Hindawi
Journal of Mathematics
Volume 2023, Article ID 9945581, 20 pages
https://doi.org/10.1155/2023/9945581

Research Article

A Modified Dai–Liao Conjugate Gradient Method Based on a Scalar Matrix Approximation of Hessian and Its Application

Branislav Ivanov,[1] **Gradimir V. Milovanović**,[2,3] **Predrag S. Stanimirović**,[3,4] **Aliyu Muhammed Awwal**,[5,6] **Lev A. Kazakovtsev**,[4] **and Vladimir N. Krutikov**[7]

[1]*University of Belgrade, Technical Faculty in Bor, Vojske Jugoslavije 12, Bor 19210, Serbia*
[2]*Serbian Academy of Sciences and Arts, Mathematical Institute, Kneza Mihaila 35, Belgrade 11000, Serbia*
[3]*University of Niš, Faculty of Sciences and Mathematics, Višegradska 33, Niš 18000, Serbia*
[4]*Laboratory "Hybrid Methods of Modelling and Optimization in Complex Systems", Siberian Federal University, Prosp. Svobodny 79, Krasnoyarsk 660041, Russia*
[5]*Department of Mathematics, Faculty of Science, King Mongkut's University of Technology Thonburi (KMUTT), 126 Pracha-Uthit Road, Bang Mod, Thrung Khru, Bangkok 10140, Thailand*
[6]*Department of Mathematics, Faculty of Science, Gombe State University, Gombe 760214, Nigeria*
[7]*Kemerovo State University, 6 Krasnaya Street, Kemerovo 650043, Russia*

Correspondence should be addressed to Predrag S. Stanimirović; pecko@pmf.ni.ac.rs

Received 24 August 2022; Revised 30 January 2023; Accepted 6 February 2023; Published 8 April 2023

Academic Editor: Xian-Ming Gu

Copyright © 2023 Branislav Ivanov et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We introduce and investigate proper accelerations of the Dai–Liao (DL) conjugate gradient (CG) family of iterations for solving large-scale unconstrained optimization problems. The improvements are based on appropriate modifications of the CG update parameter in DL conjugate gradient methods. The leading idea is to combine search directions in accelerated gradient descent methods, defined based on the Hessian approximation by an appropriate diagonal matrix in quasi-Newton methods, with search directions in DL-type CG methods. The global convergence of the modified Dai–Liao conjugate gradient method has been proved on the set of uniformly convex functions. The efficiency and robustness of the newly presented methods are confirmed in comparison with similar methods, analyzing numerical results concerning the CPU time, a number of function evaluations, and the number of iterative steps. The proposed method is successfully applied to deal with an optimization problem arising in 2D robotic motion control.

1. Introduction and Overview of Related Results

Our research area is the large-scale multivariable unconstrained optimization problem.

$$\min f(x), \quad x \in \mathbb{R}^n, \tag{1}$$

in which the function $f\colon \mathbb{R}^n \longrightarrow \mathbb{R}$ is uniformly convex and twice continuously differentiable. Quasi-Newton (QN) methods and conjugate gradient (CG) methods are the two most popular approaches in solving nonlinear optimization problems.

Various and numerous modifications of Dai–Liao (DL) conjugate gradient (CG) methods [1] with acceleration parameters arise from the natural demand for solving large-scale problems (1). The motivation of this research is based on the wide applications of unconstrained optimization problems and the efficiency of conjugate gradient methods for solving them [2–11]. The main result obtained in this study is the verification and investigation of the correlation between QN and CG approaches. More specifically, in this research, we study the possibilities of applying QN methods in improving CG-type algorithms [12–16].

The generic iterative scheme that aimed to solve (1) is as follows:

$$x_{k+1} = x_k + \alpha_k d_k, \tag{2}$$

where x_k is the previous iterative point, x_{k+1} is a new iterative point, $g_k = \nabla f(x_k)$ is the gradient vector in x_k, d_k is a search direction defined upon the descent condition $g_k^T d_k < 0$, and $\alpha_k > 0$ is a step length. The basic descent direction is the direction opposite to the gradient $d_k = -g_k$, which leads to the template of gradient descent (GD) iterations [17, 18].

$$x_{k+1} = x_k - \alpha_k g_k, \tag{3}$$

in which α_k is defined by the backtracking line search.

Algorithm 1 from [19] is selected as a framework for implementing the inexact line search which determines the step length α_k.

The starting point of our investigation is iterations of the Newton method with line search.

$$x_{k+1} = x_k - \alpha_k G_k^{-1} g_k, \tag{4}$$

where G_k^{-1} is the inverse of the Hessian $G_k = \nabla^2 f(x_k)$. The quasi-Newton type iterations

$$x_{k+1} = x_k - \alpha_k B_k^{-1} g_k, \tag{5}$$

are based on the assumption that B_k (resp., H_k) is an appropriate symmetric positive definite estimation of G_k (resp., G_k^{-1}) [18]. The update from B_k to B_{k+1} is specified on the quasi-Newton property (secant equation)

$$B_{k+1} s_k = y_k, \quad \text{where } s_k = x_{k+1} - x_k, y_k = g_{k+1} - g_k. \tag{6}$$

The quasi-Newton methods based on matrix approximations of B_k show some shortcomings in solving large-scale problems due to the requirement to compute and store matrices during iterations. Because of that, we choose the simplest scalar approximation of G_k according to the classification presented in [20]. Therefore,

$$B_k = \gamma_k I, \quad \gamma_k > 0. \tag{7}$$

This defines the simplest and numerically efficient approximation of G_k by the identity matrix I and approximate scalar $\gamma_k > 0$. Such reduction results in the iterative flow as follows:

$$x_{k+1} = x_k - \alpha_k \gamma_k^{-1} g_k. \tag{8}$$

One efficient definition of γ_k was proposed in [19] based on the Taylor expansion of the objective function f, resulting in

$$\gamma_{k+1}^{\mathrm{SM}} = 2\gamma_k^{\mathrm{SM}} \frac{\gamma_k^{\mathrm{SM}} [f(x_{k+1}) - f(x_k)] + \alpha_k \|g_k\|^2}{\alpha_k^2 \|g_k\|^2}. \tag{9}$$

Initiated SM iterations of the form (8) and (9)

$$x_{k+1} = x_k - \alpha_k \left(\gamma_k^{\mathrm{SM}}\right)^{-1} g_k, \tag{10}$$

were defined in [19].

Furthermore, the next modified SM (MSM) scheme was proposed in [21], using the output α_k of the backtracking Algorithm 1 and the gain parameter $\beth_k := 1 + \alpha_k - \alpha_k^2 > 1$ in the form of iterates.

$$x_{k+1} = x_k - \alpha_k \beth_k \left(\gamma_k^{\mathrm{MSM}}\right)^{-1} g_k, \tag{11}$$

where γ_k^{MSM} was defined in [21] by the rule

$$\gamma_{k+1}^{\mathrm{MSM}} = 2\gamma_k^{\mathrm{MSM}} \frac{\gamma_k^{\mathrm{MSM}} [f(x_{k+1}) - f(x_k)] + \alpha_k \beth_k \|g_k\|^2}{(\alpha_k \beth_k)^2 \|g_k\|^2}. \tag{12}$$

Since $\alpha_k \in (0, 1]$, the main idea used in the MSM iterates is to accelerate the SM iterations by the parameter $\beth_k := 1 + \alpha_k - \alpha_k^2 \geq 1$. More details about accelerated gradient methods can be found in [19, 22, 23]. Since $\beth_k'(\alpha_k) = 0$ for $\alpha_k = (1/2)$ and $\beth_k(0) = \beth_k(1) = 1$, mathematical analysis of the function $\beth_k(\alpha_k)$ in the interval $\alpha_k \in (0, 1]$ reveals $\max \beth_k(\alpha_k) = \beth_k(1/2) = (5/4)$ and $1 \leq \beth_k \leq (5/4)$. Figure 1 presents the graph of $\beth_k(\alpha_k)$ for $\alpha_k \in (0, 1]$.

We observe that the choice $\alpha_k := 1$ reduces iterations (8) to a kind of the GD iterative rule.

$$x_{k+1} = x_k - \gamma_k^{-1} g_k, \tag{13}$$

in which γ_k can be determined in various approaches. Barzilai and Borwein in [29] suggested two mutually dual variations of the GD method, known as BB iterations, defined by the step length $\gamma_k^{BB} := \gamma_k^{-1}$ in (13) equal to

$$\gamma_k^{BB1} = \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}, \tag{14}$$

$$\gamma_k^{BB2} = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}. \tag{15}$$

Suitable adaptive strategies for choosing among the first and the second BB step length enhance greatly the performance of the BB method [30]. The *BB* method has been improved in numerous articles, such as [31, 32].

In this research, the acceleration parameters $\beth_k$ and γ_k^{MSM}, used in the iterative process (11), will be exploited to improve the efficiency of the DL conjugate gradient method which is based on the rule (2) with the search direction

$$d_k = \begin{cases} -g_0, & k = 0, \\ -g_k + \beta_k^{\mathrm{DL}} d_{k-1}, & k \geq 1. \end{cases} \tag{16}$$

Determined by the real parameter

Require: Objective $f(x)$, the search direction d_k at the point x_k, and real numbers $\beta \in (0,1)$ and $0 < \sigma < 0.5$.
(1) Initialize α: $= 1$.
(2) While $f(x_k + \alpha d_k) > f(x_k) + \sigma \alpha g_k^T d_k$, update α: $= \alpha\beta$.
(3) Return $\alpha_k = \alpha$.

Algorithm 1: The backtracking line search.

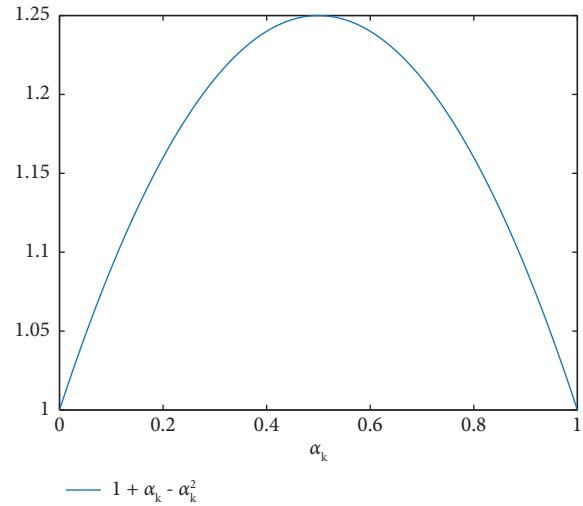

Figure 1: The graph of the function $\lambda(\alpha_k) = 1 + \alpha_k - \alpha_k^2$ for $\alpha_k \in (0,1]$.

$$\beta_k^{DL} = \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} - t\frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}}, \quad t > 0. \tag{17}$$

The parameter β_k^{DL} is known as the CG update parameter. Table 6 in appendix shows the abbreviations and full names of the methods considered in this paper.

The conjugation condition

$$d_k^T y_{k-1} = -t g_k^T s_{k-1}, \quad t > 0, \tag{18}$$

was introduced in [1] by Dai and Liao. The condition (18) has been an inspiration for many researchers, of which the most important are Hager and Zhang [24, 25], Dai and Kou [26], Babaie–Kafaki and Ghanbari [27], Ivanov et al. [28], Lotfi and Hosseini [15], and Zheng and Zheng [33] to create new DL-type CG methods.

Some of the most significant rules to determine t are collected in Table 1. The diversity in definitions of the DL parameter t is confirmed in Table 1. The parameter t in the MDL method proposed in [15] is based on the improvement of the Dai–Liao CG class by a modified BFGS method.

But not all possibilities are exhausted. Since the line search used in this research gives the output $\alpha_k \in (0,1)$, it follows $\lambda_k > 1$ and consequently λ_k in common with γ_k^{MSM} are useful in the proposal of a novel rule which determines the CG parameter t. Our main idea is to find the DL parameter t in (17) after the unification of descent directions in the MSM method (11) and in the DL iterations (17). In the present manuscript, we use an original approach which is developed on the unification of two search directions included in the MSM method or in the BB method (which belongs to the class of quasi-Newton methods) and the DL method (from the CG class). The unification of the MSM and DL methods leads to an equation with respect to the unknown parameter t whose solution gives a new DL parameter and corresponding DL-type method of the CG class, termed as the MSMDL method. On the other hand, the hybridization of the BB1 method with the DL class leads to the BB1DL method.

Main contributions achieved in this article are highlighted as follows:

(1) A novel approach to finding the DL parameter t is proposed, based on the equalization of search directions included in a diagonal matrix approximation of quasi-Newton methods with the search direction from the CG class;
(2) Convergence analysis of the proposed MSMDL method is conducted under standard assumptions;
(3) Numerical examples on standard test examples are presented with the aim to show the effectiveness of the proposed MSMDL and BB1DL methods.

The global contents of the remaining sections are as follows: in Section 2, we present an algorithm for the MSMDL method for solving unconstrained optimization problems with a new CG parameter t_k^{MSMDL} which contains an acceleration parameter from the MSM quasi-Newton method. Section 3 explores the convergence properties of the presented MSMDL method. Some numerical results are

Table 1: Some of the most significant rules to determine t.

