ELECTRONIC RESEARCH ARCHIVE
Volume **28**, Number **4**, December **2020**

doi:10.3934/era.2020115
pp. **1573–1624**

A SURVEY OF GRADIENT METHODS FOR SOLVING NONLINEAR OPTIMIZATION

Predrag S. Stanimirović*

University of Niš, Faculty of Sciences and Mathematics
Višegradska 33, 18000 Niš, Serbia

Branislav Ivanov[1], Haifeng Ma[2] and Dijana Mosić[3]

[1] Technical Faculty in Bor, University of Belgrade
Vojske Jugoslavije 12, 19210 Bor, Serbia
[2] School of Mathematical Science, Harbin Normal University
Harbin 150025, China
[3] University of Niš, Faculty of Sciences and Mathematics
Višegradska 33, 18000 Niš, Serbia

Abstract. The paper surveys, classifies and investigates theoretically and numerically main classes of line search methods for unconstrained optimization. Quasi-Newton (QN) and conjugate gradient (CG) methods are considered as representative classes of effective numerical methods for solving large-scale unconstrained optimization problems. In this paper, we investigate, classify and compare main QN and CG methods to present a global overview of scientific advances in this field. Some of the most recent trends in this field are presented. A number of numerical experiments is performed with the aim to give an experimental and natural answer regarding the numerical one another comparison of different QN and CG methods.

1. **Introduction and preliminaries.** In this survey, we focus on solving the unconstrained nonlinear optimization problem

$$\min f(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n, \tag{1.1}$$

where the function $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and bounded from below. The general iterative rule for solving (1.1) starts from an initial approximation $\mathbf{x}_0 \in \mathbb{R}^n$ and generates a sequence $\{\mathbf{x}_k, k \geq 0\}$ using the general iterative scheme

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k, \quad k \geq 0, \tag{1.2}$$

where the step-size α_k is a positive real parameter determined after the exact or inexact line search, $\mathbf{x}_k$ is the last generated iterative point, $\mathbf{x}_{k+1}$ is the current iterative point, and $\mathbf{d}_k$ is an appropriate search direction. General class of algorithms of the form (1.2) is known as the *line search algorithms*. These algorithms require only the search direction $\mathbf{d}_k \in \mathbb{R}^n$ and the step-size $\alpha_k \in \mathbb{R}$.

The following notations will be used, as usual:

$$\mathbf{g}(\mathbf{x}) = \nabla f(\mathbf{x}),\ G(\mathbf{x}) = \nabla^2 f(\mathbf{x}),\ \mathbf{g}_k = \nabla f(\mathbf{x_k}),\ G_k = \nabla^2 f(\mathbf{x}_k),$$

2020 *Mathematics Subject Classification.* Primary: 65K05, 90C30; Secondary: 90C06, 90C26.
Key words and phrases. Gradient methods, conjugate gradient methods, unconstrained optimization, nonlinear programming, sufficient descent condition, global convergence, line search.

* Corresponding author: Predrag S. Stanimirović.

where $\nabla f(\mathbf{x})$ denotes the gradient and $\nabla^2 f(\mathbf{x})$ denotes the Hessian of f. For the sake of simplicity, the notation f_k will point to $f(\mathbf{x}_k)$. Further, $\mathbf{x}^{\mathrm{T}}$ denotes the transpose of $\mathbf{x} \in \mathbb{R}^n$. Taylor's approximation of the function f at the point $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$ is defined by

$$f(\mathbf{x}_{k+1}) \approx f(\mathbf{x}_k) + \alpha_k \mathbf{g}_k^{\mathrm{T}} \mathbf{d}_k. \tag{1.3}$$

Therefore, an appropriate descent search direction $\mathbf{d}_k$ must be determined on the basis of the *descent condition*

$$\mathbf{g}_k^{\mathrm{T}} \mathbf{d}_k < 0, \ \text{ for all } \ k. \tag{1.4}$$

Primary choice for descent direction is $\mathbf{d}_k = -\mathbf{g}_k$, which reduces the general line search iterations (1.2) into the *gradient descent* (GD) iterative scheme

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k. \tag{1.5}$$

In this paper, we survey gradient methods satisfying the descent condition (1.4). If there exists a constant $c > 0$ such that

$$\mathbf{g}_k^{\mathrm{T}} \mathbf{d}_k < -c\|\mathbf{g}_k\|^2, \ \text{ for all } \ k, \tag{1.6}$$

where c is a positive constant independent of k, then it is said that the vector $\mathbf{d}_k$ satisfies the *sufficient descent condition*.

As for the choice of search direction, one of the possible choices for the search direction in unconditional optimization is to move from the current point along the negative gradient in each iteration, which correspond to $\mathbf{d}_k = -\mathbf{g}_k$. This choice of search direction leads us to a class of methods known as *gradient descent methods*. One negative feature of gradient methods is relatively frequent occurrence of, so called, zig-zagging phenomenon, which initiates very slow convergence of GD algorithms to the optimal point, or even divergence [90].

Advantages and disadvantages of GD methods can be summarized as follows.

1. GD methods are globally convergent, i.e., converge to a local minimizer regardless of the starting point.
2. Many optimization methods switch to GD rule when they do not make sufficient progress to the convergence.
3. The convergence is linear and usually very slow.
4. Numerically, GD methods are often not convergent.

Another important direction of the search is the *Newton's direction* $\mathbf{d}_k = -G_k^{-1}\mathbf{g}_k$, obtained from the second-order Taylor-development, assuming that Hessian G_k is positive-definite. The pure Newton method (without line search) for minimization of a function $f : \mathbb{R}^n \to \mathbb{R}$ is defined using a quadratic approximation of $f(\mathbf{x}_{k+1})$:

$$\Phi(\mathbf{d}) := f(\mathbf{x}_k + \mathbf{d}) \approx f(\mathbf{x}_k) + \mathbf{g}_k^{\mathrm{T}} \mathbf{d} + \frac{1}{2} \mathbf{d}^{\mathrm{T}} G_k \mathbf{d}. \tag{1.7}$$

The solution $\mathbf{d}_k = \min_{\mathbf{d}}(\Phi(\mathbf{d}))$ is given by

$$\mathbf{d}_k = -G_k^{-1}\mathbf{g}_k.$$

So, the pure Newton method is defined by

$$\mathbf{x}_{k+1} = \mathbf{x}_k - G_k^{-1}\mathbf{g}_k. \tag{1.8}$$

The Newton method with line search uses an appropriate step-size α_k in (1.8) with the aim to ensure global stability. The resulting iterations are of the form

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k G_k^{-1}\mathbf{g}_k, \tag{1.9}$$

wherein the step-size α_k is computed performing a line search.

The Newton method exhibits three major drawbacks in practical applications.

1. The descent (and convergence) may not be achieved if the iterations (1.8) are started far away from the local minimizer.
2. Another drawback is numerically expensive and tedious necessity to compute the second derivative matrix (Hessian) and its inverse in every iteration. Moreover, the second derivatives may be sometimes unavailable.
3. The main disadvantages of the Newton method are the possibility that the Hessian G_k is not positive definite.

Due to that, numerous modifications of it were created, which can be globally divided into two large groups: modified Newton's methods and *quasi-Newton* (QN) methods. The QN methods are aimed to address all the above difficulties of the Newton method. The first drawback is overcome by taking an appropriately defined positive definite matrix B_k that approximates the Hessian G_k or an appropriately defined positive definite matrix H_k that approximates the true Hessian inverse G_k^{-1} and then performing a line search at each iteration. For a given initial point $\mathbf{x}_0 \in \mathbb{R}^n$ and a symmetric positive definite matrix H_0, the search direction in the kth iteration of the quasi-Newton method is defined by $\mathbf{d}_k = -H_k\mathbf{g}_k$, where H_k is a symmetric and positive-definite matrix.

QN methods and modified Newton methods belong to most powerful methods for solving unconditional optimization and applicable in many nonlinear optimization problems. A survey of QN methods for solving nonlinear least-squares problems was considered in [86]. The optimization methods have found a number of applications in fluid mechanics [44], free surface flow and solid body contact [13], finding the optimal trajectory for an aircraft or a robot arm, designing a portfolio of investments, controlling a chemical process, computing the optimal shape of an automobile. See [90] for more details. A modification of the quasi-Newton method in defining the two-phase approximate greatest descent was used in [71]. Several variants of multistep spectral gradient methods for solving large scale unconstrained optimization problems were proposed in [111]. Usage of an optimization algorithm in artificial neural networks was considered in [75]. Properties of Hessian matrix which appear in distributed gradient-based multi-agent control systems was considered in [117]. A survey of derivative-free optimization methods was given in [127]. An application of unconstrained optimization in solving the risk probability was presented in [76].

The study of conjugate gradient (CG) methods was started by Hestenes and Stiefel in 1952 in [60], and the development of CG methods for solving large-scale unconstrained optimization problems is still ongoing. After all these years, there is still a need to find a more efficient CG method that will solve unconstrained optimization problems with thousands of variables in the shortest possible time interval as well as with a minimal number of iterations and function evaluations.

