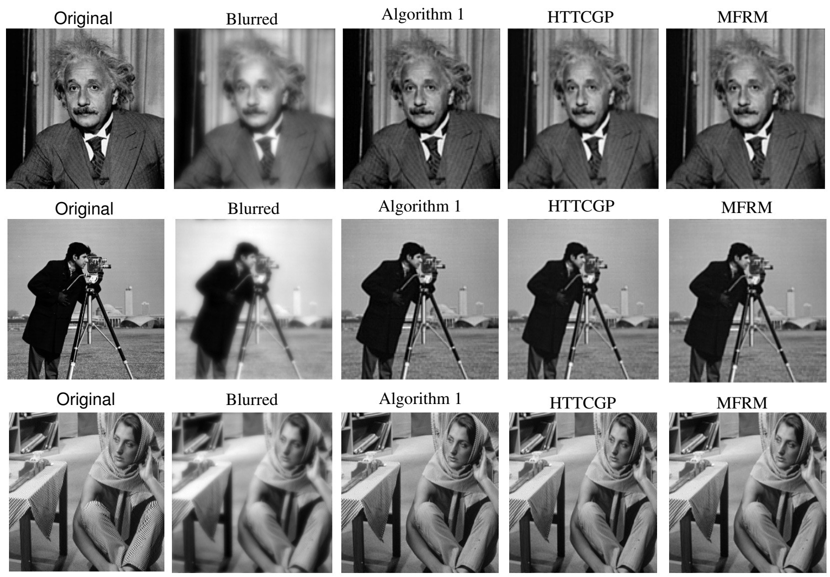

Figure 4: Recovered images under Gaussian blur kernel with standard deviation 0.5

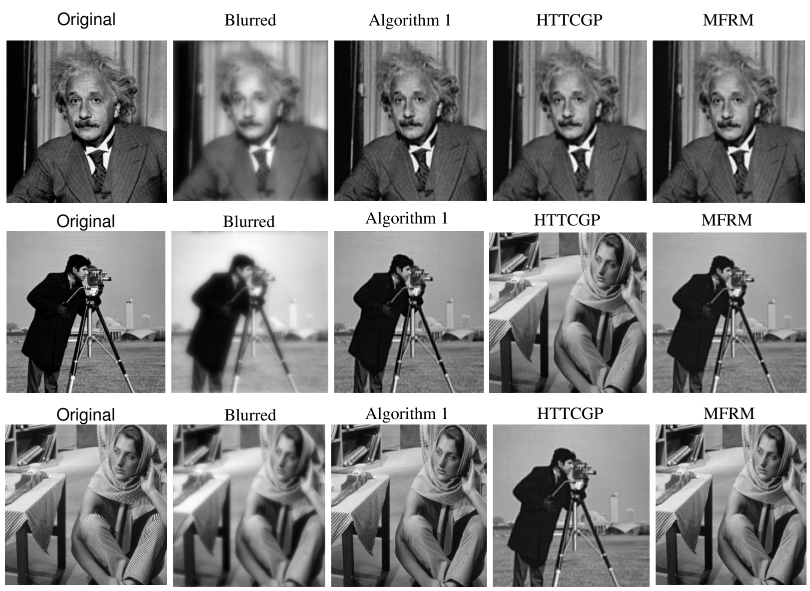

Figure 5: Recovered images under Gaussian blur kernel with standard deviation 0.75

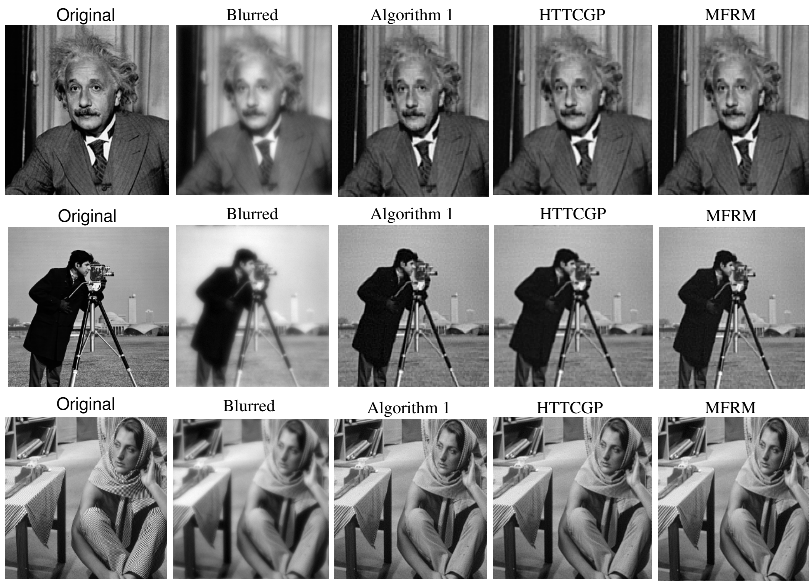

Figure 6: Recovered images under Gaussian blur kernel with standard deviation 1.25