Parameters t_k	Methods	Reference
$t_k = 2(\|y_{k-1}\|^2/y_{k-1}^T s_{k-1})$	CG-DESCENT	[24, 25]
$t_k = \tau_k + (\|y_{k-1}\|^2/y_{k-1}^T s_{k-1}) - (y_{k-1}^T s_{k-1}/\|s_{k-1}\|^2)$	DK	[26]
$t_k = (s_{k-1}^T y_{k-1}/\|s_{k-1}\|^2) + (\|y_{k-1}\|/\|s_{k-1}\|)$	M1	[27]
$t_k = \max\{t_k^*, \theta(\|y_{k-1}\|^2/s_{k-1}^T y_{k-1})\}$ $t_k^* = ((1-h_k\|g_{k-1}\|^r)s_{k-1}^T g_k + (g_k^T y_{k-1}/y_{k-1}^T s_{k-1})h_k\|g_{k-1}\|^r\|s_{k-1}\|^2/g_k^T s_{k-1} +$ $(g_k^T s_{k-1}/s_{k-1}^T y_{k-1})h_k\|g_{k-1}\|^r\|s_{k-1}\|^2)$ $h_k = C + \max\{-(s_{k-1}^T y_{k-1}/\|s_{k-1}\|^2), 0\}\|g_{k-1}\|^{-r}$ $\theta > 1/4, C > 0, r > 0$	MDL	[15]
$t_k = (\|g_k\|^2/\max\{1, d_{k-1}^T g_k\} + (\max\{0, d_{k-1}^T g_k/\|g_k\|^2\} + 1)\|g_k\|^2)$	EDL	[28]

proposed and discussed in Section 4 as well as a comparison of the suggested methods against some similar existing methods. The application of the suggested MSMDL method on 2D robotic motion control is discussed in Section 5. Some final conclusions and discussion are stated in Section 6.

2. New Dai–Liao CG Method with the Acceleration Parameter

The first basis of the proposed iterations is the MSM scheme (11) for solving unconstrained optimization (1). In order to fulfill the *Second-Order Necessary Condition* and *Second-Order Sufficient Condition*, inappropriate values $\gamma_{k+1}^{\mathrm{MSM}} \le 0$ which appear in (12) will be replaced by $\gamma_{k+1}^{\mathrm{MSM}} = 1$. To avoid such situations, in accordance with [19, 21], the following acceleration parameter will be used:

$$\gamma_{k+1}^{\mathrm{MSM}} = \begin{cases} \gamma_{k+1}^{\mathrm{MSM}}, & \gamma_{k+1}^{\mathrm{MSM}} > 0, \\ 1, & \gamma_{k+1}^{\mathrm{MSM}} \le 0. \end{cases} \tag{19}$$

The resulting iterations are termed as MSM iterative scheme and defined by

$$\begin{aligned} x_{k+1} &= x_k - \alpha_k \lambda_k \left(\gamma_k^{\mathrm{MSM}}\right)^{-1} g_k \\ &= x_k + \alpha_k d_k^{\mathrm{MSM}}. \end{aligned} \tag{20}$$

Therefore, the search direction underlying in the MSM method is determined by the vector

$$d_k^{\mathrm{MSM}} = -\lambda_k \left(\gamma_k^{\mathrm{MSM}}\right)^{-1} g_k, \tag{21}$$

where the parameter γ_k^{MSM} is defined in (12) using the Taylor expansion as in [21].

On the other hand, the CG update parameter t from (17) can be determined by putting (17) into (16), which leads to

$$\begin{aligned} d_k^{\mathrm{DL}} &:= -g_k + \beta_k^{DL} d_{k-1} \\ &= -g_k + \left(\frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} - t\frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}}\right) d_{k-1}. \end{aligned} \tag{22}$$

After equalization of $d_k^{\mathrm{MSM}} := -\lambda_k (\gamma_k^{\mathrm{MSM}})^{-1} g_k$ from (21) with d_k^{DL} from (22), the following equation with respect to the unknown t is obtained:

$$-\lambda_k \left(\gamma_k^{\mathrm{MSM}}\right)^{-1} g_k = -g_k + \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} d_{k-1} - t\frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}} d_{k-1}. \tag{23}$$

Our idea is to find the Dai–Liao parameter t as a solution to equation (23). Application of the scalar product by g_k^{T} on the left- and right-hand side in the equation (23) gives the following equation with respect to t:

$$\begin{aligned} -\lambda_k \left(\gamma_k^{\mathrm{MSM}}\right)^{-1} g_k^T g_k &= -g_k^T g_k + \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} g_k^T d_{k-1} \\ &- t\frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}} g_k^T d_{k-1}. \end{aligned} \tag{24}$$

Thus, on the basis of (24), it further follows

$$\begin{aligned} t\frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}} g_k^T d_{k-1} &= -\|g_k\|^2 + \lambda_k \left(\gamma_k^{\mathrm{MSM}}\right)^{-1} \|g_k\|^2 + \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} g_k^T d_{k-1} \\ &= \left(\lambda_k \left(\gamma_k^{\mathrm{MSM}}\right)^{-1} - 1\right)\|g_k\|^2 + \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} g_k^T d_{k-1}. \end{aligned} \tag{25}$$

Now, the parameter t is expressed from the equation (25) as follows:

$$t = \frac{\left(\lambda_k\left(\gamma_k^{\mathrm{MSM}}\right)^{-1} - 1\right)\|g_k\|^2 + \left(g_k^T y_{k-1}/d_{k-1}^T y_{k-1}\right) g_k^T d_{k-1}}{\left(g_k^T s_{k-1}/d_{k-1}^T y_{k-1}\right) g_k^T d_{k-1}}. \quad (26)$$

Since $s_{k-1} = x_k - x_{k-1} = x_{k-1} + \alpha_{k-1} d_{k-1} - x_{k-1} = \alpha_{k-1} d_{k-1}$ and $\alpha_{k-1} > 0$, after the substitution of $d_{k-1} = (s_{k-1}/\alpha_{k-1})$ in (26), the following solution is obtained after some simplifications:

$$\begin{aligned} t &= \frac{\left(\lambda_k\left(\gamma_k^{\mathrm{MSM}}\right)^{-1} - 1\right)\|g_k\|^2 + \left(g_k^T y_{k-1}/(1/\alpha_{k-1}) s_{k-1}^T y_{k-1}\right) g_k^T (1/\alpha_{k-1}) s_{k-1}}{\left(g_k^T s_{k-1}/(1/\alpha_{k-1}) s_{k-1}^T y_{k-1}\right) g_k^{\mathrm{T}} (1/\alpha_{k-1}) s_{k-1}} \\ &= \frac{\left(\lambda_k\left(\gamma_k^{\mathrm{MSM}}\right)^{-1} - 1\right)\|g_k\|^2 + \left(g_k^T y_{k-1}/s_{k-1}^T y_{k-1}\right) g_k^T s_{k-1}}{\left(\left(g_k^T s_{k-1}\right)^2/s_{k-1}^T y_{k-1}\right)} \\ &= \frac{\left(\lambda_k\left(\gamma_k^{\mathrm{MSM}}\right)^{-1} - 1\right)\|g_k\|^2 s_{k-1}^T y_{k-1} + g_k^T y_{k-1} g_k^T s_{k-1}}{\left(g_k^T s_{k-1}\right)^2} \\ &= \tau_k^{\mathrm{MSM}}. \end{aligned} \quad (27)$$

It is known that the DL parameter is calculated to generate the direction of maximum enhancement, utilizing the search direction matrix to be orthogonal to the gradient vector [34]. To make sure that the new DL method satisfies the descent condition, the definition of t in (27) is altered using concepts from [15, 34] in the final form:

$$t_k^{\mathrm{MSMDL}} = \max\left\{\tau_k^{\mathrm{MSM}}, \theta \frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}}\right\}, \quad \theta > \frac{1}{4}. \quad (28)$$

Considering $t := t_k^{\mathrm{MSMDL}}$ in (17), the following improvement of the Dai–Liao CG parameter β_k^{DL} is proposed:

$$\beta_k^{\mathrm{MSMDL}} = \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} - t_k^{\mathrm{MSMDL}} \frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}}. \quad (29)$$

The MSMDL method is based on (2), (6), (28), and (29). The algorithmic procedure of the MSMDL method is established in Algorithm 2.

The previous strategy for combining MSM and DL approaches can be applied to any quasi-Newton direction. If we replace $\gamma_{k+1}^{\mathrm{MSM}}$ by γ_k^{BB1} from (14), we get a new BB1DL method. An analogous calculation gives

$$\begin{aligned} t &= \frac{\left(\left(\gamma_k^{\mathrm{BB1}}\right)^{-1} - 1\right)\|g_k\|^2 s_{k-1}^T y_{k-1} + g_k^T y_{k-1} g_k^T s_{k-1}}{\left(g_k^T s_{k-1}\right)^2} \\ &= \tau_k^{\mathrm{BB1}}, \end{aligned}$$

$$t_k^{\mathrm{BB1DL}} = \max\left\{\tau_k^{\mathrm{BB1}}, \theta \frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}}\right\}, \quad \theta > \frac{1}{4}. \quad (30)$$

Furthermore, the replacement $t := t_k^{\mathrm{BB1DL}}$ in (17) initiates the following improvement of the Dai–Liao CG parameter β_k^{DL}:

$$\beta_k^{\mathrm{BB1DL}} = \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} - t_k^{\mathrm{BB1DL}} \frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}}. \quad (31)$$

If we apply the mentioned changes, we arrive at another variant of Algorithm 2, where the following steps are used instead of steps 6, 7, 8, and 9 in the MSMDL method:

Step 6*: We compute $\gamma_{k+1}^{\mathrm{BB1}}$ using (14).
Step 7*: We compute t_{k+1}^{BB1DL} using (30).
Step 8*: We compute $\beta_{k+1}^{\mathrm{BB1DL}}$ using (31).
Step 9*: We compute $d_{k+1} = -g_{k+1} + \beta_{k+1}^{\mathrm{BB1DL}} d_k$.

The variant of Algorithm 2 based on steps 1–5, 6*, 7*, 8*, 9*, 10, and 11: will be called the BB1DL method. More precisely, the BB1DL method is based on (2), (16), (30), and (31).

The numerical results in Section 4 show the effectiveness of the BB1DL method.

Based on all the previous discussion, we can conclude that the general framework presented in Algorithm 2 is applicable to other quasi-Newton methods.

3. Convergence Analysis

The global convergence of the proposed variant of CG methods is derived upon the standard assumptions.

Assumption 1

(1) The level set $\mathcal{M} = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$, defined by the initial guess x_0 and (2), is bounded.

Require: Goal function $f(x)$, initial approximation $x_0 \in \mathrm{dom}(f)$, and parameters $0 < \varepsilon \ll 1$, $0 < \delta \ll 1$.
(1) We set $k = 0$, $\gamma_0 = 1$ and calculate $f(x_0)$, $g_0 = \nabla f(x_0)$, $d_0 = -g_0$.
(2) If test criteria $\|g_k\| \le \varepsilon$ and $(|f(x_{k+1}) - f(x_k)|/1 + |f(x_k)|) \le \delta$ are fulfilled then go to step 11: and stop; else, go to the step 3.
(3) We compute α_k customizing Algorithm 1.
(4) We compute $x_{k+1} = x_k + \alpha_k d_k$.
(5) We compute $f(x_{k+1})$ and $g_{k+1} = \nabla f(x_{k+1})$.
(6) We compute $\gamma_{k+1}^{\mathrm{MSM}}$ using (19).
(7) We compute t_{k+1}^{MSMDL} using (28).
(8) We compute $\beta_{k+1}^{\mathrm{MSMDL}}$ using (29).
(9) We compute $d_{k+1} = -g_{k+1} + \beta_{k+1}^{\mathrm{MSMDL}} d_k$.
(10) We set k: $= k + 1$ and go to step 2.
(11) We return x_{k+1} and $f(x_{k+1})$.

Algorithm 2: MSMDL method.

(2) The objective f is continuous and differentiable in a neighborhood $\mathscr{P}$ of M with the Lipschitz continuous gradient g. As a consequence, there exists a positive constant $L > 0$ such that

$$(\forall u, v \in \mathscr{P}) \quad \|g(u) - g(v)\| \le L\|u - v\|. \tag{32}$$

Assumption 1 ensures the existence of positive values D and γ which fulfill

$$(\forall u, v \in \mathscr{P}) \quad \|u - v\| \le D, \tag{33}$$

$$(\forall u, v \in \mathscr{P}) \quad \|g(u)\| \le \gamma. \tag{34}$$

Another main element in proving the convergence of a CG method is the property

$$s_{k-1}^T y_{k-1} \ge \eta \left\| s_{k-1} \right\|^2, \tag{35}$$

of uniformly convex functions, where $\eta > 0$. The verification of this property can be found in Theorem 1.3.16 of [18]. By (32), it follows $\|y_{k-1}\| \le L\|s_{k-1}\|$, which in conjunction with (35) initiates

$$\eta \left\| s_{k-1} \right\|^2 \le s_{k-1}^T y_{k-1} \le L \left\| s_{k-1} \right\|^2. \tag{36}$$

Clearly, the inequality (36) implies $\eta \le L$. Furthermore, (36) initiates

$$s_{k-1}^T y_{k-1} = \alpha_{k-1} d_{k-1}^T y_{k-1} > 0. \tag{37}$$

Taking into account $\alpha_{k-1} > 0$ and (37), we conclude

$$d_{k-1}^T y_{k-1} > 0. \tag{38}$$

The statement of Lemma 1 will be useful in the verification of main statements. It can be verified on the basis of the results obtained in Lemma 2.2 in [50] and [35].