CG methods construct a sequence of approximations $\mathbf{x}_k$ by the line search rule (1.2), such that the search directions $\mathbf{d}_k$ are generated by

$$\mathbf{d}_k := \mathfrak{d}_k := \mathfrak{d}(\beta_k, \mathbf{g}_k, \mathbf{d}_{k-1}) = \begin{cases} -\mathbf{g}_0, & k=0, \\ -\mathbf{g}_k + \beta_k \mathbf{d}_{k-1}, & k \geq 1, \end{cases} \tag{1.10}$$

where β_k is the real value which is known as the conjugate gradient update parameter (CGUP). More precisely, the search direction $\mathbf{d}_k$ of the CG method is defined as a proper linear combination of the gradient descent direction and a positive multiple of the direction used in the previously finished iteration. From (1.10) and (1.2), it

clearly follows that the CG methods are defined simply only by the gradient direction $\mathbf{g}_k$ and by the CGUP β_k. Different CG methods arise from proper choices of the scalar β_k. According to the common agreement, $\beta_k^{\mathcal{M}}$ denotes the parameter β_k of the CG method $\mathcal{M}$. It is important to mention that some researchers propose usage of $\beta_k^{\mathcal{M}+} = \max\{\beta_k^{\mathcal{M}}, 0\}$. So, it is possible to use $\beta_k^{\mathcal{M}+}$ instead of $\beta_k^{\mathcal{M}}$ and generate corresponding dual method.

Popularity of CG methods is confirmed by a number of recent surveys and book chapters [42, 57, 85, 87, 88]. In addition to this basic information on the chronological development of the CG methods, it is also important to mention its applications. In general, CG methods are important in solving large-scale optimization problems. CG methods iterates are characterized by low memory allocation and strong local and global convergence properties. Based on this fact, these methods become useful in all areas where optimization problems of any kind are involved. The CG methods have wide use in solving systems of equations and image restoration problem [12, 16, 70, 78, 128, 135, 136, 140], the linear response eigenvalue problem [74], in regression analysis [108, 143]. On that way, CG methods have the influence on the development of an artificial neural networks learning algorithms [46, 75]. A unique approach to the ABS type CG methods was proposed in [1]. Application of CG methods in solving very large symmetric positive semi definite linear systems that appear in optimal surface parameterizations are described in [64]. Also, it is possible to mention application in data analysis [110]. A variant of the projected preconditioned conjugate gradient method and its application in solving the linear response eigenvalue problem was investigated in [74].

Main goals leading current research paper can be highlighted as follows.

(1) A survey and specific classifications of CG and QN methods for nonlinear unconstrained optimization is presented.

(2) Convergence properties of CG methods are investigated.

(3) Specific numerical testings are performed on both the CG and QN methods. Numerical testing on some classes of CG methods and hybrid CG methods as well as on some QN methods is presented. A numerical experiment about the influence of the scalar t in Dai-Liao CG methods is performed and analysed. Also, gradient descent methods defined by appropriate acceleration parameter are tested and compared.

The overall structure of the paper based on contents of each section is described as follows. Section 1 describes basic notation, introductory notions, preliminaries and motivation. Global algorithms and various line search variants are presented in Section 2. Overview of QN methods and their classification are considered in Section 3. Section 4 gives a specific overview of CG methods. Convergence properties of considered CG methods are investigated in Section 5. According to the presented taxonomy of basic CG methods, properties of CG methods with $\mathbf{y}_{k-1}^T\mathbf{g}_k$ in the numerator of β_k are considered in Subsection 5.1, properties of CG methods involving $\|\mathbf{g}_k\|^2$ in the numerator of β_k are given in Subsection 5.2, while the convergence properties of DL methods are presented in Subsection 5.3. Numerical experiments are performed in Section 6. In details, Subsection 6.1 arranges numerical results on QN methods with constant diagonal Hessian approximation, Subsection 6.2 compares numerically basic CG methods involving $\mathbf{y}_{k-1}^T\mathbf{g}_k$ in the numerator of β_k, Subsection 6.3 compares basic CG methods with $\|\mathbf{g}_k\|^2$ in the

numerator of β_k, while numerical experiments on the hybrid CG methods are presented in Subsection 6.4. Finally, Subsection 6.5 describes numerical experiments on the modified Dai-Liao methods. Concluding remarks are given in Section 7.

2. **Global algorithms and line search variants.** First, we present an algorithm that describes the general scheme of line search methods

Algorithm 1 Gloab line search algorithm

Require: Objective $f(\mathbf{x})$, initial point $\mathbf{x}_0 \in \mathbb{R}^n$ and the tolerance $0 < \varepsilon \ll 1$.
1: $k := 0$.
2: **while** $\|\mathbf{g}_k\| > \varepsilon$ **do**
3: Determine the vector $\mathbf{d}_k$ which represents the search direction.
4: Compute the step length α_k
5: Compute the new approximation of the minimum $\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k \mathbf{d}_k$
6: $k := k + 1$
7: **end while**
Ensure: $\mathbf{x}_{k+1}, f(\mathbf{x}_{k+1})$

2.1. **Line search procedures.** To achieve the global convergence of iterative methods, an appropriate step-size α_k is required. The most promising at first glance is the exact line search (ELS), which assumes the unidimensional function

$$\Phi(\alpha) := f(\mathbf{x}_k + \alpha \mathbf{d}_k) \tag{2.1}$$

and the step-size is defined after the unidimensional optimization of the form

$$f(\mathbf{x}_k + \alpha_k \mathbf{d}_k) = \min_{\alpha > 0} \Phi(\alpha). \tag{2.2}$$

The ELS rule may give the most precise minimum. However, ELS is too expensive in practice or even impossible to implement, especially in situations where the iteration is far from the exact solution.

Applying the iterative procedure (1.2), it is most logical to choose a new point so that the step length α_k reduces the value of the goal function:

$$\Phi(\alpha_k) = f(\mathbf{x}_{k+1}) < \Phi(0) = f(\mathbf{x}_k). \tag{2.3}$$

Methods that in each iterative step require the fulfillment of conditions (2.3), that is, the reduction of the value of the objective function, define iterations that in each step approach the minimum of the given function. The methods conceived in this way belong to the class of methods of *monotone line search*. Many variants of *inexact line search* (ILS) rules are proposed and dominant in the nonlinear optimization. The most commonly used ILS techniques are Armijo, Goldstein, Wolfe, Powel, Fletcher and other [4, 8, 19, 50, 55, 56, 109, 126]. In most conjugate gradient methods, one of the next ILS procedures methods is used to calculate the step length α_k: Wolfe line search developed in [55, 56], strong Wolfe line search, or backtracking line search from [4].

In contrast to the monotonic line search, the *non-monotonic line search* is also known in the literature, where it is not necessary to reduce the value of the objective function in each iteration [51, 52, 53]. Although non-monotone techniques do not provide a minimum approach to the function in each iteration, they are very common in practical applications and have very good convergence properties. A

number of nonmonotonic linear search methods have been proposed recently (see, for example, [119, 120]).

2.1.1. *Backtracking line search.* A backtracking line search scheme based on the Armijo condition, is aimed to determining the maximal value during the moving along a given search vector. It starts with a relatively large step-size estimate and iteratively reduces the step-size value until a decrease in the value of the objective function is observed, according to the local gradient of the goal function. Let $\beta \in (0,1)$, $\varphi \in (0,1)$ and $\alpha > 0$ be given. Then there exists a smallest nonnegative integer m_k satisfying

$$f(\mathbf{x}_k + \beta^{m_k} t\mathbf{d}_k) \leq f(\mathbf{x}_k) + \varphi\beta^{m_k} t\mathbf{g}_k^T\mathbf{d}_k,\ t > 0. \tag{2.4}$$

The procedure for backtracking line search proposed in [4] starts from the initial value $\alpha = 1$ and its output values are defined such that it decreases the goal function. Consequently, Algorithm 2 from [112] is used in numerical experiments as the implementation of the ILS which defines the principal step-size α_k.

Algorithm 2 The backtracking line search.

Require: Nonlinear multivariate function $f(\mathbf{x})$, the vector $\mathbf{d}_k$, previous approximation $\mathbf{x}_k$, and the real numbers $0 < \omega < 0.5$ and $\varphi \in (0,1)$.

1: $\alpha = 1$.
2: While $f(\mathbf{x}_k + \alpha\mathbf{d}_k) > f(\mathbf{x}_k) + \omega\alpha\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_k$, take $\alpha := \alpha\varphi$.
3: Return $\alpha_k = \alpha$.

2.1.2. *Goldstein line search.* In order to ensure a sufficient decrease of the objective function, Goldstein rule for ILS requires the following conditions:

$$f(\mathbf{x}_k + \alpha\mathbf{d}_k) \leq f(\mathbf{x}_k) + \rho t\mathbf{g}_k^T\mathbf{d}_k \tag{2.5}$$

and

$$f(\mathbf{x}_k + \alpha\mathbf{d}_k) \geq f(\mathbf{x}_k) + (1-\rho)t\mathbf{g}_k^T\mathbf{d}_k, \tag{2.6}$$

where $0 < \rho < \frac{1}{2}$ and $t > 0$. Conditions (2.5) and (2.6) define the Goldstein rule for inexact line search.