Generally, the restored images from blurry ones by an algorithm with larger values of SNR, PSNR, and SSIM appear much closer to the original ones than algorithms with lower values of the metrics. Also, algorithms with a lower value of MSE yield better quality of restored images than algorithms with larger values of the metrics. In our experiments, Algorithm 1 yields the best values of the aforementioned performance metrics (see underlined values in Table 5). Also, the original, blurry, and recovered images by the three algorithms are presented in Figures 4, 5, and 6. Furthermore, a number of Gaussian blur kernels were used to test robustness of the algorithms (see Table 5). In Table 5, the test problem solved with standard deviation of the Gaussian blur kernel σ is given by $Mi(\sigma)$. Therefore, based on this discussion, we conclude that Algorithm 1 is effective for image recovery problems.

5 Conclusion

In this work, an adaptive DK method was considered for nonlinear monotone systems and image recovery problems. The novelty of the work is that value of the parameter of the scheme was obtained such that the eigenvalues of the symmetric form of its iteration matrix are clustered at a point. This strategy helps to ensure that the scheme's directions automatically possess the property for global convergence without any adjustment made to the derived value of the DK parameter. The method can also be used to solve nonsmooth nonlinear problems. Also, analysis of the method's convergence proved that it converges globally, while its effectiveness was shown through experiments with four other effective methods for solving constrained nonlinear problems and image deblurring. As future research, we intend to apply the proposed method to solve signal reconstruction and motion control problems.

Acknowledgements

Authors are grateful to there anonymous referees and editor for their constructive comments.

References

[1] Abubakar, A.B., Kumam, P., Mohammed, H. and Sitthithakerngkiet, K. *A modified Fletcher-Reeves conjugate gradient method for monotone nonlinear equations with some applications*, Mathematics, 7 (745) (2019), 1–25.

[2] Ahmed, K., Waziri, M.Y. and Halilu, A.S. *On two symmetric Dai-Kou type schemes for constrained monotone equations with image recovery application*, Euro J. Comput. Optim. 11 100057 (2023) 1–32.

[3] Ahmed, K., Waziri, M.Y., Halilu, A.S. and Murtala, S. *Sparse signal reconstruction via Hager-Zhang-type schemes for constrained system of nonlinear equations*, Optimization, 73 (6) (2024), 1949–1980.

[4] Ahmed, K., Waziri, M.Y., Halilu, A.S., Murtala, S. and Sabi'u, J. *Another Hager-Zhang type method via singular value study for constrained monotone equations with application*, Numer. Algor. 96 (4) (2024), 1583–1623.

[5] Ahmed, K., Waziri, M.Y., Murtala, S., Halilu, A.S. and Sabi'u, J. *On a scaled symmetric Dai-Liao-type scheme for constrained system of nonlinear equations with applications*, J. Optim. Theory Appl. 200 (2024), 669–702.

[6] Al-Baali M. *Numerical experience with a class of self-scaling quasi-Newton algorithms*, J. Optim. Theory and Appl. 96(3) (1998), 533–553.

[7] Althobaiti, A., Sabi'u J., Emadifar, H., Junsawang, P., and Sahoo, S.K. *A scaled Dai-Yuan projection-based conjugate gradient method for solving monotone equations with applications*, Symmetry, 14 (1401) (2023), 1–28.

[8] Andrei, N. *Eigenvalues versus singular values study in conjugate gradient algorithms for large-scale unconstrained optimization*, Optim. Methods Softw. 32(3)(2017), 534–551.

[9] Andrei, N. *A Dai-Liao conjugate gradient algorithm with clustering of eigenvalues*, Numer. Algor. 77 (2018), 1273–1282.

[10] Andrei, N. *A double parameter scaled BFGS method for unconstrained optimization*, J. Comput. Appl. Math. 332 (2018), 26–44.

[11] Andrei, N. *A double-parameter scaling Broyden-Fletcher-Goldfarb-Shanno method based on minimizing the measure function of Byrd and Nocedal for unconstrained optimization*, J. Optim. Theory Appl. 178 (2018), 191–218.

[12] Babaie-Kafaki, S. and Aminifard, Z. *Two parameter scaled memoryless BFGS methods with a nonmonotone choice for the initial step length*, Numer. Algor. 82 (2019), 1345–1357.

[13] Biggs, M.C. *Minimization algorithms making use of non-quadratic properties of the objective function*, J. Inst. Math. Appl. 8 (1971), 315–327.

[14] Broyden, C.G. *A class of methods for solving nonlinear simultaneous equations*, Math. Comput. 19 (1965), 577–593.