Lemma 1. *Let the Assumption 1 be satisfied and the sequence $\{x_k\}$ be generated by the MSMDL method (2), (16), and (29). Then, it holds*

$$\sum_{k=0}^{\infty} \frac{\left\| g_k \right\|^4}{\left\| d_k \right\|^2} < +\infty. \tag{39}$$

Lemma 2. *Let Assumption 1 hold, f be uniformly convex, and the CG parameter (29) fulfils $t_k^{MSMDL} \ge \theta(\|y_{k-1}\|^2 / s_{k-1}^T y_{k-1})$, for all $k \ge 0$ and for some constant $\theta > (1/4)$.*

Then, MSMDL satisfies the sufficient descent condition

$$g_k^T d_k \le -c \left\| g_k \right\|^2, \tag{40}$$

with $c = 1 - (1/4\theta)$.

Proof. Assumption 1 guarantees (38) for search directions (16) in the proposed MSMDL method. The inequality (40) will be confirmed by the induction. For $k = 0$, it follows that $g_0^T d_0 = -\|g_0\|^2 \le -c\|g_0\|^2$. So, (40) is fulfilled for $k = 0$. We assume that (52) is satisfied for k. Multiplying the identity (16) in the case $k + 1$ and corresponding to the MSMDL method by g_{k+1}^T, it can be derived

$$\begin{aligned} g_{k+1}^T d_{k+1} &= -\left\| g_{k+1} \right\|^2 + \beta_{k+1}^{\mathrm{MSMDL}} g_{k+1}^T d_k \\ &= -\left\| g_{k+1} \right\|^2 + \left(\frac{g_{k+1}^T y_k}{d_k^T y_k} - t_{k+1}^{\mathrm{MSMDL}} \frac{g_{k+1}^T s_k}{d_k^T y_k} \right) g_{k+1}^T d_k \\ &= -\left\| g_{k+1} \right\|^2 + \frac{g_{k+1}^T y_k}{d_k^T y_k} g_{k+1}^T d_k - t_{k+1}^{\mathrm{MSMDL}} \frac{g_{k+1}^T s_k}{d_k^T y_k} g_{k+1}^T d_k \\ &= -\left\| g_{k+1} \right\|^2 + \frac{g_{k+1}^T y_k}{d_k^T y_k} g_{k+1}^T d_k - t_{k+1}^{\mathrm{MSMDL}} \frac{\alpha_k \left(g_{k+1}^T d_k \right)^2}{d_k^T y_k}. \end{aligned} \tag{41}$$

Now, from the equality (41), it follows

$$g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2 + \frac{g_{k+1}^T y_k}{d_k^T y_k} g_{k+1}^T d_k - \frac{t_{k+1}^{\mathrm{MSMDL}} \alpha_k \left(g_{k+1}^T d_k\right)^2}{d_k^T y_k}$$
$$= \frac{-\|g_{k+1}\|^2 \left(d_k^T y_k\right)^2 + \left(g_{k+1}^T y_k\right)\left(d_k^T y_k\right)\left(g_{k+1}^T d_k\right)}{\left(d_k^T y_k\right)^2} - \frac{t_{k+1}^{\mathrm{MSMDL}} \alpha_k \left(g_{k+1}^T d_k\right)^2 \left(d_k^T y_k\right)}{\left(d_k^T y_k\right)^2}. \tag{42}$$

Use the inequality

$$u^T v \le \frac{1}{2}\left(\|u\|^2 + \|v\|^2\right), \tag{43}$$

with

$$u = \frac{1}{\sqrt{2\theta}}\left(d_k^T y_k\right) g_{k+1},$$
$$v = \sqrt{2\theta}\left(g_{k+1}^T d_k\right) y_k. \tag{44}$$

We get

$$g_{k+1}^T d_{k+1} \le \frac{\|g_{k+1}\|^2 \left(d_k^T y_k\right)^2 + 1/2\left(\left\|1/\sqrt{2\theta}\left(d_k^T y_k\right) g_{k+1}\right\|^2 + \left\|\sqrt{2\theta}\left(g_{k+1}^T d_k\right) y_k\right\|^2\right)}{\left(d_k^T y_k\right)^2} - \frac{t_{k+1}^{\mathrm{MSMDL}} \alpha_k \left(g_{k+1}^T d_k\right)^2 \left(d_k^T y_k\right)}{\left(d_k^T y_k\right)^2}$$
$$= \frac{-\|g_{k+1}\|^2 \left(d_k^T y_k\right)^2 + 1/2\left(1/2\theta \left(d_k^T y_k\right)^2 \|g_{k+1}\|^2 + 2\theta \left(g_{k+1}^T d_k\right)^2 \|y_k\|^2\right)}{\left(d_k^T y_k\right)^2} - \frac{t_{k+1}^{\mathrm{MSMDL}} \alpha_k \left(g_{k+1}^T d_k\right)^2}{d_k^T y_k}$$
$$= \frac{-\|g_{k+1}\|^2 \left(d_k^T y_k\right)^2 + 1/4\theta \left(d_k^T y_k\right)^2 \|g_{k+1}\|^2 + \theta \left(g_{k+1}^T d_k\right)^2 \|y_k\|^2}{\left(d_k^T y_k\right)^2} - t_{k+1}^{\mathrm{MSMDL}} \frac{\alpha_k \left(g_{k+1}^T d_k\right)^2}{d_k^T y_k}$$
$$\le -\|g_{k+1}\|^2 + \frac{1}{4\theta}\|g_{k+1}\|^2 + \frac{\theta \left(g_{k+1}^T d_k\right)^2 \|y_k\|^2}{\left(d_k^T y_k\right)^2} - \theta \frac{\|y_k\|^2}{s_k^T y_k} \frac{\alpha_k \left(g_{k+1}^T d_k\right)^2}{d_k^T y_k}$$
$$= -\|g_{k+1}\|^2 + \frac{1}{4\theta}\|g_{k+1}\|^2 + \frac{\theta \left(g_{k+1}^T d_k\right)^2 \|y_k\|^2}{\left(d_k^T y_k\right)^2} - \theta \frac{\|y_k\|^2}{\alpha_k d_k^T y_k} \frac{\alpha_k \left(g_{k+1}^T d_k\right)^2}{d_k^T y_k}$$
$$= -\|g_{k+1}\|^2 + \frac{1}{4\theta}\|g_{k+1}\|^2$$
$$= -\left(1 - \frac{1}{4\theta}\right)\|g_{k+1}\|^2. \tag{45}$$

Because $\theta > (1/4)$, the inequality (40) is fulfilled for $c = 1 - (1/4\theta)$ in (45), and arbitrary $k \ge 0$.

Global convergence of the MSMDL iterations is verified in Theorem 1. Assumption 2 and the proofs of Lemma 3 can be found in [36–38]. □

Assumption 2. The function $f: \mathbb{R}^n \longrightarrow \mathbb{R}$ is twice continuously differentiable and uniformly convex on $\mathbb{R}^n$.

If the conditions in Assumption 2 hold, then Assumption 1 is satisfied.

Lemma 3 (see [36–38]). *Under the conditions in Assumption 2, there exist real numbers m, M satisfying*

$$0 < m \le 1 \le M. \tag{46}$$

It is such that $f(x_k)$ has a unique minimizer x^* and

$$m\|y\|^2 \le y^T \nabla^2 f(x_k) y \le M\|y\|^2, \quad \forall x_k,\ y \in \mathbb{R}^n, \tag{47}$$

$$\frac{1}{2} m \|x_k - x^*\|^2 \le f(x_k) - f(x^*) \le \frac{1}{2} M \|x_k - x^*\|^2, \quad \forall x_k \in \mathbb{R}^n, \tag{48}$$

$$m\|x_k - x^*\| \le \|g_k\| \le M\|x_k - x^*\|, \quad \forall x_k \in \mathbb{R}^n. \tag{49}$$

Theorem 1. *Let restrictions in Assumption 1 hold. If f is uniformly convex, then the series $\{x_k\}$ generated inside the MSMDL iterations satisfies*

$$\liminf_{k\longrightarrow\infty}\|g_k\| = 0. \tag{50}$$

Proof. We suppose the opposite, i.e., that (50) is not true. This initiates the existence of a positive constant $\complement > 0$ such that for all k,

$$\|g_k\| \geq \complement. \tag{51}$$

Then, from (32) and (35), we obtain

$$\begin{aligned}
\|d_k\| &= \|-g_k + \beta_k^{\mathrm{MSMDL}} d_{k-1}\| \\
&\leq \|g_k\| + \left|\beta_k^{\mathrm{MSMDL}}\right|\|d_{k-1}\| \\
&= \|g_k\| + \left|\frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} - t_k^{\mathrm{MSMDL}}\frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}}\right|\|d_{k-1}\| \\
&= \|g_k\| + \left|\frac{g_k^T y_{k-1}}{\alpha_{k-1} d_{k-1}^T y_{k-1}} - t_k^{\mathrm{MSMDL}}\frac{g_k^T s_{k-1}}{\alpha_{k-1} d_{k-1}^T y_{k-1}}\right|\|\alpha_{k-1} d_{k-1}\| \\
&= \|g_k\| + \left|\frac{g_k^T y_{k-1}}{s_{k-1}^T y_{k-1}} - t_k^{\mathrm{MSMDL}}\frac{g_k^T s_{k-1}}{s_{k-1}^T y_{k-1}}\right|\|s_{k-1}\| \\
&\leq \|g_k\| + \left(\frac{\|g_k\|\|y_{k-1}\|}{s_{k-1}^T y_{k-1}} + \left|t_k^{\mathrm{MSMDL}}\right|\frac{\|g_k\|\|s_{k-1}\|}{s_{k-1}^T y_{k-1}}\right)\|s_{k-1}\| \\
&= \|g_k\| + \frac{\|g_k\|\left(\|y_{k-1}\| + \left|t_k^{\mathrm{MSMDL}}\right|\|s_{k-1}\|\right)}{s_{k-1}^T y_{k-1}}\|s_{k-1}\|.
\end{aligned} \tag{52}$$

Further calculations and approximations on the basis of (32) are given as follows:

$$\begin{aligned}
\|d_k\| &\leq \|g_k\| + \frac{\|g_k\|\left(L\|s_{k-1}\| + \left|t_k^{\mathrm{MSMDL}}\right|\|s_{k-1}\|\right)}{s_{k-1}^{\mathrm{T}} y_{k-1}}\|s_{k-1}\| \\
&= \|g_k\| + \frac{\|g_k\|\left(L + \left|t_k^{\mathrm{MSMDL}}\right|\right)\|s_{k-1}\|}{s_{k-1}^T y_{k-1}}\|s_{k-1}\| \\
&\leq \|g_k\| + \frac{\|g_k\|\left(L + \left|t_k^{\mathrm{MSMDL}}\right|\right)\|s_{k-1}\|}{\eta\|s_{k-1}\|^2}\|s_{k-1}\| \\
&= \left(1 + \frac{L + \left|t_k^{\mathrm{MSMDL}}\right|}{\eta}\right)\|g_k\|.
\end{aligned} \tag{53}$$

To complete the theorem, it is necessary to prove that t_k^{MSMDL} is bounded. Two cases should be distinguished based on the definition of t_k^{MSMDL} in (28). □

Case 1. If $t_k^{\mathrm{MSMDL}} = \theta\|y_{k-1}\|^2/s_{k-1}^T y_{k-1}$, one concludes

$$\left|t_k^{\mathrm{MSMDL}}\right| = \left|\theta\frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}}\right| \leq \theta\frac{L^2\|s_{k-1}\|^2}{\eta\|s_{k-1}\|^2} = \frac{\theta L^2}{\eta}. \tag{54}$$

Case 2. In the case $t_k^{\mathrm{MSMDL}} = \tau_k$, it follows

$$\begin{aligned}
\left|t_k^{\mathrm{MSMDL}}\right| &= \frac{\left|\left(\lrcorner_k\left(\gamma_k^{\mathrm{MSM}}\right)^{-1} - 1\right)\|g_k\|^2 s_{k-1}^T y_{k-1} + g_k^T y_{k-1} g_k^T s_{k-1}\right|}{\left(g_k^T s_{k-1}\right)^2} \\
&\leq \frac{\left|\lrcorner_k\left(\gamma_k^{\mathrm{MSM}}\right)^{-1} - 1\right|\|g_k\|^2 L\|s_{k-1}\|^2 + \|g_k\|\|y_{k-1}\|\|g_k\|\|s_{k-1}\|}{\|g_k\|^2\|s_{k-1}\|^2} \\
&\leq \frac{\left|\lrcorner_k\left(\gamma_k^{\mathrm{MSM}}\right)^{-1} - 1\right|\|g_k\|^2 L\|s_{k-1}\|^2 + \|g_k\|^2 L\|s_{k-1}\|^2}{\|g_k\|^2\|s_{k-1}\|^2} \\
&= L\left|\lrcorner_k\left(\gamma_k^{\mathrm{MSM}}\right)^{-1} - 1\right| + L.
\end{aligned} \tag{55}$$