2.1.3. *Wolfe line search.* Wolfe line search conditions are well-known and these are given by

$$f(\mathbf{x}_k + \alpha_k\mathbf{d}_k) \leq f(\mathbf{x}_k) + \eta\alpha_k\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_k, \tag{2.7}$$

$$\mathbf{g}(\mathbf{x}_k + \alpha_k\mathbf{d}_k)^{\mathrm{T}}\mathbf{d}_k \geq \sigma_1\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_k, \tag{2.8}$$

where $0 < \eta < \sigma_1 < 1$. In addition, for conjugate gradient methods, the generalized strong Wolfe conditions, which are a conjunction of (2.7) and

$$-\sigma_2\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_k \geq g(\mathbf{x}_k + \alpha_k\mathbf{d}_k)^{\mathrm{T}}\mathbf{d}_k \geq \sigma_1\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_k \tag{2.9}$$

are often used, where $\sigma_1 > 0$. In the case $\sigma_1 = \sigma_2$, the generalized strong Wolfe conditions reduce to the strong Wolfe conditions, which are a conjunction of (2.7) and

$$|\mathbf{g}(\mathbf{x}_k + \alpha_k\mathbf{d}_k)^{\mathrm{T}}\mathbf{d}_k| \leq -\sigma_1\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_k. \tag{2.10}$$

The condition (2.7) of the Wolfe conditions is called the Armijo condition, which is often used apart or in the form of its variants.

3. **Quasi-Newton methods and their classification.** The most general iterative rule of QN type with line search is of the form

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k H_k \, \mathbf{g}_k, \tag{3.1}$$

such that H_k is an approximation of the inverse Hesiian G_k^{-1}. Further, it is assumed that B_k is an appropriately generated symmetric positive definite approximation of G_k [118]. The following notations in defining an appropriate updating formula

$$\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k, \quad \mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k \tag{3.2}$$

are typical. The update B_{k+1} of B_k is defined using the rule

$$B_{k+1} = B_k + E_k, \tag{3.3}$$

where E_k is defined on the basis of the quasi-Newton property (*secant condition*)

$$B_{k+1}\mathbf{s}_k = \mathbf{y}_k. \tag{3.4}$$

The quasi-Newton condition for the matrix H_k is given by

$$H_{k+1}\mathbf{y}_k = \mathbf{s}_k. \tag{3.5}$$

Methods that require the calculation or approximation of the Hessian matrix and its inverse belong to the class of QN methods as well as its numerous modifications. The pure Newton method requires calculation of second derivatives matrix, which is avoided in QN methods. As a consequence, the Newton method is computationally expensive and exhibits slow computation, while QN methods are computationally cheap and of faster computation. On the other hand, the Newton method requires lesser number of iterative steps and generates more precise convergence path than QN methods.

3.1. **Symmetric Rank-One update.** The Symmetric Rank-One update (SR1) assumes the matrix H_{k+1}in the form

$$H_{k+1} = H_k + E_k,$$

where E_k is assumed to be a symmetric rank-one matrix. Therefore,

$$H_{k+1} = H_k + \mathbf{u}_k \mathbf{v}_k^T, \tag{3.6}$$

where

$$\mathbf{u}_k, \mathbf{v}_k \in \mathbb{R}^n.$$

The quasi-Newton condition (3.5) initiates

$$H_{k+1}\mathbf{y}_k = (H_k + \mathbf{u}_k \mathbf{v}_k^T)\mathbf{y}_k = \mathbf{s}_k,$$

that is

$$(\mathbf{v}_k^T \mathbf{y}_k)\mathbf{u}_k = \mathbf{s}_k - H_k \mathbf{y}_k. \tag{3.7}$$

The conclusion is that $\mathbf{u}_k$ must be in the direction $\mathbf{s}_k - H_k\mathbf{y}_k$. Suppose that

$$\mathbf{s}_k - H_k\mathbf{y}_k \neq 0,$$

(otherwise H_k would satisfy the quasi-Newton equation) and the vector $\mathbf{v}_k$ satisfies $\mathbf{v}_k^T\mathbf{y}_k \neq 0$. Then, on the basis of (3.6) and (3.7), it follows that

$$H_{k+1} = H_k + \frac{1}{\mathbf{v}_k^T \mathbf{y}_k}(\mathbf{s}_k - H_k\mathbf{y}_k)\mathbf{v}_k^T. \tag{3.8}$$

The condition that the approximation H_{k+1} of the Hessian inverse is symmetric requires us to take $\mathbf{v}_k = \mathbf{s}_k - H_k\mathbf{y}_k$, so it was obtained

$$H_{k+1} = H_k + \frac{(\mathbf{s}_k - H_k\mathbf{y}_k)(\mathbf{s}_k - H_k\mathbf{y}_k)^T}{(\mathbf{s}_k - H_k\mathbf{y}_k)^T\mathbf{y}_k}, \tag{3.9}$$

which is the Symmetric Rank One (SR1) update.

For general functions, Conn, Gould and Toint in [20] proved that the sequence of SR1 Hessian approximations converges to the true Hessian provided that the steps are uniformly linearly independent.

3.2. **DFP update.** The Hessian update B_{k+1} is defined as the solution to the problem

$$\min_B \|B - B_k\|, \quad \text{s.t. } B = B^{\mathrm{T}},\ B\mathbf{s}_k = \mathbf{y}_k. \tag{3.10}$$

The solution to (3.10) is equal to

$$B_{k+1}^{\mathrm{DFP}} = (I - \rho_k\mathbf{y}_k\mathbf{s}_k^{\mathrm{T}})B_k^{\mathrm{DFP}}(I - \rho_k\mathbf{s}_k\mathbf{y}_k^{\mathrm{T}}) + \rho_k\mathbf{y}_k\mathbf{y}_k^{\mathrm{T}}, \quad \varrho_k = \frac{1}{y_k^{\mathrm{T}}s_k}.$$

The inverse Hessian update can be generated using the Sherman - Morrison - Woodbury.

Moreover, the DFP update is known as a method of updating, of rank 2, that is, H_{k+1} is formed by adding the matrix H_k with two symmetric matrices, each of which is rank 1:

$$H_{k+1} = H_k + a\mathbf{u}_k\mathbf{u}_k^T + b\mathbf{v}_k\mathbf{v}_k^T,$$

where $\mathbf{u}_k, \mathbf{v}_k \in \mathbb{R}^n$, and a, b are scalars. From the quasi-Newton condition (3.5) it follows that

$$H_k\,\mathbf{y}_k + a\,\mathbf{u}_k\mathbf{u}_k^T\mathbf{y}_k + b\,\mathbf{v}_k\mathbf{v}_k^T\mathbf{y}_k = \mathbf{s}_k. \tag{3.11}$$

The vectors $\mathbf{u}_k$ and $\mathbf{v}_k$ are not unique, but they can obviously be determined in the following way

$$\mathbf{u}_k = \mathbf{s}_k, \quad \mathbf{v}_k = H_k\mathbf{y}_k.$$

Now, (3.11) implies

$$a = \frac{1}{\mathbf{u}_k^T\mathbf{y}_k} = \frac{1}{\mathbf{s}_k^T\mathbf{y}_k}, \quad b = -\frac{1}{\mathbf{v}_k^T\mathbf{y}_k} = -\frac{1}{\mathbf{y}_k^TH_k\mathbf{y}_k}.$$

So we get

$$H_{k+1}^{\mathrm{DFP}} = H_k + \frac{\mathbf{s}_k\mathbf{s}_k^T}{\mathbf{s}_k^T\mathbf{y}_k} - \frac{H_k\mathbf{y}_k\mathbf{y}_k^TH_k}{\mathbf{y}_k^TH_k\mathbf{y}_k}. \tag{3.12}$$

Formula (3.12) was proposed by Davidon and later was developed by Fletcher and Powell, so that it is called DFP update.

3.3. **BFGS update.** One famous broadly used updating formula is the Broyden-Fletcher-Goldfarb–Shanno (BFGS) rule. The inverse Hessian update H_{k+1} is defined as the solution to

$$\min_H \|H - H_k\|, \quad \text{s.t. } H = H^{\mathrm{T}},\ H\mathbf{y}_k = \mathbf{s}_k. \tag{3.13}$$

Certainly, the BFGS update is overtly known as

$$\begin{aligned} H_{k+1}^{\mathrm{BFGS}} &= (I - \rho_k\mathbf{s}_k\mathbf{y}_k^{\mathrm{T}})H_k^{\mathrm{BFGS}}(I - \rho_k\mathbf{y}_k\mathbf{s}_k^{\mathrm{T}}) + \rho_k\mathbf{s}_k\mathbf{s}_k^{\mathrm{T}}, \quad \rho_k = \frac{1}{y_k^{\mathrm{T}}s_k} \\ &= H_k^{\mathrm{BFGS}} - \frac{H_k\mathbf{y}_k\mathbf{s}_k^{\mathrm{T}} + \mathbf{s}_k\mathbf{y}_k^{\mathrm{T}}H_k}{\mathbf{s}_k^{\mathrm{T}}\mathbf{y}_k} + \left(1 + \frac{\mathbf{y}_k^{\mathrm{T}}H_k\mathbf{y}_k}{\mathbf{s}_k^{\mathrm{T}}\mathbf{y}_k}\right)\frac{\mathbf{s}_k\mathbf{s}_k^{\mathrm{T}}}{\mathbf{s}_k^{\mathrm{T}}\mathbf{y}_k}, \end{aligned} \tag{3.14}$$

where $\mathbf{s}_k$ and $\mathbf{y}_k$ are defined in (3.2).