[15] Broyden, C.G. *The convergence of a class double-rank minimization algorithms*, J. Inst. Math. Appl. 6 (1970) 76–90.

[16] Byrd, R.H., Liu, D.C. and Nocedal, J. *On the behavior of Broyden's class of quasi-Newton methods*, SIAM J. Optim. 2 (1992), 533–557.

[17] Byrd, R. and Nocedal, J. *A tool for the analysis of quasi-Newton methods with application to unconstrained minimization*, SIAM J. Numer. Anal. 26 (1989), 727–739.

[18] Dai, Y.H. and Kou, CX. *A nonlinear conjugate gradient algorithm with an optimal property and an improved wolfe line search*, SIAM J. Optim. 23 (2013), 296–320.

[19] Dai, Y.H. and Liao, L.Z. *New conjugacy conditions and related nonlinear conjugate gradient methods*, Appl. Math. Optim. 43 (1) (2001), 87–101.

[20] Dai, Y.H. and Yuan, Y. *A nonlinear conjugate gradient method with a strong global convergence property*, SIAM J. Optim. 10 (1999), 177–182.

[21] Dennis, J.E. and Schnabel, R.B. *Numerical methods for unconstrained optimization and nonlinear equations*, Englewood Cliffs, NJ: Prentice-Hall, 1983.

[22] Ding, Y., Xiao, Y. and Li, J. *A class of conjugate gradient methods for convex constrained monotone equations*, Optimization, 66 (12) (2017), 2309–2328.

[23] Dolan, E.D. and More, J.J. *Benchmarking optimization software with performance profiles*, Math. Program, 91 (2002), 201–2013.

[24] Figueiredo, M., Nowak, R. and Wright, S.J. *Gradient projection for sparse reconstruction, application to compressed sensing and other inverse problems*, IEEE J-STSP IEEE Press, Piscataway, NJ. (2007), 586–597.

[25] Fletcher, R. and Reeves, C. *Function minimization by conjugate gradients*, Comput. J. 7 (1964), 149–154.

[26] Fletcher, R. *A new approach to variable metric algorithms*, Computer J. 13 (1970) 317–322.

[27] Fletcher, R. *Practical method of optimization*, Volume 1: Unconstrained Optimization, 2nd ed., Wiley, New York, (1997).

[28] Gao, P., Zheng, W., Wang, T., Li, Y. and Li, F. *Signal recovery with constrained monotone nonlinear equations*, J. Appl. Anal. Comput. 13 (4) (20023), 2006–2025.

[29] Gill, P.E. and Leonard, M.W. *Reduced-Hessian quasi Newton methods for unconstrained optimization*, SIAM J. Optim. 12 (2001), 209–237.

[30] Goldfarb, D. *A family of variable metric methods derived by variation mean*, Math. Comput. 23 (1970), 23–26.

[31] Hager, W.W. and Zhang, H. *A survey of nonlinear conjugate gradient methods*, Pacific J. Optim. 2 (2006), 35–58.

[32] Halilu, A.S., Majumder, A., Waziri, M.Y., Ahmed, K. and Awwal, A.M. *Motion control of the two joint planar robotic manipulators through accelerated Dai-Liao method for solving system of nonlinear equations*, Eng. Comput. 39 (5) (2021), 1–39.

[33] Hestenes, M.R. and Stiefel, E.L. *Methods of conjugate gradients for solving linear systems*, J. Research Nat. Bur. Standards, 49 (1952), 409–436.

[34] Ivanov, B., Milanovic, G.V. and Stanimirovic, P.S. *Accelerated Dai-Liao projection method for solving systems of monotone nonlinear equations with application to image deblurring*, J. Global Optim. 85 (2023) 377–420.

[35] Kaporin, I.E. *New convergence results and preconditioning strategies for the conjugate gradient methods*, Numer. Linear Alg. Appl. 1(2) (1994), 179–210.

[36] Kiri, A.I., Waziri, M.Y. and Ahmed, K. *Amodified Liu-Storey scheme for nonlinear systems with an application to image recovery*, Iran, J. Numer. Anal. and Optim. 3 (1) (2023), 38–58.

[37] Kratzer, D., Parter, S.V. and Steuerwalt, M. *Block splittings for the conjugate gradient method*, Comp. Fluid, 11 (1983), 255–279.

[38] La Cruz, W. *A Spectral algorithm for large-scale systems of nonlinear monotone equations*, Numer. Algor. 76 (2017), 1109–1130.

[39] La Cruz, W., Martinez, J.M. and Raydan, M. *Spectral residual method without gradient information for solving large-scale nonlinear systems of equations*, Theory and experiments, Technical Report RT-04-08, 2005.