Based on Algorithm 1 and the initial value for $\alpha_k = 1$, it follows that $\alpha_k \in (0, 1]$. Now, the fact $\alpha_k \in (0, 1]$ implies $1 \leq \lrcorner_k \leq (5/4)$ (Figure 1), which further in common with (55) gives as follows:

$$\begin{aligned}\left|t_k^{\mathrm{MSMDL}}\right| &\le L\left|\lambda_k\left(\gamma_k^{\mathrm{MSM}}\right)^{-1}-1\right|+L\\ &\le L\left(\lambda_k\left(\gamma_k^{\mathrm{MSM}}\right)^{-1}+1\right)+L\\ &\le L\left(2+\frac{5}{4}\left(\gamma_k^{\mathrm{MSM}}\right)^{-1}\right).\end{aligned} \tag{56}$$

Since γ_k is an approximation of the Hessian $\nabla^2 f(x_k)$, the inequality (47) implies $m \le \gamma_k \le M$. Now, from (56), we conclude

$$\left|t_k^{\mathrm{MSMDL}}\right| \le L\left(2+\frac{5}{4m}\right). \tag{57}$$

Based on the cases 1 and 2, it follows

$$\left|t_k^{\mathrm{MSMDL}}\right| \le \max\left\{\frac{\theta L^2}{\eta}, L\left(2+\frac{5}{4m}\right)\right\} = T. \tag{58}$$

Now, on the basis of (53) and (58), it follows

$$\|d_k\| \le \left\{1+\frac{L+T}{\eta}\right\}\|g_k\| = \frac{\eta+L+T}{\eta}\|g_k\|. \tag{59}$$

Squaring both sides in (59) implies

$$\|d_k\|^2 \le \frac{(\eta+L+T)^2}{\eta^2}\|g_k\|^2. \tag{60}$$

Next, dividing both sides of inequalities (60) by $\|g_k\|^4$ and using (61), it can be concluded that

$$\frac{\|d_k\|^2}{\|g_k\|^4} \le \frac{(\eta+L+T)^2}{\eta^2}\cdot\frac{1}{\complement^2} \Leftrightarrow \frac{\|g_k\|^4}{\|d_k\|^2} \ge \frac{\eta^2\cdot\complement^2}{(\eta+L+T)^2}. \tag{61}$$

The inequalities in (61) imply

$$\sum_{k=0}^{\infty}\frac{\|g_k\|^4}{\|d_k\|^2} \ge \sum_{k=0}^{\infty}\frac{\eta^2\cdot\complement^2}{(\eta+L+T)^2} = \infty. \tag{62}$$

Therefore, $\|g_k\| \ge \complement$ causes a contradiction with Lemma 1.

Theorem 2. *Let the restrictions in Assumption 1 hold. If the goal function $f(x)$ is uniformly convex, then the series $\{x_k\}$ generated within the BB1DL iterations satisfies (60).*

Proof. Proof Theorem 2 is similar to proof Theorem 1. It differs in the part where it is necessary to prove that t_k^{BB1DL} is bounded. In the sequel, we prove that t_k^{BB1DL} is bounded. Two cases should be distinguished based on the definition of t_k^{BB1DL} in (30).

Case (i): If $t_k^{\mathrm{BB1DL}} = (\theta\|y_{k-1}\|^2/s_{k-1}^T y_{k-1})$, based on (63), we have

$$\left|t_k^{\mathrm{BB1DL}}\right| \le \frac{\theta L^2}{\eta}. \tag{63}$$

Case (ii): In the case $t_k^{\mathrm{BB1DL}} = \tau_k^{BB1}$, it follows

$$\begin{aligned}\left|t_k^{\mathrm{BB1DL}}\right| &= \frac{\left|\left(\left(\gamma_k^{\mathrm{BB1}}\right)^{-1}-1\right)\|g_k\|^2 s_{k-1}^T y_{k-1}+g_k^T y_{k-1} g_k^T s_{k-1}\right|}{\left(g_k^T s_{k-1}\right)^2}\\ &\le \frac{\left|\left(\gamma_k^{\mathrm{BB1}}\right)^{-1}-1\right|\|g_k\|^2 L\|s_{k-1}\|^2+\|g_k\|\|y_{k-1}\|\|g_k\|\|s_{k-1}\|}{\|g_k\|^2\|s_{k-1}\|^2}\\ &\le \frac{\left|\left(\gamma_k^{\mathrm{BB1}}\right)^{-1}-1\right|\|g_k\|^2 L\|s_{k-1}\|^2+\|g_k\|^2 L\|s_{k-1}\|^2}{\|g_k\|^2\|s_{k-1}\|^2}\\ &= L\left|\left(\gamma_k^{\mathrm{BB1}}\right)^{-1}-1\right|+L.\end{aligned} \tag{64}$$

Now, from (14) and (37), it follows $\gamma_k^{\mathrm{BB1}} > 0$, i.e.,

$$\begin{aligned}\left|t_k^{\mathrm{BB1DL}}\right| &\le L\left|\left(\gamma_k^{\mathrm{BB1}}\right)^{-1}-1\right|+L\\ &\le L\left(\left(\gamma_k^{\mathrm{BB1}}\right)^{-1}+1\right)+L\\ &= L\left(\frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}\right)^{-1}+2L\\ &= L\frac{\left\|y_{k-1}\right\|^2}{s_{k-1}^T y_{k-1}}+2L\\ &\le L\frac{L^2\left\|s_{k-1}\right\|^2}{\eta\left\|s_{k-1}\right\|^2}+2L\\ &\le L\left(\frac{L^2}{\eta}+2\right).\end{aligned} \tag{65}$$

The penultimate inequality in (65) directly follows from (32) and (35). Based on the cases (i) and (ii), it follows

$$\left|t_k^{\mathrm{BB1DL}}\right| \le \max\left\{\frac{\theta L^2}{\eta}, L\left(\frac{L^2}{\eta}+2\right)\right\} = T. \tag{66}$$

The proof is completed.

An additional limit-type convergence result is proved in Theorem 3. Theorem 3 shows the linear convergence of the MSMDL method under Assumption 2. In order to prove the linear convergence of the MSMDL method, we present Lemma 4 which gives a lower bound of the step length α_k. The proof is similar as in the case of Lemma 4 in Danmalam et al. in [39] or Lemma 4 in [40]. □

Lemma 4. *We suppose that the conditions in Assumption 2 hold, and the sequence $\{x_k\}$ be generated by the MSMDL method with the backtracking line search. Then, there is a constant $\lambda > 0$ such that*

$$\alpha_k \ge \lambda, \forall k > 0. \tag{67}$$

Proof. The backtracking line search condition gives as follows:

$$f\left(x_k+\alpha_k d_k\right) \le f\left(x_k\right)+\sigma\alpha_k g_k^T d_k. \tag{68}$$

If $\alpha_k \ne 1$, then $\rho^{-1}\alpha_k$ does not satisfy (68), that is,

$$f\left(x_k+\rho^{-1}\alpha_k d_k\right)-f\left(x_k\right) > \sigma\rho^{-1}\alpha_k g_k^T d_k. \tag{69}$$

The mean value theorem and (32) ensure the existence of $\xi_k \in [0,1]$, such that

$$\begin{aligned}f\left(x_k+\rho^{-1}\alpha_k d_k\right)-f\left(x_k\right) &= \rho^{-1}\alpha_k\left(g\left(x_k+\xi_k\rho^{-1}\alpha_k d_k\right)\right)^T d_k\\ &= \rho^{-1}\alpha_k g_k^T d_k+\rho^{-1}\alpha_k\left(g\left(x_k+\xi_k\rho^{-1}\alpha_k d_k\right)-g_k\right)^T d_k\\ &\le \rho^{-1}\alpha_k g_k^T d_k+L\rho^{-2}\alpha_k^2\left\|d_k\right\|^2.\end{aligned} \tag{70}$$

From (60), we obtain

$$\frac{\left\|g_k\right\|^2}{\left\|d_k\right\|^2} \ge \frac{\eta^2}{(\eta+L+T)^2} = \lambda_1. \tag{71}$$

Now, the following inequalities hold on the basis of (69) and (70):

$$\begin{aligned}\sigma\rho^{-1}\alpha_k g_k^T d_k < f\left(x_k+\rho^{-1}\alpha_k d_k\right)-f\left(x_k\right) &\le \rho^{-1}\alpha_k g_k^T d_k+L\rho^{-2}\alpha_k^2\left\|d_k\right\|^2\\ \sigma\rho^{-1}\alpha_k g_k^T d_k-\rho^{-1}\alpha_k g_k^T d_k &\le L\rho^{-2}\alpha_k^2\left\|d_k\right\|^2\\ (\sigma-1)\rho^{-1}\alpha_k g_k^T d_k &\le L\rho^{-2}\alpha_k^2\left\|d_k\right\|^2\\ (\sigma-1)g_k^T d_k &\le L\rho^{-1}\alpha_k\left\|d_k\right\|^2\\ -(1-\sigma)g_k^T d_k &\le L\rho^{-1}\alpha_k\left\|d_k\right\|^2.\end{aligned} \tag{72}$$

The inequalities (72), (40), and (71) give

$$(1-\sigma)c\left\|g_k\right\|^2 \le L\rho^{-1}\alpha_k\left\|d_k\right\|^2, \tag{73}$$

which leads to

$$\alpha_k \ge \frac{\rho(1-\sigma)c}{L}\frac{\left\|g_k\right\|^2}{\left\|d_k\right\|^2} \ge \frac{\rho(1-\sigma)c}{L}\lambda_1. \tag{74}$$

The last inequality gives the required inequality (67) by setting $\lambda = (\rho(1-\sigma)c/L)\lambda_1$. The proof is completed. □

Theorem 3. *Let Assumption 2 hold and x^* be the unique minimizer of (1). Then, there are constants $p > 0$ and $r \in (0,1)$ such that the sequence $\{x_k\}$ generated by the MSMDL method fulfills*

$$\|x_k - x^*\| \leq pr^k. \tag{75}$$

Proof. From the backtracking line search (40), Lemma 2, and Lemma 4, it follows

$$\begin{aligned} f(x_{k+1}) - f(x^*) &\leq f(x_k) - f(x^*) + \sigma\alpha_k g_k^T d_k \\ &\leq f(x_k) - f(x^*) - \sigma\alpha_k c\|g_k\|^2 \\ &\leq f(x_k) - f(x^*) - \sigma\lambda c\|g_k\|^2. \end{aligned} \tag{76}$$

Using the left inequality in (49) and afterwards the right inequality in (48), the inequality in (76) is further approximated as follows:

$$\begin{aligned} f(x_{k+1}) - f(x^*) &\leq f(x_k) - f(x^*) - \sigma\lambda c m^2\|x_k - x^*\|^2 \\ &\leq f(x_k) - f(x^*) - 2\sigma\lambda c\frac{m^2}{M}(f(x_k) \\ &\quad - f(x^*)) \\ &= \left(1 - 2\sigma\lambda c\frac{m^2}{M}\right)(f(x_k) - f(x^*)). \end{aligned} \tag{77}$$

We consider the replacement $r = 1 - 2\sigma\lambda c(m^2/M)$ in the inequality (77). Clearly, on the basis of $0 < \sigma < 0.5$, $\lambda > 0$, $(m^2/M) > 0$, and $c = 1 - (1/4\theta) > 0$, $\theta > (1/4)$, it follows $r < 1$. On the other hand, $\sigma < 0.5$, $\lambda \leq \alpha_k \leq 1$, $(m^2/M) < 1$, and $c < 1$ imply $r > 1 - \lambda c > 0$.

Furthermore, it follows

$$\begin{aligned} f(x_{k+1}) - f(x^*) &\leq r(f(x_k) - f(x^*)) \leq \cdots \\ &\leq r^{k+1}(f(x_0) - f(x^*)). \end{aligned} \tag{78}$$

Combining the left inequality of (48) with (78), we obtain

$$\begin{aligned} \|x_k - x^*\| &\leq \frac{2}{m}(f(x_k) - f(x^*)) \\ &\leq \frac{2}{m}(f(x_0) - f(x^*))r^k \leq pr^k, \end{aligned} \tag{79}$$

which shows that the inequality (75) holds for $p = (2/m)(f(x_0) - f(x^*)) > 0$. The proof is completed. □

Remark 1. The result as in Theorem 3 can be directly applied to the *BB1DL* method.