The update BFGS formula for the Hessian matrix can be generated using the Sherman-Morrison-Woodbury formula. A rank-one-modification (or perturbation) $M = A + \mathbf{bc}^*$ of a matrix $A \in \mathbb{C}^{m\times n}$ uses two vectors $\mathbf{b} \in \mathbb{C}^{m\times 1}$ and $\mathbf{c} \in \mathbb{C}^{n\times 1}$. The Sherman-Morrison formula establishes a relationship between M^{-1} and A^{-1} as follows [45]:

$$M^{-1} = A^{-1} - \left(1 + \mathbf{c}^* A^{-1}\mathbf{b}\right)^{-1} A^{-1}\mathbf{bc}^* A^{-1}. \tag{3.15}$$

As a result, the following update for B_k is obtained:

$$B_{k+1}^{\mathrm{BFGS}} = B_k^{\mathrm{BFGS}} - \frac{B_k\mathbf{s}_k\mathbf{s}_k^{\mathrm{T}}B_k}{\mathbf{s}_k^{\mathrm{T}}B_k\mathbf{s}_k} + \frac{\mathbf{y}_k\mathbf{y}_k^{\mathrm{T}}}{\mathbf{s}_k^{\mathrm{T}}\mathbf{y}_k}. \tag{3.16}$$

3.4. **Broyden family of methods.** The weighted combinations of DFP and BFGS updates give the whole update class, which is known as the Broyden class. This class of update is defined by

$$H_{k+1}^{\phi} = (1-\phi)H_{k+1}^{\mathrm{DFP}} + \phi H_{k+1}^{\mathrm{BFGS}}, \tag{3.17}$$

where ϕ is a real parameter. If $\phi \in [0,1]$, then (3.17) is called the Broadden convex update class. It is obvious that Broyden's class (3.17) satisfies the quasi-Newton equation (3.4). Also, the expression (3.17) can be rewritten in the following form

$$\begin{aligned} H_{k+1}^{\phi} &= H_{k+1}^{\mathrm{DFP}} + \phi\upsilon_k\upsilon_k^T \\ &= H_{k+1}^{\mathrm{BFGS}} + (\phi-1)\upsilon_k\upsilon_k^T \\ &= H_k + \frac{\mathbf{s}_k\mathbf{s}_k^T}{\mathbf{s}_k^T\mathbf{y}_k} - \frac{H_k\mathbf{y}_k\mathbf{y}_k^T H_k}{\mathbf{y}_k^T H_k\mathbf{y}_k} + \phi\upsilon_k\upsilon_k^T, \end{aligned} \tag{3.18}$$

where

$$\upsilon_k = (\mathbf{y}_k^T H_k\mathbf{y}_k)^{1/2}\left[\frac{\mathbf{s}_k}{\mathbf{s}_k^T\mathbf{y}_k} - \frac{H_k\mathbf{y}_k}{\mathbf{y}_k^T H_k\mathbf{y}_k}\right]. \tag{3.19}$$

If we put in (3.18):

- $\phi = 0$, then we will obtain DFP update (3.12)
- $\phi = 1$, then we will obtain BFGS update (3.14)
- $\phi = \frac{\mathbf{s}_k^{\mathrm{T}}\mathbf{y}_k}{(\mathbf{s}_k - H_k\mathbf{y}_k)^T\mathbf{y}_k}$, then we will obtain SR1 update (3.9).

The Broyden class of methods can be derived directly from the quasi-Newton equation. Consider the general formula for updating rank 2, which contains $\mathbf{s}_k$ and $H_k\mathbf{y}_k$:

$$H_{k+1} = H_k + a\mathbf{s}_k\mathbf{s}_k^{\mathrm{T}} + b(H_k\mathbf{y}_k\mathbf{s}_k^{\mathrm{T}} + \mathbf{s}_k\mathbf{y}_k^{\mathrm{T}}H_k) + cH_k\mathbf{y}_k`y_k^{\mathrm{T}}H_k, \tag{3.20}$$

where a, b, c are unknown scalars. We obtain

$$\begin{aligned} 1 &= a\mathbf{s}_k^{\mathrm{T}}\mathbf{y}_k + b\mathbf{y}_k^{\mathrm{T}}H_k\mathbf{y}_k, \\ 0 &= 1 + b\mathbf{s}_k^{\mathrm{T}}\mathbf{y}_k + c\mathbf{y}_k^{\mathrm{T}}H_k\mathbf{y}_k. \end{aligned} \tag{3.21}$$

Here we have two equations with three unknowns, so we can introduce the replacement

$$b = -\phi/\mathbf{s}_k^T\mathbf{y}_k,$$

where ϕ is a parameter. Solving system (3.21) and substituting the obtained result in (3.20), we obtain

$$H_{k+1}^{\phi} = H_k + \frac{\mathbf{s}_k\mathbf{s}_k^{\mathrm{T}}}{\mathbf{s}_k^T\mathbf{y}_k} - \frac{H_k\mathbf{y}_k\mathbf{y}_k^{\mathrm{T}}H_k}{\mathbf{y}_k^{\mathrm{T}}H_k\mathbf{y}_k} + \phi\upsilon_k\upsilon_k^{\mathrm{T}} = H_{k+1}^{\mathrm{DFP}} + \phi\upsilon_k\upsilon_k^{\mathrm{T}},$$

where υ_k is defined by means of (3.19). Previous expression is identical to (3.18).

3.5. **Some modifications of quasi-Newton methods.** A great effort has been invested to discover QN methods that do not merely possess convergence but it is also better from the BFGS update in the numerical performance. Table 1 shows some of these modifications of the quasi-Newton equations.

TABLE 1. Some modifications of quasi-Newton equations.

Quasi-Newton Eqs.	$\tilde{\mathbf{y}}_{k-1}$	**Ref.**
$B_k\mathbf{s}_{k-1}=\tilde{\mathbf{y}}_{k-1}$	$\tilde{\mathbf{y}}_{k-1}=\varphi_{k-1}\mathbf{y}_{k-1}+(1-\varphi_{k-1})B_{k-1}\mathbf{s}_{k-1}$	[104]
$B_k\mathbf{s}_{k-1}=\tilde{\mathbf{y}}_{k-1}$	$\tilde{\mathbf{y}}_{k-1}=\mathbf{y}_{k-1}+t_{k-1}\mathbf{s}_{k-1}, t_{k-1}\leq 10^{-6}$	[72]
$B_k\mathbf{s}_{k-1}=\tilde{\mathbf{y}}_{k-1}$	$\tilde{\mathbf{y}}_{k-1}=\mathbf{y}_{k-1}+\frac{2(f_{k-1}-f_k)+(\mathbf{g}_k+\mathbf{g}_{k-1})^{\mathrm{T}}\mathbf{s}_{k-1}}{\|\mathbf{s}_{k-1}\|^2}\mathbf{s}_{k-1}$	[123]
$B_k\mathbf{s}_{k-1}=\tilde{\mathbf{y}}_{k-1}$	$\tilde{\mathbf{y}}_{k-1}=\mathbf{y}_{k-1}+\frac{\max(0,2(f_{k-1}-f_k)+(\mathbf{g}_k+\mathbf{g}_{k-1})^{\mathrm{T}}\mathbf{s}_{k-1})}{\|\mathbf{s}_{k-1}\|^2}\mathbf{s}_{k-1}$	[133]
$B_k\mathbf{s}_{k-1}=\tilde{\mathbf{y}}_{k-1}$	$\tilde{\mathbf{y}}_{k-1}=\mathbf{y}_{k-1}+\frac{\max(0,6(f_{k-1}-f_k)+3(\mathbf{g}_k+\mathbf{g}_{k-1})^{\mathrm{T}}\mathbf{s}_{k-1})}{\|\mathbf{s}_{k-1}\|^2}\mathbf{s}_{k-1}$	[134]
$B_k\mathbf{s}_{k-1}=\tilde{\mathbf{y}}_{k-1}$	$\tilde{\mathbf{y}}_{k-1}=\frac{1}{2}\mathbf{y}_{k-1}+\frac{(f_{k-1}-f_k)-\frac{1}{2}\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{s}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}\mathbf{y}_{k-1}$	[59]

Also, it is important to state spectral gradient method. Therein the updating of the formula for B_{k+1} is done in the following way [111]

$$B_{k+1}=\operatorname{diag}(r_k^{(i)}) \tag{3.22}$$

with

$$r_k^{(i)}=\frac{1}{1+\frac{\sum_{i=1}^n(\hat{s}_k^{(i)})^2-\sum_{i=1}^n(\hat{\mathbf{s}}_k^{(i)})(\hat{\mathbf{y}}_k^{(i)})}{\sum_{i=1}^n(\hat{\mathbf{s}}_k^{(i)})^4}(\hat{\mathbf{s}}_k^{(i)})^2},\ \hat{\mathbf{s}}_k=\mathbf{s}_k-\frac{7}{11}\mathbf{s}_{k-1}+\frac{2}{11}\mathbf{s}_{k-2},$$

$$\hat{\mathbf{y}}_k=\mathbf{y}_k-\frac{7}{11}\mathbf{y}_{k-1}+\frac{2}{11}\mathbf{y}_{k-2}.$$

The general framework of the QN algorithm is given in Algorithm 3.

Algorithm 3 General framework of the quasi-Newton algorithm

Require: Objective $f(\mathbf{x})$, initial point $\mathbf{x}_0\in\mathbb{R}^n$, initial inverse Hessian approximation $H_0\in\mathbb{R}^{n\times n}$, and the tolerance $0<\varepsilon\ll 1$.