[40] Liu, J.K., and Li, S.J. *A projection method for convex constrained monotone nonlinear equations with applications*, Comput. Math. Appl. 70 (10) (2015), 2442–2453.

[41] Liu, Y. and Storey, C. *Efficient generalized conjugate gradient algorithms*, Part 1: Theory, J. Optim. Theory Appl. 69 (1991), 129–137.

[42] Narushima, Y. and Yabe, H. *A survey of sufficient descent conjugate gradient methods for unconstrained optimization*, SUT J. Math. 50(2) (2014), 167–203.

[43] Nocedal, J. *Theory of algorithms for unconstrained optimization*, Acta Numer. 1 (1992), 199–242.

[44] Nocedal, J. and Wright, S.J. *Numerical optimization*, Springer, New York, 2006.

[45] Oren, S.S. and Luenberger, D.G. *Self-scaling variable metric (SSVM) algorithms*, part I: criteria and sufficient conditions for scaling a class of algorithms, Manag. Sci. 20 (1974), 845–862.

[46] Oren S.S., and Luenberger D.G. *Self scaling variable metric (SSVM) algorithms*, part I: criteria and sufficient conditions for scaling a class of algorithms, Manag. Sci. 20(5) (1974), 845–862.

[47] Oren S.S., and Spedicato E. *Optimal conditioning of self scaling variable metric algorithms*, Math. Prog. 10(1) (1976), 70–90.

[48] Ortega, J.M. and Rheinboldt, W.C. *Iterative solution of nonlinear equations in several variables*, New York: Academic Press, 1970.

[49] Pang, J.S. *Inexact Newton methods for the nonlinear complementarity problem*, Math. Program. 36 (1986), 54–71.

[50] Polak, E. and Ribiére, G. *Note Sur la convergence de directions conjugèes*, Rev. Francaise Informat. Recherche Operationelle, 3e Annè. 16 (1969), 35–43.

[51] Polyak, B.T. *The conjugate gradient method in extreme problems*, USSR Comp. Math. Math. Phys. 9 (1969), 94–112.

[52] Powell, M.J.D. *Some global convergence properties of a variable metric algorithm for minimization without exact line search*, In: Cottle, R.W., Lemke, C.E. (eds.) Nonlinear Programming, SIAM-AMS Proceedings, SIAM, Philadelphia. 9 (1976), 53–72.

[53] Shanno, D.F. *Conditioning of quasi-Newton methods for function minimization*, Math. Comput. 24 (1970), 647–656.

[54] Solodov, M.V. and Svaiter, B.F. *A globally convergent inexact Newton method for systems of monotone equations*, in: M. Fukushima, L. Qi (Eds.), Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods, Kluwer Academic Publishers. (1998), 355–369.

[55] Sun, W. and Yuan, Y.X. *Optimization theory and methods*, Nonlinear programming, Springer, New York 2006.

[56] Waziri, M.Y., Ahmed, K. and Halilu, A.S. *A modified Dai-Kou-type method with applications to signal reconstruction and blurred image restoration*, Comput. Appl. Math. 41(232) (2022), 1–33.

[57] Waziri, M.Y., Ahmed, K., Halilu, A.S. and Sabi'u, J. *Two new Hager-Zhang iterative schemes with improved parameter choices for monotone nonlinear systems and their applications in compressed sensing.* Rairo Oper. Res. 56 (1) (2021), 239–273.

[58] Waziri, M.Y., Ahmed, K. and Sabi'u, J. *A Dai-Liao conjugate gradient method via modified secant equation for system of nonlinear equations*, Arab. J. Math. 9 (2020), 443–457.

[59] Waziri, M.Y., Ahmed, K., Sabi'u, J. and Halilu, A.S. *Enhanced Dai-Liao conjugate gradient methods for systems of monotone nonlinear equations*, SeMA J. 78 (2020), 15–51.

[60] Winther, R. *Some superlinear convergence results for the conjugate gradient method*, SIAM J. Numer. Anal. 17 (1980), 14–17.

[61] Xiao, Y., Wang, Q. and Hu, Q. *Non-smooth equations based method for $\ell_1 -$ norm problems with applications to compressed sensing*, Nonlinear Anal. Theory Methods Appl. 74(11) (2011), 3570–3577.

[62] Xiao, Y. and Zhu, H. *A conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing*, J. Math. Anal. Appl. 405 (1) (2013), 310–319.

[63] Yin, J., Jian, J., Jiang, X., Liu, M. and Wang, L. *A hybrid three-term conjugate gradient projection method for constrained nonlinear monotone equations with applications*, Numer. Algor. 88 (2021), 389–418.

[64] Zhou, W.J. and Li, D.H. *Limited memory BFGS methods for nonlinear monotone equations*, J. Comput. Math. 25 (2007), 89–96.