4. Numerical Experiments

In this section, we are going to prove the numerical efficiency of the MSMDL and BB1DL methods. To this aim, we perform two competitions on standard test functions with given initial points from [41, 42]. The first competition is between CG-DESCENT [24], M1 [27], DK [26], and MSMDL methods, and the second one is between BB1DL, MSMDL, and two recently developed DL CG methods with global convergence property (EDL [28] and MDL [15]). We compare all of these methods into three criteria:

(i) The CPU time in seconds, CPUts

(ii) The number of iterative steps, NI

(iii) The number of function evaluations, NFE

The methods which participate in the competition are presented in Section 1 (Table 1). Test problems are evaluated in ten dimensions (100, 500, 1000, 3000, 5000, 7000, 8000, 10000, 15000, and 20000). Codes implementing the tested methods are evaluated in MATLAB R2017a and on a LAP's (Intel (R) Core (TM) i3-7020U, up to 2.3 GHz, and 8 GB memory) with the Windows 10 Pro operating system.

Algorithms MSMDL, BB1DL, CG-DESCENT, M1, DK, EDL, and MDL were compared using the backtracking line search with parameters $\sigma = 0.0001$, $\beta = 0.8$. Tested algorithms are stopped after 50000 iterations or

$$\begin{aligned} \|g_k\| &\leq \varepsilon = 10^{-6}, \\ \frac{|f(x_{k+1}) - f(x_k)|}{1 + |f(x_k)|} &\leq \delta = 10^{-16}. \end{aligned} \tag{80}$$

Specific parameters used only in the MDL and MSMDL methods are defined as follows:

(i) In the MDL method, $\theta = 0.26$, $C = 1$, and $r = r_k = \theta\|g_{k-1}\|$

(ii) In the MSMDL method, $\theta = 0.26$

The symbol "∗" in the subsequent tables means that the method failed to achieve the prescribed accuracy after 50000 iterations for one or more tested dimensions of the observed test function.

Summary numerical results for the first competition (between MSMDL, CG-DESCENT, M1, and DK methods) are obtained by testing 34 test functions and presented in Table 2. This table includes numerical results obtained by monitoring the criteria NI, NFE, and CPUts in the MSMDL, CG-DESCENT, M1, and DK methods.

The performance profiles proposed in [43] are applied to compare obtained numerical data for criteria CPUts, NI, and NFE generated by the tested methods listed at the beginning of the section. The left-hand side of each performance profile in Figures 2–5 indicates the percentage of test problems in which the considered method is the best among tested methods, whereas the right-hand side gives the percentage of the test problems that are successfully solved by each method.

Benchmark comparison ranges the solvers included in the set $\mathcal{S}$ on the set of test problems $\mathcal{P}$. The performance profile ratio $r_{p,s}$ is defined for each problem $p \in \mathcal{P}$ and each solver $s \in \mathcal{S}$, and it is formulated as follows:

Table 2: Summary test results of MSMDL, CG-DESCENT, M1, and DK methods for the NI, NFE, and CPUts.

Test function	CG-DESCENT NI/NFE/CPUts	MSMDL NI/NFE/CPUts	M1 NI/NFE/CPUts	DK NI/NFE/CPUts
Extended penalty function	2205/72128/21.656	1575/49766/18.766	2264/73854/23.219	1275/47664/18.375
Perturbed quadratic function	*/*/*	*/*/*	*/*/*	*/*/*
Raydan 1 function	126771/3900575/1572.109	30614/837631/408.953	119648/3667586/1457.172	53905/1508669/649.625
Raydan 2 function	441/1162/1.219	57/124/0.313	441/1162/1.313	57/124/0.25
Diagonal 1 function	85038/3530672/3958.094	21812/822865/1005	79024/3271384/3613.234	31406/1212581/1472.969
Diagonal 2 function	*/*/*	*/*/*	*/*/*	*/*/*
Diagonal 3 function	109803/4480102/6578.172	34503/1294525/2266.453	106160/4317169/6554.422	47699/1816310/2718.922
Hager function	6093/128794/299.625	1430/22994/52.063	6074/127372/284.891	3087/56298/117.359
Generalized tridiagonal 1 function	1525/29126/30.094	1124/17065/19.25	1499/28511/29.813	604/9965/11.141
Extended tridiagonal 1 function	*/*/*	1053/3292/12.797	*/*/*	*/*/*
Extended TET function	5818/75694/55.813	1090/7769/7.25	5759/74907/55.203	2014/20160/17.406
Diagonal 4 function	23674/525926/79.516	2058/34012/5.906	21963/483613/73.734	7986/153701/25.313
Diagonal 5 function	25121/50463/105.547	40/90/0.438	25120/50460/104.656	40/90/0.297
Extended Himmelblau function	8028/192482/33.438	1220/21461/4.578	7837/187928/31.531	1350/25610/5.453
Extended quadratic penalty QP1 function	595/13170/5.063	894/13296/5.406	598/13631/4.969	549/10835/4.313
Extended quadratic exponential EP1 function	606/18463/7.188	499/12243/4.938	606/18463/7.063	496/13643/5.203
ARWHEAD function (CUTE)	82994/3970405/2213.172	18117/774822/474.234	69143/3280771/1753.766	38236/1728522/1018.219
Almost perturbed quadratic function	*/*/*	*/*/*	*/*/*	*/*/*
ENGVAL1 function (CUTE)	1436/23198/8.781	1008/13576/6.484	1423/22977/8.75	569/8306/3.656
QUARTC function (CUTE)	262291/524662/4001.109	*/*/*	262291/524662/3917.75	*/*/*
Diagonal 6 function	513/1393/1.578	57/124/0.281	509/1555/1.563	57/124/0.297
Generalized quartic function	2517/41813/17.469	832/6769/2.984	1470/21907/10.641	1473/19908/8.891
Diagonal 7 function	655/6838/5.922	189/1750/2.594	655/6838/5.844	538/3321/3.766
Diagonal 8 function	698/6926/6.063	270/2318/2.844	698/6926/6.453	579/3837/4.016
Full Hessian FH3 function	5350/212795/54.609	2249/73823/25.156	5361/213236/55.313	2494/88852/25.203
Diagonal 9 function	*/*/*	66739/2712392/1231.156	*/*/*	138771/5914443/1726.516
Extended Rosenbrock	50/110/0.313	50/110/0.328	50/110/0.344	50/110/0.328
Extended BD1 function (block diagonal)	1101/15879/7.703	1175/10059/5.188	1101/15870/6.641	618/7286/3.406
Extended Maratos function	93270/2927642/440.75	9716/250596/40.563	88036/2749653/422.266	30626/866995/135.375
NONDQUAR function (CUTE)	24/2324/9.516	31/2721/10.156	40/2708/10.594	34/2503/9.984
DQDRTIC function (CUTE)	38363/1087562/306.25	7082/170118/49.594	36248/1022842/291.016	12486/316078/96.109
Extended Freudenstein and Roth function	291520/11172996/2176.25	31821/1040192/213.016	277174/10582041/2018.797	92706/3270158/656.016
Extended Beale function	31457/590812/1474.203	1880/24895/65.016	29002/538469/1354.422	10725/169640/427.641
EDENSCH function (CUTE)	984/15904/54.594	1050/13383/55.078	990/15918/56.281	719/10307/31.75

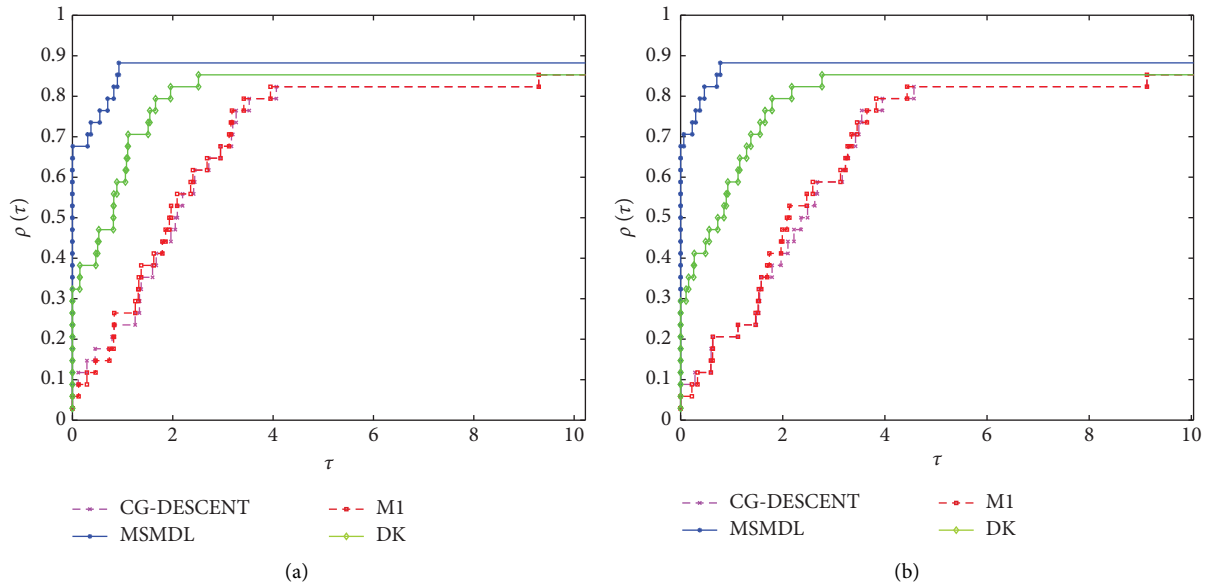

Figure 2: (a) NI and (b) NFE performance profiles for MSMDL, M1, CG-DESCENT, and DK.

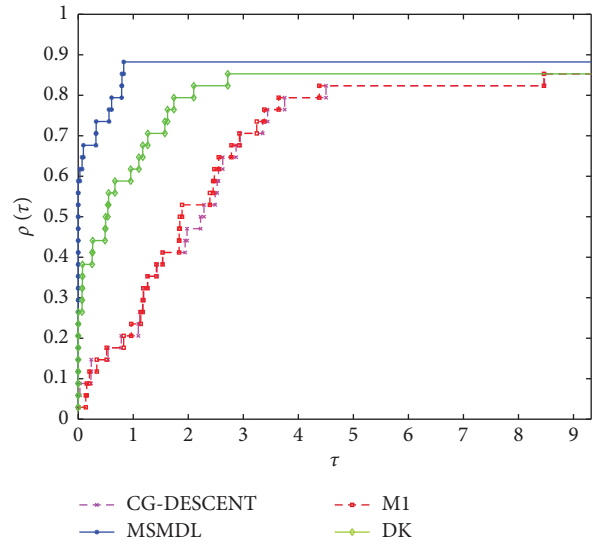

Figure 3: CPUts performance profiles for MSMDL, CG-DESCENT, M1, and DK.

$$r_{p,s} = \frac{x_{p,s}}{\min\{x_{p,s}:\ p \in \mathcal{P} \text{ and s} \in \mathcal{S}\}}, \tag{81}$$

where $x_{p,s}$ denotes the NI or NFE or CPUts needed to solve the problem p by the solver s. Then, the performance profile for a solver s in the log 2-scale is defined by the following:

$$\rho_s(\tau) = \frac{1}{n_p} \text{size}\{p \in \mathcal{P}:\ \log_2 r_{p,s} \leq \tau\}. \tag{82}$$

Solvers with a greater probability $\rho_s(\tau)$ are more desirable. If the solver s_1 achieves better results compared to the solver s_2, then the curve $\rho_{s_1}(\tau)$ of the performance profile generated by the solver s_1 is located above the corresponding curve $\rho_{s_2}(\tau)$ of the performance profile generated by the solver s_2.

In Figure 2, we compare the performance profiles NI and NFE for CG-DESCENT, MSMDL, M1, and DK methods based on the numerical values covered in Table 2. A careful analysis reveals that the MSMDL method solves 64.71% of the test problems with the least NI compared to the CG-DESCENT (8.82%), M1 (5.88%), and DK (32.35%). From Figure 2(a), it is perceptible that the MSMDL graphs attain the top level first, which indicates that MSMDL outperforms other considered methods with respect to the criterion NI. Figure 2(b) shows that the MSMDL method is more efficient than the CG-DESCENT, MSMDL, M1, and DK methods, with respect to NFE, since it solves 67.65% of the test

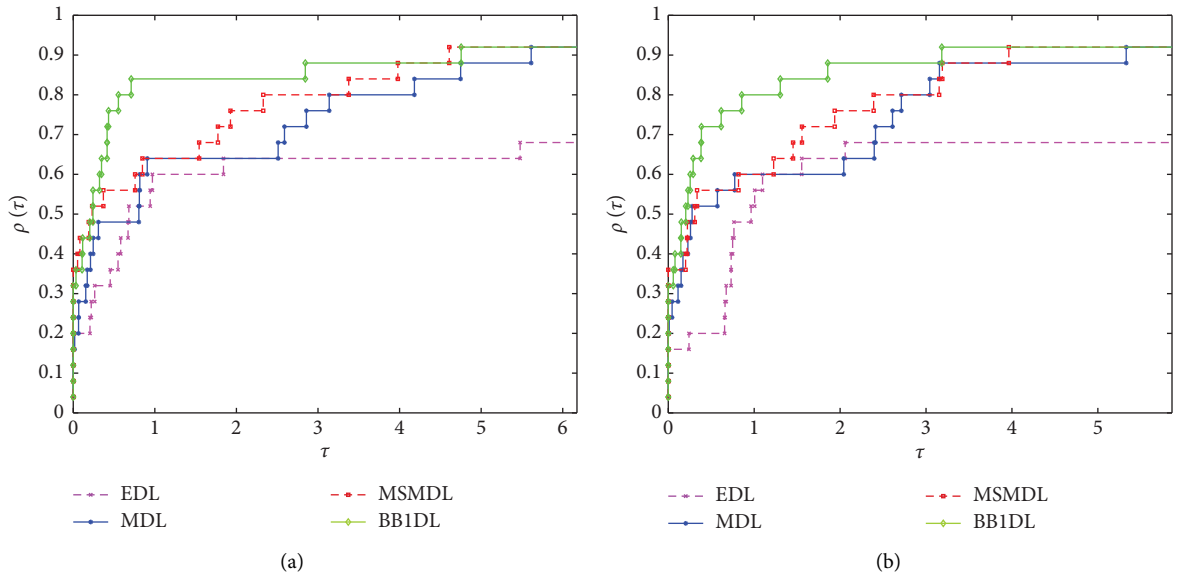

Figure 4: (a) NI and (b) NFE performance profiles for EDL, MDL, MSMDL, and BB1DL.