1: $k:=0$.
2: **while** $\|\mathbf{g}_k\|>\varepsilon$ **do**
3: Compute

$$\mathbf{d}_k=-H_k\mathbf{g}_k. \tag{3.23}$$

4: Compute $\alpha_k>0$ using exact or inexact line search.
5: Compute $\mathbf{x}_{k+1}:=\mathbf{x}_k+\alpha_k\mathbf{d}_k$.
6: Update H_k into H_{k+1}, such that (3.5) holds.
7: $k:=k+1$.
8: **end while**

Ensure: $\mathbf{x}_{k+1}, f(\mathbf{x}_{k+1})$.

3.6. **QN methods based on constant diagonal matrix approximation.** The general QN iterative rule with line search

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k H_k \, \mathbf{g}_k \tag{3.24}$$

assumes that B_k (resp. $H_k = B_k^{-1}$) is a positive definite approximation of G_k (resp. of G_k^{-1}) [118]. The update B_{k+1} of B_k is defined on the basis of the quasi-Newton property (3.4) or (3.5).

According to Brezinski's classification in [15], the structure of updating B_k can be divided into three categories: scalar matrix $B_k = \lambda_k I$, diagonal matrix $B_k = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$ and an appropriate full matrix B_k. Optimization methods included in this class of iterations are based on simplest approximation of the Hessian and its inverse as

$$B_k = \gamma_k I \approx G_k, \ \gamma_k > 0, \tag{3.25}$$

where I is a proper $n \times n$ identity matrix and and $\gamma_k > 0$ is a parameter. Such choice leads to the iterative rule

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \gamma_k^{-1} \alpha_k \mathbf{g}_k. \tag{3.26}$$

Usually, the parameter α_k is defined using an available ILS, and γ_k is defined according to the Taylor's development of $f(\mathbf{x})$. The iterations (3.26) are termed as *improved gradient descent* (IGD) methods in [63].

Andrei in [4, 6] defined iterations

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \theta_k \alpha_k \mathbf{g}_k. \tag{3.27}$$

Usage of random values of θ_k was proposed in [6]. Later, Andrei in [4] proposed appropriate algorithm for defining θ_k in (3.27). The iterative rule (3.27) was called in [4] as *Accelerated Gradient Descent* (AGD):

$$\mathbf{x}_{k+1}^{\mathrm{AGD}} = \mathbf{x}_k^{\mathrm{AGD}} - \theta_k^{\mathrm{AGD}} \alpha_k \mathbf{g}_k^{\mathrm{AGD}}. \tag{3.28}$$

A few modifications of the scheme (3.26) were promoted in [94, 96, 97, 112, 114]. The iterations defined in [112] are of the form (3.26), in which $\gamma_k I$ is the Hessian approximation, where $\gamma_k = \gamma(\mathbf{x}_k, \mathbf{x}_{k-1}) > 0$ is the parameter. The SM method from [112] was defined by the iterations

$$\mathbf{x}_{k+1}^{\mathrm{SM}} = \mathbf{x}_k^{\mathrm{SM}} - \alpha_k \left(\gamma_k^{\mathrm{SM}}\right)^{-1} \mathbf{g}_k^{\mathrm{SM}}, \tag{3.29}$$

where $\gamma_k^{\mathrm{SM}} > 0$ is the acceleration parameter defined using the Taylor's development of the objective f at the point $\mathbf{x}_{k+1}^{\mathrm{SM}}$, as follows:

$$\gamma_{k+1}^{\mathrm{SM}} = 2\gamma_k^{\mathrm{SM}} \frac{\gamma_k^{\mathrm{SM}} \left[f(\mathbf{x}_{k+1}^{\mathrm{SM}}) - f(\mathbf{x}_k^{\mathrm{SM}})\right] + \alpha_k \|\mathbf{g}_k^{\mathrm{SM}}\|^2}{\alpha_k^2 \|\mathbf{g}_k^{\mathrm{SM}}\|^2}. \tag{3.30}$$

The *Double direction* and *double step-size* accelerated methods, termed as ADSS and ADD, respectively, were originated in [94, 96].

The next iterations are known as *Accelerated double step-size* (ADSS) iterations [94]:

$$\mathbf{x}_{k+1}^{\mathrm{ADSS}} = \mathbf{x}_k^{\mathrm{ADSS}} - \left(\alpha_k (\gamma_k^{\mathrm{ADSS}})^{-1} + l_k\right) \mathbf{g}_k^{\mathrm{ADSS}}, \tag{3.31}$$

where α_k and l_k are step-sizes, derived by two independent backtracking procedures. The TADSS method from [114] is proposed using the assumption $\alpha_k + l_k = 1$, which gives

$$\mathbf{x}_{k+1}^{\mathrm{TADSS}} = \mathbf{x}_k^{\mathrm{TADSS}} - \psi_k \, \mathbf{g}_k^{\mathrm{TADSS}},$$

where $\psi_k = \alpha_k \left((\gamma_k^{\mathrm{TADSS}})^{-1} - 1\right) + 1$. An application of the TADSS iterations in aviation industry was investigated in [68].

The particular choice $\gamma_k = 1$ transforms the IGD iterations (3.26) into the GD iterative rule (1.5). Further, the IGD iterations (3.26) in the case $\alpha_k = 1$ can be viewed as the GD iterations

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \gamma_k^{-1}\mathbf{g}_k, \tag{3.32}$$

where γ_k becomes the primary step length which should be appropriately defined. Barzilai and Borwein in [11] originated two well known variants of the GD method, known as BB method, with the step length $\gamma_k^{BB} := \gamma_k^{-1}$ in (3.32). The step length γ_k^{BB} in the first case is defined by the vector minimization $\min_\gamma \|\mathbf{s}_{k-1} - \gamma\mathbf{y}_{k-1}\|^2$, which yields

$$\gamma_k^{\mathrm{BB}} = \frac{\mathbf{s}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}. \tag{3.33}$$

The symmetric case assumes the minimization $\|\gamma\mathbf{s}_{k-1} - \mathbf{y}_{k-1}\|^2$, which produces

$$\gamma_k^{\mathrm{BB}} = \frac{\mathbf{s}_{k-1}^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{s}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}. \tag{3.34}$$

The BB iterations are defined using γ_k^{BB} as follows:

$$\mathbf{x}_{k+1}^{\mathrm{BB}} = \mathbf{x}_k^{\mathrm{BB}} - \gamma_k^{\mathrm{BB}}\mathbf{g}_k^{\mathrm{BB}}.$$

The BB method was improved in a number of research articles, main of which are [22, 24, 34, 35, 36, 37, 106, 107, 122, 137].

Another member of the IGD iterates is the *Scalar Correction* (SC) method [84], defined in (3.32) by the rule

$$\gamma_{k+1}^{\mathrm{SC}} = \begin{cases} \frac{\mathbf{s}_k^{\mathrm{T}}\mathbf{r}_k}{\mathbf{y}_k^{\mathrm{T}}\mathbf{r}_k}, & \mathbf{y}_k^{\mathrm{T}}\mathbf{r}_k > 0, \\ \frac{\|\mathbf{s}_k\|}{\|\mathbf{y}_k\|}, & \mathbf{y}_k^{\mathrm{T}}\mathbf{r}_k \le 0, \end{cases} \qquad \mathbf{r}_k = \mathbf{s}_k - \gamma_k\mathbf{y}_k. \tag{3.35}$$

Accordingly, the SC iterations are defined by the relation

$$\mathbf{x}_{k+1}^{\mathrm{SC}} = \mathbf{x}_k^{\mathrm{SC}} - \gamma_k^{\mathrm{SC}}\mathbf{g}_k^{\mathrm{SC}}.$$

Relaxed BB method by an additional step $\theta_k \in (0, 2)$ is proposed in [105].

A modification of GD method (1.5) was proposed in [63]. It is defined by MGD = $\mathcal{M}$(GD) with

$$\mathbf{x}_{k+1} = \mathcal{M}(\mathrm{GD})(\mathbf{x}_k) = \mathbf{x}_k - \left(\alpha_k + \alpha_k^2 - \alpha_k^3\right)\mathbf{g}_k. \tag{3.36}$$

Further, the next scheme was proposed as the modified SM (MSM) method in [63]:

$$\mathbf{x}_{k+1}^{\mathrm{MSM}} = \mathbf{x}_k^{\mathrm{MSM}} - (\alpha_k + \alpha_k^2 - \alpha_k^3)(\gamma_k^{\mathrm{MSM}})^{-1}\mathbf{g}_k^{\mathrm{MSM}}. \tag{3.37}$$

The leading principle used in defining the iterations (3.37) is the replacement of α_k in the *GD* methods (1.5) by the slightly longer step $\alpha_k^{\mathrm{MSM}} = \alpha_k + \alpha_k^2 - \alpha_k^3$. The underlying idea in defining α_k^{MSM} is the observation $\alpha_k^{\mathrm{MSM}} > \alpha_k$, which means that MSM method proposes a slightly longer step with the aim to additionally accelerate the method. As before, $\alpha_k \in (0, 1)$ is defined by Algorithm 2. The rationale of this modification lies in the inequalities

$$\alpha_k \le \alpha_k + \alpha_k^2 - \alpha_k^3 \le \alpha_k + \alpha_k^2.$$

So, (3.37) is based on a small increase of α_k inside the interval $[\alpha_k, \alpha_k + \alpha_k^2]$.