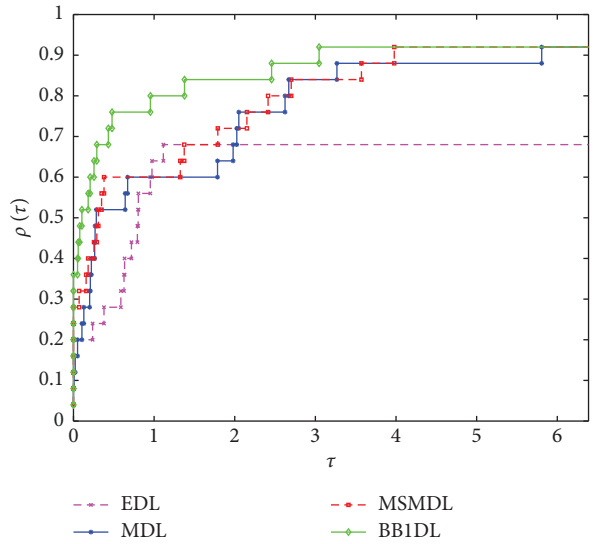

Figure 5: CPUts performance profiles for EDL, MDL, MSMDL, and BB1DL.

problems with the least NFE compared to the CG-DESCENT (8.82%), M1 (5.88%), and DK (29.41%). From Figure 2 (bottom), it is notified that the MSMDL graph first reaches the top, so MSMDL is the winner relative to other considered methods.

Figure 3 shows the performance profile of the CG-DESCENT, MSMDL, M1, and DK methods based on the CPUts included in Table 2. The MSMDL method solves 58.82% of the test problems with the least CPUts compared to CG-DESCENT (5.88%), M1 (2.94%), and DK (23.53%). According to Figure 3, the MSMDL graph comes first to the top, which verifies its dominance in terms of CPUts.

The MSMDL method did not successfully solve 4 (11.76%) of all the test functions in Table 2, while each of the CG-DESCENT, M1, and DK methods did not successfully solve 5 (14.71%) of the test functions. A detailed summary of the results for each method is arranged in Table 3.

With the total of 330 solved test problems, MSMDL is able to solve the largest number of test problems (97.06% of all tested problems), while M1 and CG-DESCENT solved only 92.35% of all tested problems.

Based on the data involved in Tables 2 and 3 and graphs involved in Figures 2 and 3, it is noticed that the MSMDL method achieves the best results compared to CG-DESCENT, M1, and DK methods with respect to three basic criteria: NI, NFE, and CPUts.

In addition to standard analysis of numerical results, we performed additional analysis for the MSMDL method. The goal of additional tests is to monitor the usage of the

Table 3: Statistics of not successfully solved test functions by each method.

Method	Test functions $\sum = 34$	Percentage (%)	Test problems $\sum = 340$	Percentage (%)
CG-DESCENT	5	14.71	26	7.65
MSMDL	4	11.76	10	2.94
M1	5	14.71	26	7.65
DK	5	14.71	22	6.47

imposed value $\gamma_k = 1$ in iterations for each of the tested functions. We also count the assignments $t_k^{\text{MSMDL}} = \tau_k$ in (28) for each individual function. The third analysed parameter is the maximum value for $(g_k^T d_k / \|g_k\|^2)$ obtained during testing. The test results are given in Table 4.

The total number of assigned values $\gamma_k = 1$ (respectively. $t_k^{\text{MSMDL}} = \tau_k$) from Table 4 will be denoted by $\prec_{\gamma_k=1}$ (respectively. $\prec_{t_k=\tau_k}$). Furthermore, the values $\mu = \max\{(g_k^{\mathrm{T}} d_k / \|g_k\|^2)\}$ will be monitored. The total sum of individual values NI, $\prec_{\gamma_k=1}$, and $\prec_{t_k=\tau_k}$ across the tested functions will be denoted by $\sum \text{NI}$, $\sum \prec_{\gamma_k=1}$, and $\sum \prec_{t_k=\tau_k}$, respectively. The total sum of all iterative steps across all test functions is equal to $\sum \text{NI} = 240235$, which further implies $\sum \prec_{\gamma_k=1} = 117966 = 0.491044 \sum NI$ and $\sum \prec_{t_k=\tau_k} = 118352 = 0.492651 \sum NI$. In this way, the behavior $\sum \prec_{\gamma_k=1} \approx \sum \prec_{t_k=\tau_k} \approx 0.5 \sum \text{NI}$ is observable.

In the subsequent numerical experiments, we compare the MSMDL and BB1DL methods versus EDL and MDL methods.

The performance comparisons of the MSMDL and BB1DL solvers against the EDL and MDL methods are shown in Figures 4 and 5. Figure 4(a) compares considered solvers with respect to the profile NI, Figure 4(b) in terms of the NFE. Graphs of CPUts performance profiles for EDL, MDL, MSMDL, and BB1DL are arranged in Figure 5.

Figure 4(b) shows that the MSMDL and BB1DL methods achieved more efficient results than EDL and MDL methods in terms of NFE, which is confirmed by upper positions of the graphs of their performance profiles. Figure 4(a) shows that the MSMDL and BB1DL methods achieved slightly superior results compared to the EDL and MDL methods in terms of NI, which is confirmed by the dominant graphs of their performance profiles.

Summary numerical results for the second competition (between EDL, MDL, MSMDL, and BB1DL) are obtained by testing 25 test functions and arranged in Table 5. This table shows numerical data obtained by monitoring the criteria NI, NFE, and CPUts for the EDL, MDL, MSMDL, and BB1DL methods.

Numerical results in Table 5 show that the MSMDL method solves about 36%, while the BB1DL method successfully solves 32% of the test problems with the least values of NI and NFE.

Profile performances based on CPUts of the MSMDL and BB1DL in Figure 5 show better performances of these solvers compared to the profile performances of the EDL and MDL solvers. In numerical results in Table 5, we found that the MSMDL method solves about 28%, while the BB1DL method successfully solves 36% of the test problems with the minimal CPUts.

We observe that the EDL and MDL methods, which are currently among the best DL conjugate gradient methods proposed in the literature, give worse numerical results than the MSMDL and BB1DL methods in terms of the NI, NFE, and CPUts.

5. Application in 2D Robotic Motion Control

Problems arising from the concept of robot system have attracted the attention of researchers and subsequently, some algorithms for handling them have been developed [6, 44]. For instance, Zhang et al. [45] discussed the fundamentals of *n-link* robots known as a *1-link* robot system. Qiang et al. [46] pointed out that the importance of taking the characteristics of motor dynamics into account for the accuracy and stability requirements of robot movements to be achieved. More so, among the criteria that the motor dynamics need to satisfy is for the actual output of the system to track the desired output within an acceptable minimal error [47]. Motivated by the work of Zhang et al. [11], Sun et al. [7] applied an algorithm for solving real-valued unconstrained optimization to solve 2D robotic motion control.

We consider a motion control problem involving a two-joint planar robotic manipulator as described in [11]. Let $\zeta_k \in \mathbb{R}^2$ and $\varphi_k \in \mathbb{R}^2$ denote the joint angle vector and end effector position vector, respectively. A discrete time kinematics equation of a two-joint planar robot manipulator at a position level is governed by the following model:

$$h(\zeta_k) = \varphi_k. \tag{83}$$

The vector-valued function $h(\cdot)$ is referred to the kinematics mapping which has the following structure:

$$h(\zeta) = \left[\ell_1 \cos(\zeta_1) + \ell_2 \cos(\zeta_1 + \zeta_2), \ell_2 \sin(\zeta_1) + \ell_2 \sin(\zeta_1 + \zeta_2)\right]^T, \tag{84}$$

where the parameters ℓ_1 and ℓ_2 represent the lengths of the first and second rod, respectively. Now, with regards to robotic motion control, the following unconstrained optimization problem,

Table 4: Summary test results for behavior of parameters in the MSMDL method.

Test	NI	$\measuredangle_{\gamma_k=1}$	$\measuredangle_{t_k=\tau_k}$	μ
Extended penalty function	1575	758	757	−0.2533
Raydan 1 function	30614	14981	14983	−0.2565
Raydan 2 function	57	57	57	−1
Diagonal 1 function	21812	10876	10876	−0.2608
Diagonal 3 function	34503	17203	17203	−0.2711
Hager function	1430	606	607	−0.2600
Generalized tridiagonal 1 function	1124	477	477	−0.2600
Extended tridiagonal 1 function	1053	463	444	−0.2600
Extended TET function	1090	330	415	−0.2201
Diagonal 4 function	2058	925	925	−0.2557
Diagonal 5 function	40	40	34	−1
Extended Himmelblau function	1220	555	555	−0.2619
Extended quadratic penalty QP1 function	894	378	383	−0.2465
Extended quadratic exponential EP1 function	499	247	247	−0.26
ARWHEAD function (CUTE)	18117	9074	9059	−0.2494
ENGVAL1 function (CUTE)	1008	391	391	−0.2600
Diagonal 6 function	57	57	57	−1
Generalized quartic function	832	246	351	−0.2600
Diagonal 7 function	189	73	179	−0.26
Diagonal 8 function	270	109	263	−0.26
Full Hessian FH3 function	2249	1139	1143	−0.26
Diagonal 9 function	66739	33345	33345	−0.2617
Extended Rosenbrock	50	0	10	−0.26
Extended BD1 function (block diagonal)	1175	411	437	−0.2403
Extended Maratos function	9716	4760	4751	−0.1634
NONDQUAR function (CUTE)	31	15	15	−0.2719
DQDRTIC function (CUTE)	7082	3452	3452	−0.2600
Extended Freudenstein and Roth function	31821	15851	15851	−0.2612
Extended Beale function	1880	758	698	−0.1753
EDENSCH function (CUTE)	1050	389	387	−0.2600

Table 5: Summary test results of EDL, MDL, MSMDL, and BB1DL methods for NI, NFE, and CPUts.

Test function	EDL NI/NFE/CPUts	MDL NI/NFE/CPUts	MSMDL NI/NFE/CPUts	BB1DL NI/NFE/CPUts
Extended penalty function	2304/82602/32.719	1866/55194/20.516	1575/49766/18.766	1862/55274/19.813
Raydan 2 function	*/*/*	325/660/1.078	57/124/0.313	76/162/0.422
Hager function	1940/33206/85.703	1274/20621/45.203	1430/22994/52.063	1218/19995/43.641
Generalized tridiagonal 1 function	2161/33285/37.266	1250/18463/22.453	1124/17065/19.25	1329/19648/22.969
Extended tridiagonal 1 function	308/4129/11.063	5590/11602/32.516	1053/3292/12.797	452/1407/5.109
Diagonal 5 function	*/*/*	290/590/1.813	40/90/0.438	50/110/0.453
Extended Himmelblau function	50/2413/0.859	1344/21510/5.297	1220/21461/4.578	1350/21921/4.719
Extended quadratic penalty QP1 function	1157/18043/7.359	845/10736/5.125	894/13296/5.406	865/11335/4.766
Extended quadratic exponential EP1 function	21431/43829/6.25	481/10518/4.391	499/12243/4.938	481/10518/3.797
ENGVAL1 function (CUTE)	1975/27260/10.063	1167/16242/7.891	1008/13576/6.484	1160/16202/7.359
Diagonal 6 function	*/*/*	343/757/1.109	57/124/0.281	76/190/0.344
DIXON3DQ function (CUTE)	*/*/*	*/*/*	*/*/*	*/*/*
BIGGSB1 function (CUTE)	*/*/*	*/*/*	*/*/*	*/*/*
Generalized quartic function	959/10662/2.344	1459/10061/3.656	832/6769/2.984	1361/7935/3.266
Diagonal 7 function	*/*/*	110/457/0.75	189/1750/2.594	105/1128/1.453
Diagonal 8 function	*/*/*	125/1338/1.781	270/2318/2.844	71/254/0.438
Full Hessian FH3 function	*/*/*	1330/41888/20.891	2249/73823/25.156	1795/54550/20.578
Extended Rosenbrock	60/130/0.406	50/110/0.375	50/110/0.328	50/110/0.313
Extended BD1 function (block diagonal)	1200/12605/4.938	1160/8976/4.813	1175/10059/5.188	1030/7974/4.188
Extended cliff function	950/13187/4.891	46500/530421/273.781	4773/36089/21.719	1208/23850/12.703
NONDQUAR function (CUTE)	86/4989/15.781	45/2732/9.156	31/2721/10.156	24/2332/9.109
DQDRTIC function (CUTE)	3637/92315/28.766	2460/58286/19.141	7082/170118/49.594	2430/57850/19.938
Extended Freudenstein and Roth function	2018/66654/13.516	3524/113935/21.531	31821/1040192/213.016	2186/69397/14.531
Extended Beale function	181/4748/5.469	1593/19562/52.656	1880/24895/65.016	1300/17186/45.172
EDENSCH function (CUTE)	1684/22731/96.219	1300/16218/66.328	1050/13383/55.078	1132/14810/63.563

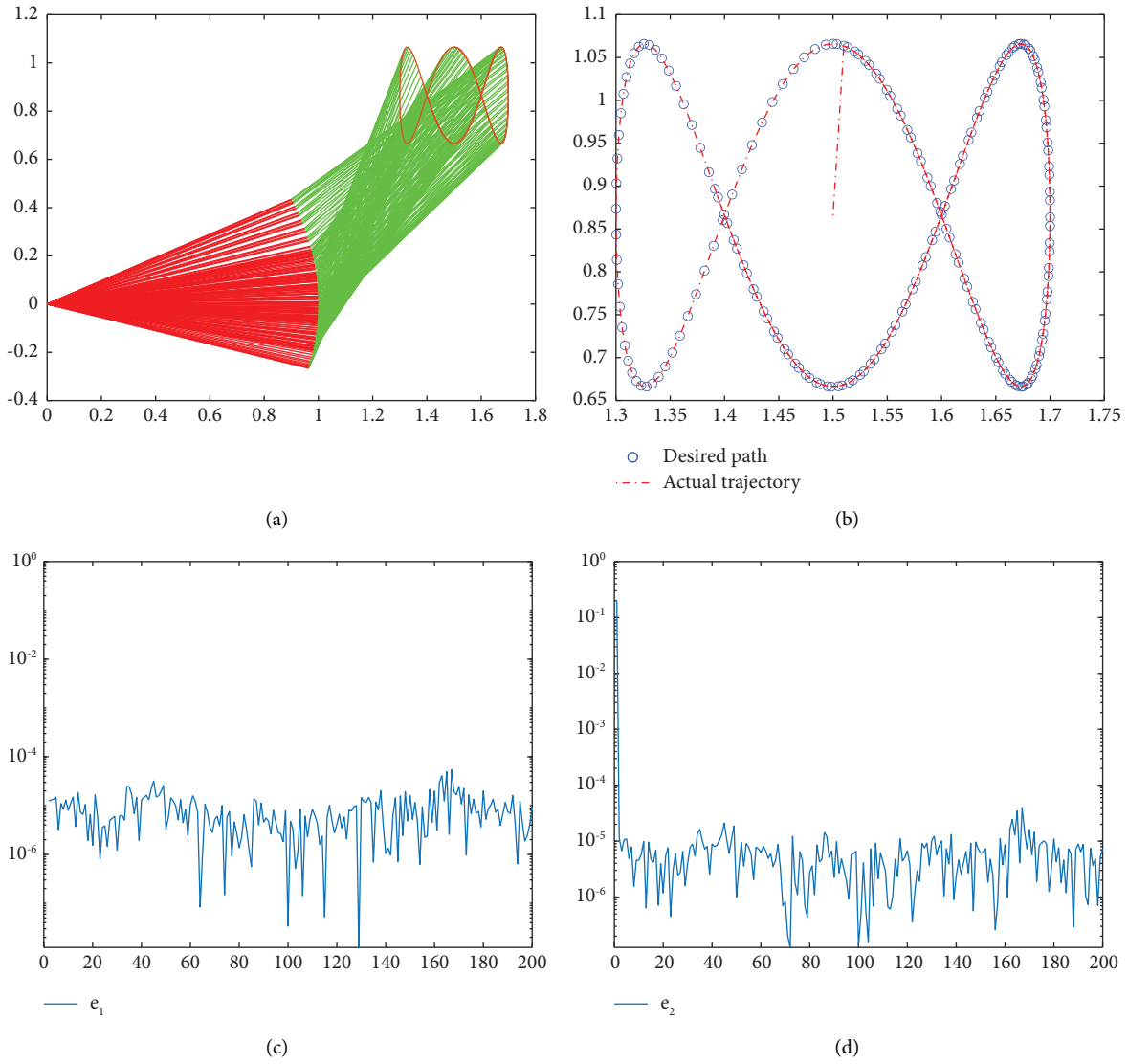

Figure 6: Numerical results generated by the MSMDL method for $\widehat{\varphi}_k = \widehat{\varphi}_{1k}$: (a) synthesized robot trajectories, (b) end effector trajectory and desired path, (c) tracking residual error on the x-axis, and (d) tracking residual error on the y-axis.

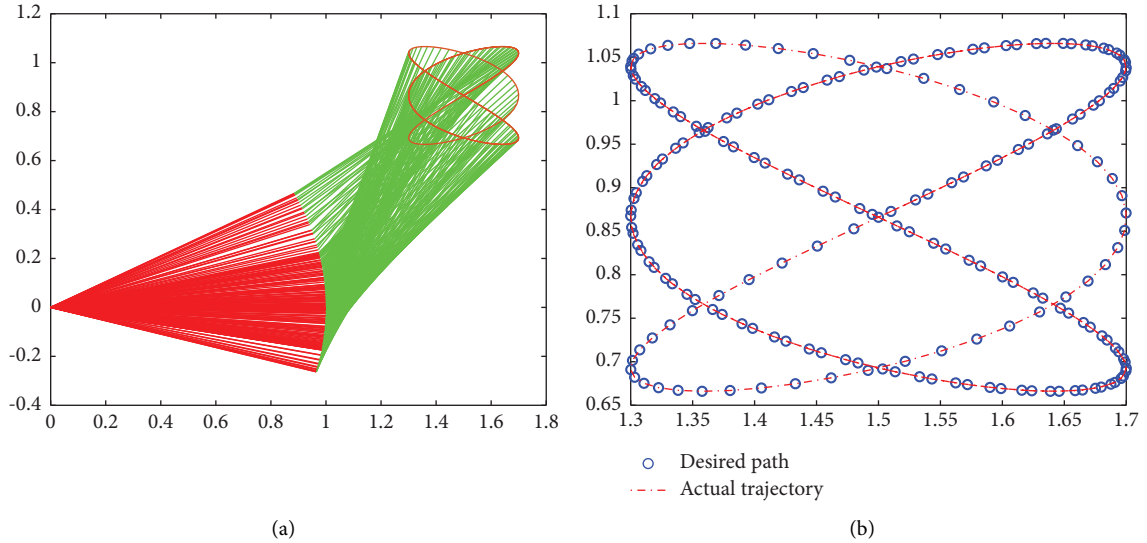

Figure 7: Continued.

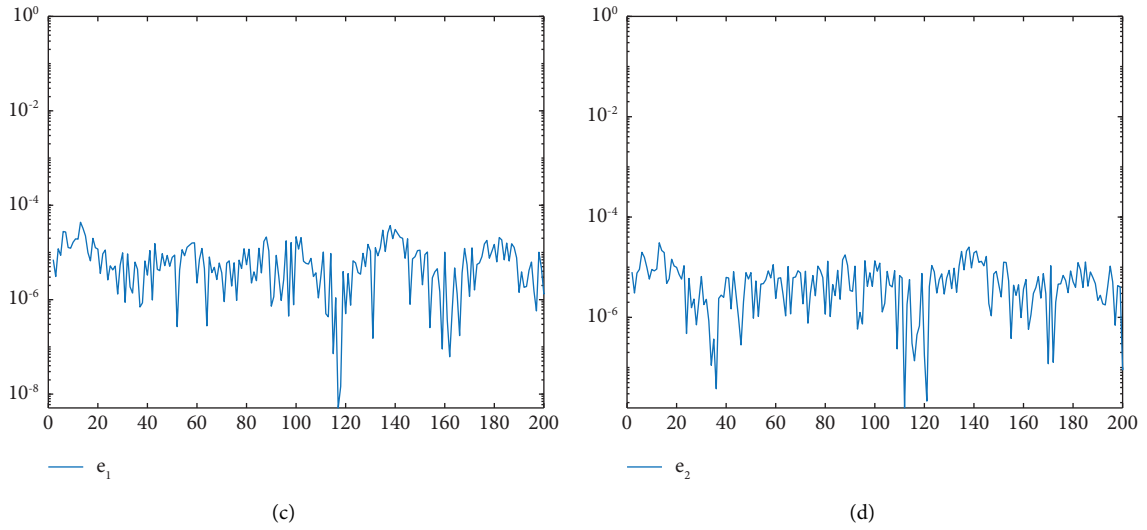

Figure 7: Numerical results generated by the MSMDL method for $\widehat{\varphi}_k = \widehat{\varphi}_{2k}$: (a) synthesized robot trajectories, (b) end effector trajectory and desired path, (c) tracking residual error on the x-axis, and (d) tracking residual error on the y-axis.

Table 6: Abbreviations for the methods.

Abbreviations	Full name
CG-DESCENT	CG-DESCENT
MSMDL	MSMDL
M1	M1
DK	Dai and Kou
BB1DL	BB1DL
EDL	Effective Dai-Liao
MDL	MDL

$$\min_{\varphi_k \in \mathbb{R}^2} \frac{1}{2}\left\| \varphi_k - \hat{\varphi}_k \right\|^2, \tag{85}$$

is solved at each instantaneous time t_k within the interval $[0, t_{\text{end}}]$ with t_{end} being the task duration. The end effector φ_k is usually controlled to track a Lissajous curve denoted by $\widehat{\varphi}_k$. We note that by taking $f(x) \equiv (1/2)\|\varphi_k - \hat{\varphi}_k\|^2$, then problem (85) has the form of problem (1), and therefore, the proposed MSMDL method can be used to solve it.

In this experiment, unlike the Lissajous curve used in [7, 48, 49], we require the end effector φ_k to track the following two Lissajous curves,

$$\begin{aligned} \widehat{\varphi}_{1k} &= \left[\frac{3}{2} + \frac{1}{5}\sin(t_k), \frac{\sqrt{3}}{2} + \frac{1}{5}\sin\left(3t_k + \frac{\pi}{2}\right)\right]^T, \\ \widehat{\varphi}_{2k} &= \left[\frac{3}{2} + \frac{1}{5}\sin(3t_k), \frac{\sqrt{3}}{2} + \frac{1}{5}\sin(2t_k)\right]^T. \end{aligned} \tag{86}$$

The implementation of the proposed MSMDL with regard to the motion control experiment was coded in MATLAB R2019b and run on a PC with an Intel Core (TM) i5-8250u processor with 4 GB of RAM and CPU 1.60 GHZ. In this experiment, the lengths of the first and second rods are taken as $\ell_1 = \ell_2 = 1$, where the initial point is $\zeta_0 = [\zeta_1, \zeta_2] = [0, (\pi/3)]^T$ with the task duration $[0, t_{\text{end}}]$ divided into 200 equal parts, where $t_{\text{end}} = 10$ seconds. The results experimental with $\widehat{\varphi}_k = \widehat{\varphi}_{1k}$ are presented in Figure 6, while that of $\widehat{\varphi}_k = \widehat{\varphi}_{2k}$ are given in Figure 7. Figures 6(a) and 7(a) depict the robot trajectories synthesized by the MSMDL for $\widehat{\varphi}_{1k}$ and $\widehat{\varphi}_{2k}$, respectively. On the other hand, Figures 6(b) and 7(b), respectively, plot the end effector trajectory and desired path for $\widehat{\varphi}_{1k}$ and $\widehat{\varphi}_{2k}$.

The errors recorded by MSMDL during the course of the experiment with respect to the horizontal axis are reported in Figures 6(c) and 7(c) for $\widehat{\varphi}_{1k}$ and $\widehat{\varphi}_{2k}$, respectively, while those of the vertical axis are equally presented in Figures 6(d) and 7(d) for $\widehat{\varphi}_{1k}$ and $\widehat{\varphi}_{2k}$, respectively.

Figures 6(a), 6(b), 7(a), and 7(b) confirm that the MSMDL method successfully and efficiently executed the task given to it with acceptable error. In addition, Figures 6(c), 6(d), 7(c), and 7(d) show that the proposed MSMDL method not only solves test problems but completes them with acceptable error. This further demonstrates the efficiency and applicability of the MSMDL method.

6. Conclusion

The main result obtained in this study is the verification and investigation of the correlation between QN and CG approaches. More specifically, it has been shown that QN and CG are not independent approaches, and QN iterates can be used to improve CG-type iterations.