The acceleration parameter γ_{k+1} in ADD, ADSS, TADSS and MSM methods are defined, respectively, as:

$$\gamma_{k+1}^{\mathrm{ADD}} = 2\frac{f(\mathbf{x}_{k+1}^{\mathrm{ADD}}) - f(\mathbf{x}_k^{\mathrm{ADD}}) - \alpha_k(\mathbf{g}_k^{\mathrm{ADD}})^{\mathrm{T}}\left(\alpha_k\mathbf{d}_k^{\mathrm{ADD}} - (\gamma_k^{\mathrm{ADD}})^{-1}\mathbf{g}_k^{\mathrm{ADD}}\right)}{\left(\alpha_k\mathbf{d}_k^{\mathrm{ADD}} - (\gamma_k^{\mathrm{ADD}})^{-1}\mathbf{g}_k^{\mathrm{ADD}}\right)^{\mathrm{T}}\left(\alpha_k\mathbf{d}_k^{\mathrm{ADD}} - (\gamma_k^{\mathrm{ADD}})^{-1}\mathbf{g}_k^{\mathrm{ADD}}\right)}, \quad \text{(ADD method [96])}$$

$$\gamma_{k+1}^{\mathrm{ADSS}} = 2\frac{f(\mathbf{x}_{k+1}^{\mathrm{ADSS}}) - f(\mathbf{x}_k^{\mathrm{ADSS}}) + \left(\alpha_k(\gamma_k^{\mathrm{ADSS}})^{-1} + l_k\right)\|\mathbf{g}_k^{\mathrm{ADSS}}\|^2}{\left(\alpha_k(\gamma_k^{\mathrm{ADSS}})^{-1} + l_k\right)^2\|\mathbf{g}_k^{\mathrm{ADSS}}\|^2}, \quad \text{(ADSS method [94])}$$

$$\gamma_{k+1}^{\mathrm{TADSS}} = 2\frac{f(\mathbf{x}_{k+1}^{\mathrm{TADSS}}) - f(\mathbf{x}_k^{TADSS}) + \psi_k\|\mathbf{g}_k^{\mathrm{TADSS}}\|^2}{\psi_k^2\|\mathbf{g}_k^{\mathrm{TADSS}}\|^2},$$
$$\psi_k = \alpha_k\left((\gamma_k^{\mathrm{TADSS}})^{-1} - 1\right) + 1, \quad \text{(TADSS method [114])}$$

$$\gamma_{k+1}^{\mathrm{MSM}} = 2\gamma_k^{\mathrm{MSM}}\frac{\gamma_k^{\mathrm{MSM}}\left[f(\mathbf{x}_{k+1}^{\mathrm{MSM}}) - f(\mathbf{x}_k^{\mathrm{MSM}})\right] + (\alpha_k + \alpha_k^2 - \alpha_k^3)\|\mathbf{g}_k^{\mathrm{MSM}}\|^2}{(\alpha_k + \alpha_k^2 - \alpha_k^3)^2\|\mathbf{g}_k^{\mathrm{MSM}}\|^2}, \quad \text{(MSM method [63]).}$$

The efficiency of IGD methods was numerically tested in [98].

The author of [41] proposed two *Relaxed Gradient Descent Quasi Newton* (RGDQN and RGDQN1) iteration rules

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \xi_k\alpha_k\,\gamma_k^{-1}\mathbf{g}_k, \tag{3.38}$$

such that ξ_k is a proper real value. The RGDQN iterations are defined with randomly generated $\xi_k \in (0,1)$, while the RGDQN1 algorithm exploits

$$\xi_k = \frac{\gamma_k}{\alpha_k\,\gamma_{k+1}}.$$

The following algorithm is known as the SM method and introduced in [112].

Algorithm 4 The SM method.

Require: Objective function $f(\mathbf{x})$ and chosen initial point $\mathbf{x}_0 \in \mathrm{dom}(f)$.
1: Set $k = 0$ and compute $f(\mathbf{x}_0)$, $\mathbf{g}_0 = \nabla f(\mathbf{x}_0)$ and take $\gamma_0 = 1$.
2: If stopping criteria are satisfied, then STOP; otherwise, go to the next step.
3: Find the step-size $\alpha_k \in (0,1]$ using Algorithm 2 with $\mathbf{d}_k = -\gamma_k^{-1}\mathbf{g}_k$.
4: Compute $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k\gamma_k^{-1}\mathbf{g}_k$, $f(\mathbf{x}_{k+1})$ and $\mathbf{g}_{k+1} = \nabla f(\mathbf{x}_{k+1})$.
5: Define γ_{k+1}using (3.30).
6: If $\gamma_{k+1} < 0$, then put $\gamma_{k+1} = 1$.
7: Set $k := k + 1$, go to the Step 2.
8: Return $\mathbf{x}_{k+1}$ and $f(\mathbf{x}_{k+1})$.

3.7. **Gradient methods accelerated by Picard-Mann hybrid iterative process.** An application of the Picard-Mann hybrid iterative process from [66] is another possibility to accelerate iterations for solving nonlinear optimization problems. The function $T : \mathbb{C} \to \mathbb{C}$ in (3.40) is defined on a convex subset $\mathbb{C}$ of a normed space $\mathbb{E}$. The hybrid iterations define two sequences $\mathbf{x}_k$, $\mathbf{y}_k$ by the rules:

$$\begin{cases} \mathbf{x}_1 = \mathbf{x} \in \mathbb{C}, \\ \mathbf{x}_{k+1} = T\mathbf{y}_k, \\ \mathbf{y}_k = (1 - \Upsilon_k)\mathbf{x}_k + \Upsilon_k T\mathbf{x}_k, \ k \in \mathbb{N}. \end{cases} \tag{3.39}$$

The real number $\Upsilon_k \in (0,1)$ from (3.39) is termed as the *correction parameter* in [97]. Instead of (3.39), it suffices to use an equivalent iteration given by

$$\mathbf{x}_{k+1} = \mathcal{H}(T)(\mathbf{x}_k) = T[(1 - \Upsilon_k)\mathbf{x}_k + \Upsilon_k T\mathbf{x}_k], \ k \in \mathbb{N}. \tag{3.40}$$

The iterates (3.40) are denoted by $\mathcal{H}(T,\mathbf{x}_k)=H(T)(\mathbf{x}_k)$.

In [66] it was proposed a set of constant values $\alpha=\Upsilon_k\in(0,1),\forall k$ in numerical experiments and concluded that the process (3.40) converges faster than the Picard, Mann and Ishikawa iterations from [62, 83, 101]. Further, (3.40) was applied in [97] for a hybridization of the SM method, known as HSM. Using the mapping T in (3.39) or (3.40) to coincide with the SM rule (3.29):

$$T(\mathbf{x}_k):=\mathbf{x}_k^{\mathrm{SM}}-\left(\gamma_k^{\mathrm{SM}}\right)^{-1}\alpha_k\mathbf{g}_k^{\mathrm{SM}},$$

the iterations (3.40) become the so called HSM iterative rule given as

$$\mathbf{x}_{k+1}^{\mathrm{HSM}}:=\mathcal{H}(\mathrm{SM})(\mathbf{x}_k)=\mathbf{x}_k^{\mathrm{HSM}}-(\Upsilon_k+1)\left(\gamma_k^{\mathrm{HSM}}\right)^{-1}\alpha_k\mathbf{g}_k^{\mathrm{HSM}},\tag{3.41}$$

where $\gamma_k^{\mathrm{HSM}}>0$ is the acceleration defined by

$$\gamma_{k+1}^{\mathrm{HSM}}=2\gamma_k^{\mathrm{HSM}}\frac{\gamma_k^{\mathrm{HSM}}\left[f(\mathbf{x}_{k+1}^{\mathrm{HSM}})-f(\mathbf{x}_k^{\mathrm{HSM}})\right]+(\Upsilon_k+1)t_k\|\mathbf{g}_k^{\mathrm{HSM}}\|^2}{(\Upsilon_k+1)^2\,t_k^2\,\|\mathbf{g}_k^{\mathrm{HSM}}\|^2}.$$

A modified HSM (MHSM) method is defined in [92] by proposing an appropriate initial value in the backtracking procedure.

A hybridization of the ADD method was considered in [99] in the form

$$\mathbf{x}_{k+1}^{\mathrm{HADD}}=\mathbf{x}_k^{\mathrm{HADD}}-(\Upsilon_k+1)t_k(\gamma_k^{\mathrm{HADD}})^{-1}\mathbf{g}_k^{\mathrm{HADD}}+(\Upsilon_k+1)t_k^2\mathbf{d}_k,$$

wherein

$$\gamma_{k+1}^{\mathrm{HADD}}=2\frac{f(\mathbf{x}_{k+1}^{\mathrm{HADD}})-f(\mathbf{x}_k^{\mathrm{HADD}})-(\Upsilon_k+1)(\mathbf{g}_k^{\mathrm{HADD}})^{\mathrm{T}}\left(t_k^2\mathbf{d}_k-t_k(\gamma_k^{\mathrm{HADD}})^{-1}\mathbf{g}_k^{\mathrm{HADD}}\right)}{(\Upsilon_k+1)^2t_k^2\left(t_k\mathbf{d}_k-(\gamma_k^{\mathrm{HADD}})^{-1}\mathbf{g}_k^{\mathrm{HADD}}\right)^{\mathrm{T}}\left(t_k\mathbf{d}_k-(\gamma_k^{\mathrm{HADD}})^{-1}\mathbf{g}_k^{\mathrm{HADD}}\right)}.$$

Recently, the hybridization HTADSS $\equiv\mathcal{H}(\mathrm{TADSS})$ was proposed, investigated and tested in [95].

4. **Overview of conjugate gradients methods.**

Nonlinear conjugate gradient (CG) methods form a class of important methods for solving unconstrained nonlinear optimization and solving system of nonlinear equations. Nonlinear CG methods are defined by the line search iterates (1.2) where the search direction $\mathbf{d}_k$ is defined by (1.10) and the CGUP β_k is given using one of many available rules.

In this article, a review on CG methods for unconstrained optimization is given. Main convergence theorems are provided for the conjugate gradient method assuming the descent property of each search direction. Some research issues on conjugate gradient methods are mentioned.

In [21], the nonlinear CG methods are divided into three classes: early conjugate gradient methods, descent conjugate gradient methods, and sufficient descent conjugate gradient methods. Andrei classified the CG methods in three classes: classical CG methods, scaled CG methods, and hybrid and parameterized CG methods.

The classification presented in this paper divides CG methods into the following categories: basic conjugate gradients methods, considered in Subsection 4.1, Dai-Liao class of methods, presented in Subsection 4.2, hybrid conjugate gradient methods, described in Subsection 4.3, and combined BFGS-CG iterations, in Subsection 4.4.

TABLE 2. Some modifications of quasi-Newton equations.

β_k	Title	Year	Reference
$\beta_k^{\mathrm{HS}} = \dfrac{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k}{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}}$	Hestenses–Stiefel	1952	[60]
$\beta_k^{\mathrm{FR}} = \dfrac{\|\mathbf{g}_k\|^2}{\|\mathbf{g}_{k-1}\|^2}$	Fletcher–Reeves	1964	[48]
$\beta_k^{\mathrm{D}} = \dfrac{\mathbf{g}_k^{\mathrm{T}}\mathbf{G}_{k-1}\mathbf{d}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{G}_{k-1}\mathbf{d}_{k-1}}$		1967	[38]
$\beta_k^{\mathrm{PRP}} = \dfrac{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k}{\|\mathbf{g}_{k-1}\|^2}$	Polak–Ribiere–Polyak	1969	[102, 103]
$\beta_k^{\mathrm{CD}} = -\dfrac{\|\mathbf{g}_k\|^2}{\mathbf{g}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}}$	Conjugate Descent	1987	[47]
$\beta_k^{\mathrm{LS}} = -\dfrac{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k}{\mathbf{g}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}}$	Liu–Storey	1991	[79]
$\beta_k^{\mathrm{DY}} = \dfrac{\|\mathbf{g}_k\|^2}{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}}$	Dai–Yuan	1999	[30]

4.1. **Basic conjugate gradients methods.** The CG methods included in Table 2 are known as early or classical conjugate gradient methods.

where $\mathbf{y}_{k-1} = \mathbf{g}_k - \mathbf{g}_{k-1}$, $\mathbf{s}_{k-1} = \mathbf{x}_k - \mathbf{x}_{k-1}$, $\mathbf{G}_{k-1} = \Delta^2 f(\mathbf{x}_{k-1})$ and $\|\cdot\|$ stands for the Euclidean vector norm.

In the listed CG methods, the numerator of the update parameter β_k is either $\|\mathbf{g}_k\|^2$ or $\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k$ and the denominator is either $\|\mathbf{g}_{k-1}\|^2$ or $\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}$ or $-\mathbf{g}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}$. Two possible choices for the numerator and the 3 possible choices for the denominator lead to 6 different choices for β_k.

TABLE 3. Classification of CG methods.

	Denominator		
Numerator	$\|\mathbf{g}_{k-1}\|^2$	$\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}$	$-\mathbf{g}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}$
$\|\mathbf{g}_k\|^2$	FR	DY	CD
$\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k$	PRP	HS	LS

Define the following functions

$$\mathfrak{n}_1 := \|\mathbf{g}_k\|^2, \quad \mathfrak{n}_2 = \mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k, \quad \mathfrak{d}_1 := \|\mathbf{g}_{k-1}\|^2, \quad \mathfrak{d}_2 = \mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}, \quad \mathfrak{d}_3 = -\mathbf{g}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}.$$

Then

$$\beta_k^{\mathrm{FR}} = \frac{\mathfrak{n}_1}{\mathfrak{d}_1}, \ \beta_k^{\mathrm{PRP}} = \frac{\mathfrak{n}_2}{\mathfrak{d}_1}, \ \beta_k^{\mathrm{DY}} = \frac{\mathfrak{n}_1}{\mathfrak{d}_2}, \ \beta_k^{\mathrm{HS}} = \frac{\mathfrak{n}_2}{\mathfrak{d}_2}, \ \beta_k^{\mathrm{CD}} = \frac{\mathfrak{n}_1}{\mathfrak{d}_3}, \ \beta_k^{\mathrm{LS}} = \frac{\mathfrak{n}_2}{\mathfrak{d}_3}.$$

But, there exist exceptions to these rules. One example is given in [38]

$$\beta_k^{\mathrm{D}} = \frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{G}_{k-1}\mathbf{d}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{G}_{k-1}\mathbf{d}_{k-1}} \quad (1967).$$

From the presented chronological development of the CGUP, we can see that the β_k^D choice of the CG parameter differs structurally from the other choices.

For a large-scale unconstrained nonlinear optimization problem, in practice, choices for updating a CG parameter that do not require computation or approximation of the Hessian and its inverse are preferred over methods that require Hessian or its approximation in each iteration.

Wei *et al.* [124] gave a variant of the PRP method which we call the VPRP method, with the parameter β_k

$$\beta_k^{\mathrm{VPRP}} = \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}}{\|\mathbf{g}_{k-1}\|^2}.$$

The VPRP method was extended to a variant of the HS method by Yao et al. in [132],

$$\beta_k^{\mathrm{VHS}} = \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}.$$

Zhang [138] took a little modification to the VPRP method and constructed the NPRP method as follows,

$$\beta_k^{\mathrm{NPRP}} = \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\|\mathbf{g}_{k-1}\|^2}.$$

Moreover, Zhang [138] extended this result to the HS method and proposed the NHS method as follows,

$$\beta_k^{\mathrm{NHS}} = \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}.$$

Recently, Wei et al. [125] proposed a variation of the FR method which we call the VFR method. the parameter β_k in the VFR method is given by

$$\beta_k^{\mathrm{VFR}} = \frac{\mu_1\|\mathbf{g}_k\|^2}{\mu_2|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}| + \mu_3\|\mathbf{g}_{k-1}\|^2},$$

where $\mu_1 \in (0, +\infty)$, $\mu_2 \in (\mu_1 + \epsilon_1, +\infty)$, $\mu_3 \in (0, +\infty)$ and ϵ_1 is an any given positive constant. Motivated by these modifications, in [29] the authors defined the modified PRP method as

$$\beta_k^{\mathrm{DPRP}} = \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\mu|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}| + \|\mathbf{g}_{k-1}\|^2}, \quad \mu > 1.$$

Recently, Wei et al. [18] gave a variant of the PRP method which we call the WYL method, that is,

$$\beta_k^{\mathrm{WYL}} = \frac{\mathbf{g}_k^{\mathrm{T}}\left(\mathbf{g}_k - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_{k-1}\right)}{\|\mathbf{g}_{k-1}\|^2}.$$

The WYL method was extended to a variant of the LS method by Yao et al. [132], that is,

$$\beta_k^{\mathrm{MLS}} = \frac{\mathbf{g}_k^{\mathrm{T}}\left(\mathbf{g}_k - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_{k-1}\right)}{-\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}}.$$

Also, the following function will be useful:

$$\mathfrak{N}_1 = \mathbf{g}_k^{\mathrm{T}}\left(\mathbf{g}_k - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_{k-1}\right) = \mathfrak{n}_1 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1} = \mathbf{g}_k^{\mathrm{T}}\widehat{\mathbf{y}_{k-1}}$$

$$\mathfrak{N}_2 = \mathfrak{n}_1 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}|.$$

Then

$$\beta_k^{\mathrm{WYL}} = \frac{\mathfrak{N}_1}{\mathfrak{d}_1}, \quad \beta_k^{\mathrm{VPRP}} = \frac{\mathfrak{N}_1}{\mathfrak{d}_1}, \quad \beta_k^{\mathrm{VHS}} = \frac{\mathfrak{N}_1}{\mathfrak{d}_2}$$
$$\beta_k^{\mathrm{NPRP}} = \frac{\mathfrak{N}_2}{\mathfrak{d}_1}, \quad \beta_k^{\mathrm{NHS}} = \frac{\mathfrak{N}_2}{\mathfrak{d}_2}.$$

Some particular CG variants are β_k^{DHS} [29] and β_k^{DLS} [139], defined by

$$\begin{aligned} \beta_k^{\mathrm{DHS}} &= \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\mu\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}\right| + \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \ (2012) \\ \beta_k^{\mathrm{DLS}} &= \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\mu\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}\right| - \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}}, \quad \mu > 1 \ (2017). \end{aligned} \tag{4.1}$$

If the functions

$$\mathfrak{D}_1(\mu) = \mu\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}\right| + \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}, \quad \mathfrak{D}_2(\mu) = \mu\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}\right| - \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}$$

are defined, then

$$\beta_k^{\mathrm{DHS}} = \frac{\mathfrak{N}_2}{\mathfrak{D}_1(\mu)}, \quad \beta_k^{\mathrm{DLS}} = \frac{\mathfrak{N}_2}{\mathfrak{D}_2(\mu)}.$$