Defining the best CG direction and also finding an optimal parameter for the DL CG class are open problems [14]. The parameters t_k^{MSMDL} of the MSMDL method and t_k^{BB1DL} of the BB1DL method are defined and selected using the unification of the MSM and the BB1 method and the CG DL class, respectively. Based on the presented numerical results in Tables 2 and 5 and Figures 2–5, we can conclude that the proposed MSMDL and BB1DL methods achieve superior numerical results with respect to all the three criteria (NI, NFE, and CPUts) versus opposing methods.

A modified Dai–Liao CG method termed the MSMDL method, intended for solving unconstrained optimization problems, is proposed and examined both theoretically and numerically. The modification is based on an appropriately scaled CG parameter t_k^{MSMDL} by means of the approximation of the Hessian matrix via the diagonal matrix $\gamma_k I$ in the MSM method. Our leading principle is to find the DL parameter t in (17) as a solution to the equation which appears after the unification of descent directions in the MSM method (11) and in the DL iterations (17). In this way, the expression $1 + \alpha_k - \alpha_k^2$ and the acceleration parameter γ_k^{MSM} from the MSM method will appear in the expression which determines t_k^{MSMDL}. The new parameter contains not only the gradient information but also some Hessian matrix information. The general perception is that the proposed iterations are defined as a hybridization of MSM and DL-type CG methods. Under some standard assumptions, the global convergence property of the MSMDL method is established. Numerical comparisons on a large class of well-known test problems, especially for solving high-dimensional problems, indicate that the MSMDL method is quite effective. The application of the MSMDL method on solving problems arising from 2D robotic motion control further confirms its efficiency as well as applicability.

The proposed unification of the search directions d_k of two methods of the general pattern (2) belonging to different classes is a continuation of a new branch of research in nonlinear optimization, which could be termed as *unification*. This research investigates the unification of MSM and BB methods with the DL CG solver to determine the parameter t. Further research may include the unification of search directions between different quasi-Newton methods and various CG methods. The method proposed in this paper is useful as a general principle for hybridizing any two methods from the conjugate gradient group or from the quasi-Newton methods group.

Appendix

Table 6 shows the abbreviations and full names of the methods. For most methods, the abbreviation and the full name of the method are identical.

Data Availability

The numerical results and graphical illustrations used to support the findings of this study are included within the article. The MATLAB code is available on request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work of the second author is supported in part by the Serbian Academy of Sciences and Arts (Φ-96). Predrag Stanimirović acknowledges support from the Science Fund of the Republic of Serbia (grant no. 7750185, Quantitative Automata Models: Fundamental Problems and Applications - QUAM). This work was supported by the Ministry of Science and Higher Education of the Russian Federation (grant no. 075-15-2022-1121).

References

[1] Y. H. Dai and L. Z. Liao, "New conjugacy conditions and related nonlinear conjugate gradient methods," *Applied Mathematics and Optimization*, vol. 43, no. 1, pp. 87–101, 2001.

[2] M. Bastani and D. K. Salkuyeh, "On the GSOR iteration method for image restoration," *Numerical Algebra, Control and Optimization*, vol. 11, no. 1, pp. 27–43, 2021.

[3] J. Cao and J. Wu, "A conjugate gradient algorithm and its applications in image restoration," *Applied Numerical Mathematics*, vol. 152, pp. 243–252, 2020.

[4] M. Dawahdeh, M. Mamat, M. Rivaie, and I. M. Sulaiman, "Application of conjugate gradient method for solution of regression models," *IJAST*, vol. 29, no. 7, pp. 1754–1763, 2020.

[5] L. Guo, X.-L. Zhao, X.-M. Gu, Y.-L. Zhao, Y.-B. Zheng, and T.-Z. Huang, "Three–dimensional fractional total variation regularized tensor optimized model for image deblurring," *Applied Mathematics and Computation*, vol. 404, Article ID 126224, 2021.

[6] A. Miele, T. Wang, and S. Mancuso, "Optimization of missions to Mars for robotic and manned spacecraft," *Nonlinear Analysis: Theory, Methods and Applications*, vol. 47, no. 3, pp. 1425–1443, 2001.

[7] M. Sun, J. Liu, and Y. Wang, "Two improved conjugate gradient methods with application in compressive sensing and motion control," *Mathematical Problems in Engineering*, vol. 2020, Article ID 9175496, 11 pages, 2020.

[8] Y. Xiao and H. Zhu, "A conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing," *Journal of Mathematical Analysis and Applications*, vol. 405, no. 1, pp. 310–319, 2013.

[9] G. Yuan, T. Li, and W. Hu, "A conjugate gradient algorithm and its application in large-scale optimization problems and image restoration," *Journal of Inequalities and Applications*, vol. 2019, no. 1, p. 247, 2019.

[10] G. Yuan, T. Li, and W. Hu, "A conjugate gradient algorithm for large-scale nonlinear equations and image restoration problems," *Applied Numerical Mathematics*, vol. 147, pp. 129–141, 2020.

[11] Y. Zhang, L. He, C. Hu, J. Guo, J. Li, and Y. Shi, "General four-stepdiscrete-time zeroing and derivative dynamics applied to time-varying nonlinear optimization," *Journal of Computational and Applied Mathematics*, vol. 347, pp. 314–329, 2019.

[12] N. Andrei, "A scaled BFGS preconditioned conjugate gradient algorithm for unconstrained optimization," *Applied Mathematics Letters*, vol. 20, no. 6, pp. 645–650, 2007.

[13] N. Andrei, "Scaled memoryless BFGS preconditioned conjugate gradient algorithm for unconstrained optimization," *Optimization Methods and Software*, vol. 22, no. 4, pp. 561–571, 2007.

[14] N. Andrei, "Open problems in nonlinear conjugate gradient algorithms for unconstrained optimization," *Bull. Malays. Math. Sci. Soc.*vol. 34, no. 2, pp. 319–330, 2011.

[15] M. Lotfi and S. M. Hosseini, "An efficient Dai–Liao type conjugate gradient method by reformulating the CG parameter in the search direction equation," *Journal of Computational and Applied Mathematics*, vol. 371, Article ID 112708, 2020.

[16] K. Samia and B. Djamel, "Hybrid conjugate gradient–BFGS methods based on Wolfe line search," *Stud. Univ. Babeş-Bolyai Math.*vol. 67, no. 4, pp. 855–869, 2022.
[17] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer-Verlag New York, Inc, New York, NY, USA, 1999.
[18] W. Sun and Y.-X. Yuan, *Optimization Theory and Methods: Nonlinear Programming*, Springer, Berlin, Germany, 2006.
[19] P. S. Stanimirović and M. B. Miladinović, "Accelerated gradient descent methods with line search," *Numerical Algorithms*, vol. 54, no. 4, pp. 503–520, 2010.
[20] C. Brezinski, "A classification of quasi-Newton methods," *Numerical Algorithms*, vol. 33, no. 1/4, pp. 123–135, 2003.
[21] B. Ivanov, P. S. Stanimirović, G. V. Milovanović, S. Djordjević, and I. Brajević, "Accelerated multiple step-size methods for solving unconstrained optimization problems," *Optimization Methods and Software*, vol. 36, no. 5, pp. 998–1029, 2021.
[22] B. Ivanov, B. I. Shaini, and P. S. Stanimirović, "Multiple use of backtracking line search in unconstrained optimization," *Facta Universitatis – Series: Mathematics and Informatics*, vol. 35, no. 5, pp. 1417–1438, 2021.
[23] P. S Stanimirović, B. Ivanov, H. Ma, and D. Mosić, "A survey of gradient methods for solving nonlinear optimization," *Electronic Research Archive*, vol. 28, no. 4, pp. 1573–1624, 2020.
[24] W. W. Hager and H. Zhang, "A new conjugate gradient method with guaranteed descent and an efficient line search," *SIAM Journal on Optimization*, vol. 16, no. 1, pp. 170–192, 2005.
[25] W. W. Hager and H. Zhang, "Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent," *ACM Transactions on Mathematical Software*, vol. 32, no. 1, pp. 113–137, 2006.
[26] Y. H. Dai and C. X. Kou, "A nonlinear conjugate gradient algorithm with an optimal property and an improved wolfe line search," *SIAM Journal on Optimization*, vol. 23, no. 1, pp. 296–320, 2013.
[27] S. Babaie-Kafaki and R. Ghanbari, "The Dai-Liao nonlinear conjugate gradient method with optimal parameter choices," *European Journal of Operational Research*, vol. 234, no. 3, pp. 625–630, 2014.
[28] B. Ivanov, P. S. Stanimirović, B. I. Shaini, H. Ahmad, and M.-K. Wang, "A novel value for the parameter in the dai-liao-type conjugate gradient method," *Journal of Function Spaces*, vol. 2021, Article ID 6693401, 10 pages, 2021.
[29] J. Barzilai and J. M. Borwein, "Two-point step size gradient methods," *IMA Journal of Numerical Analysis*, vol. 8, no. 1, pp. 141–148, 1988.
[30] D. di Serafino, V. Ruggiero, G. Toraldo, and L. Zanni, "On the steplength selection in gradient methods for unconstrained optimization," *Applied Mathematics and Computation*, vol. 318, pp. 176–195, 2018.
[31] M. Raydan, "On the Barzilai and Borwein choice of steplength for the gradient method," *IMA Journal of Numerical Analysis*, vol. 13, no. 3, pp. 321–326, 1993.
[32] M. Raydan, "The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem," *SIAM Journal on Optimization*, vol. 7, no. 1, pp. 26–33, 1997.
[33] Y. Zheng and B. Zheng, "Two new Dai-Liao-type conjugate gradient methods for unconstrained optimization problems," *Journal of Optimization Theory and Applications*, vol. 175, no. 2, pp. 502–509, 2017.
[34] Z. Aminifard and S. Babaie-Kafaki, "An optimal parameter choice for the Dai-Liao family of conjugate gradient methods by avoiding a direction of the maximum magnification by the search direction matrix," *4OR*, vol. 17, no. 3, pp. 317–330, 2019.
[35] G. Zoutendijk, "Nonlinear programming, computational methods," *Integer and Nonlinear Programming*, pp. 37–86, North-Holland, Amsterdam, Netherland, 1970.
[36] G. Wu, Y. Li, and G. Yuan, "A three-term conjugate gradient algorithm with quadratic convergence for unconstrained optimization problems," *Mathematical Problems in Engineering*, vol. 2018, Article ID 4813030, 15 pages, 2018.
[37] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equation in Several Variables*, Academic Press, New York, NY, USA, 1970.
[38] R. T. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, NJ, USA, 1970.
[39] K. U. Danmalam, H. Mohammad, and M. Y. Waziri, "Structured diagonal Gauss–Newton method for nonlinear least squares," *Computational and Applied Mathematics*, vol. 41, no. 2, p. 68, 2022.
[40] M. Li, "A family of three–term nonlinear conjugate gradient methods close to the memoryless BFGS method," *Optics Letters*, vol. 12, no. 8, pp. 1911–1927, 2018.
[41] N. Andrei, "An unconstrained optimization test functions collection," *Advanced Modeling and Optimization*, vol. 10, pp. 147–161, 2008.
[42] I. Bongartz, A. R. Conn, N. Gould, and Ph. L. Toint, "CUTE: constrained and unconstrained testing environments," *ACM Transactions on Mathematical Software*, vol. 21, no. 1, pp. 123–160, 1995.
[43] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical Programming*, vol. 91, no. 2, pp. 201–213, 2002.
[44] L. E. Yeganova, J. E. Falk, and Y. V. Dandurova, "Robust separation of multiple sets," *Nonlinear Analysis: Theory, Methods and Applications*, vol. 47, no. 3, pp. 1845–1856, 2001.
[45] Y. Zhang, W. Li, B. Qiu, Y. Ding, and D. Zhang, "Three-state space reformulation and control of MD-includedone-link robot system using direct-derivative and zhang-dynamics methods," in *Proceedings of the 2017 29th Chinese Control And Decision Conference (CCDC)*, pp. 3724–3729, IEEE, Chongqing, China, May 2017.
[46] Y. Qiang, F. Jing, J. Zeng, and Z. Hou, "Dynamic modeling and vibration mode analysis for an industrial robot with rigid links and flexible joints," in *Proceedings of the 2012 24th Chinese Control and Decision Conference (CCDC)*, IEEE, Taiyuan, China, May 2012.
[47] G. Y. Tang, L. Sun, C. Li, and M. Q. Fan, "Successive approximation procedure of optimal tracking control for nonlinear similar composite systems," *Nonlinear Analysis: Theory, Methods and Applications*, vol. 70, no. 2, pp. 631–641, 2009.
[48] A. M. Awwal, P. Kumam, L. Wang, S. Huang, and W. Kumam, "Inertial-basedderivative-free method for system of monotone nonlinear equations and application," *IEEE Access*, vol. 8, pp. 226921–226930, 2020.
[49] M. M. Yahaya, P. Kumam, A. M. Awwal, and S. Aji, "A structured quasi-Newton algorithm with nonmonotone search strategy for structured NLS problems and its application in robotic motion control," *Journal of Computational and Applied Mathematics*, vol. 395, no. 5, Article ID 113582, 2021.
[50] W. Cheng, "A two-term PRP-based descent method," *Numerical Functional Analysis and Optimization*, vol. 28, no. 11–12, pp. 1217–1230, 2007.