4.2. **Dai-Liao method and its variants.** An extension of the conjugacy condition

$$\mathbf{d}_k^{\mathrm{T}}\mathbf{y}_{k-1} = 0 \tag{4.2}$$

was studied by Perry [93]. Perry tried to incorporate the second-order information of the objective function into the CG method to accelerate it. Specifically, by using the secant condition and the search direction of the QN methods, which are respectively defined by

$$B_k\mathbf{s}_{k-1} = \mathbf{y}_{k-1} \quad \text{and} \quad B_k\mathbf{d}_k = -\mathbf{g}_k, \tag{4.3}$$

the following relation is obtained:

$$\mathbf{d}_k^{\mathrm{T}}\mathbf{y}_{k-1} = \mathbf{d}_k^{\mathrm{T}}(B_k\mathbf{s}_{k-1}) = (B_k\mathbf{d}_k)^{\mathrm{T}}\mathbf{s}_{k-1} = -\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}, \tag{4.4}$$

where B_k is a symmetric approximation to the Hessian matrix G_k. Then Perry accordingly replaced the conjugacy condition (4.2) by the following condition

$$\mathbf{d}_k^{\mathrm{T}}\mathbf{y}_{k-1} = -\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}. \tag{4.5}$$

Furthermore, Dai and Liao [26] included a nonnegative parameter t into Perry's condition and gave

$$\mathbf{d}_k^{\mathrm{T}}\mathbf{y}_{k-1} = -t\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}. \tag{4.6}$$

In order to find the search direction $\mathbf{d}_k$ in (1.10) which satisfies the conjugacy condition (4.6), it suffices to multiply (1.10) by $\mathbf{y}_{k-1}$ and use (4.6), yielding

$$\beta_k^{\mathrm{DL}} = \frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{y}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} = \frac{\mathbf{g}_k^{\mathrm{T}}\left(\mathbf{y}_{k-1} - t\mathbf{s}_{k-1}\right)}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}, \quad t > 0. \tag{4.7}$$

Expression (4.7) for defining β_k characterizes the Dai and Liao (DL) CG method. Later, motivated by the DL method, researchers in papers [18, 80, 100, 129, 131, 139, 141], suggested modified variants of the DL method. Some well-known formulas for β_k were created modifying the CG parameter β_k^{DL} [18, 26, 80, 100, 129, 131,

139, 141]. Main of them are β_k^{DL} [26], β_k^{DHSDL} [139], β_k^{DLSDL} [139], β_k^{MHSDL} [131], defined as

$$\begin{aligned}\beta_k^{\mathrm{DL}} &= \frac{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k}{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \\ &= \beta_k^{\mathrm{HS}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}, \ (2001) \end{aligned} \tag{4.8}$$

$$\begin{aligned}\beta_k^{\mathrm{DHSDL}} &= \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\mu\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}\right| + \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \\ &= \beta_k^{\mathrm{DHS}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}, \ (2017) \end{aligned} \tag{4.9}$$

$$\begin{aligned}\beta_k^{\mathrm{DLSDL}} &= \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\mu\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}\right| - \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \\ &= \beta_k^{\mathrm{DLS}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}, \ (2017) \end{aligned} \tag{4.10}$$

$$\begin{aligned}\beta_k^{\mathrm{MHSDL}} &= \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \\ &= \frac{\mathbf{g}_k^{\mathrm{T}}\widehat{\mathbf{y}_{k-1}}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}, \ (2013) \end{aligned} \tag{4.11}$$

where $t > 0$ is a scalar and $\widehat{\mathbf{y}_{k-1}} = \mathbf{g}_k - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_{k-1}$.

Clearly β_k^{DL} with $t > 0$ defines a class of nonlinear CG methods. Moreover, in the case of the exact line search, i.e., $\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1} = 0$, then $\beta_k^{\mathrm{DL}} = \beta_k^{\mathrm{HS}}$.

Some additional CG methods from the DL class are β_k^{MLSDL} [18] and β_k^{ZZDL} [141], defined as follows:

$$\beta_k^{\mathrm{MLSDL}} = \frac{\mathbf{g}_k^{\mathrm{T}}\widehat{\mathbf{y}_{k-1}}}{-\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \tag{4.12}$$

$$\beta_k^{\mathrm{ZZDL}} = \frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{z}_{k-1}}{\mathbf{z}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{z}_{k-1}}, \tag{4.13}$$

where $t > 0$ is a scalar, $\mathbf{z}_{k-1} = \mathbf{y}_{k-1} + C\|\mathbf{g}_{k-1}\|^r\mathbf{s}_{k-1}$ and $\widehat{\mathbf{y}_{k-1}} = \mathbf{g}_k - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_{k-1}$.

In order to characterize this family of CG methods, define the mapping

$$\mathfrak{F}\left(\beta_k^{\mathcal{M}}, t\right) = \beta_k^{\mathcal{M}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}.$$

Then

$$\begin{aligned}\beta_k^{\mathrm{DL}} &= \mathfrak{F}\left(\beta_k^{\mathrm{HS}}, t\right), \quad \beta_k^{\mathrm{DHSDL}} = \mathfrak{F}\left(\beta_k^{\mathrm{DHS}}, t\right), \\ \beta_k^{\mathrm{DLSDL}} &= \mathfrak{F}\left(\beta_k^{\mathrm{DLS}}, t\right), \quad \beta_k^{\mathrm{MHSDL}} = \mathfrak{F}\left(\beta_k^{\mathrm{DHS}}, t\right).\end{aligned}$$

The research for the best values of the parameter t was divided into two directions. One direction was to find the best fixed value for t and the other the best approximation for t in each iteration. Analyzing the results from [18, 26, 131, 139], we conclude that the scalar t was defined by a fixed value of 0.1 in numerical experiments. Also, numerical experience related to the fixed valued $t = 1$ was reported in [26]. Common numerical experience is that different choices of t initiate totally

different numerical experience. This was the reason for further research to focus on the values of t that change through iterations. The value of t in arbitrary kth iteration will be denoted by t_k. Hager and Zhang in [55] defined t_k by the rule

$$t_k = 2\frac{\|\mathbf{y}_{k-1}\|^2}{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{s}_{k-1}}.$$

Babaie-Kafaki and Ghanbari [9] presented two appropriate choices of the parameter t in (4.7):

$$t_k = \frac{\mathbf{s}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}{\|\mathbf{s}_{k-1}\|^2} + \frac{\|\mathbf{y}_{k-1}\|}{\|\mathbf{s}_{k-1}\|}$$

and

$$t_k = \frac{\|\mathbf{y}_{k-1}\|}{\|\mathbf{s}_{k-1}\|}.$$

Andrei in [5] suggested the following value for t in (4.7) which becomes a variant of the DL method, denoted by DLE:

$$t_k = \frac{\mathbf{s}_{k-1}^{T}\mathbf{y}_{k-1}}{\|\mathbf{s}_{k-1}\|^2}. \tag{4.14}$$

Lotfi and Hosseini in [81] discovered the most recent approximations of the parameter t_k.

4.3. **Hybrid conjugate gradient methods.** In the subsequent sections, we will survey recent advances in CG methods. Two main research streams can be observed: the algorithms which improve the scalar parameter t and the algorithms which improve the CG parameter β_k.

Hybrid CG methods can be segmented into two classes: mixed methods as well as methods combined together by introducing one or more parameters.

The following hybrid CG method was suggested in [121]:

$$\beta_k = \begin{cases} \beta_k^{\mathrm{PRP}}, & \text{if } 0 \le \beta_k^{\mathrm{PRP}} \le \beta_k^{\mathrm{FR}}, \\ \beta_k^{\mathrm{FR}}, & \text{otherwise.} \end{cases} \tag{4.15}$$

When a jam in iterations occurs again, the PRP update parameter is used. Hu and Storey in [61] had a similar motivation and suggested the following rule

$$\beta_k = \max\{0, \min\{\beta_k^{\mathrm{PRP}}, \beta_k^{\mathrm{FR}}\}\}. \tag{4.16}$$

In [49] it is pointed out that β_k^{PRP} can be negative, even for strongly convex functions. In an effort to extend the allowed choices for the PRP update parameter, while retaining global convergence, Nocedal and Gilbert [49] suggested the choice

$$\beta_k = \max\{-\beta_k^{\mathrm{FR}}, \min\{\beta_k^{\mathrm{PRP}}, \beta_k^{\mathrm{FR}}\}\}. \tag{4.17}$$

Dai and Yuan [31] combined the DY method with other CG methods, which leads to the following CGUP parameters:

$$\beta_k = \max\{0, \min\{\beta_k^{\mathrm{HS}}, \beta_k^{\mathrm{DY}}\}\} \tag{4.18}$$

and

$$\beta_k = \max\{-c\,\beta_k^{\mathrm{DY}}, \min\{\beta_k^{\mathrm{HS}}, \beta_k^{\mathrm{DY}}\}\}, \quad c = \frac{1-\sigma}{1+\sigma}. \tag{4.19}$$

In [28], they tested different CG methods for large-scale unconstrained optimization problems and concluded that the hybrid CG method (4.18) gave the best results.

Next hybrid CG method, proposed by Dai in [23], employs either the DY scheme or the CD scheme:

$$\beta_k = \frac{\|\mathbf{g}_k\|^2}{\max\{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}, -\mathbf{g}_{k-1}^{\mathrm{T}}\mathbf{d}_{k-1}\}}. \tag{4.20}$$

A modified CG method defined as the hybridization of known LS and CD conjugate gradient methods is presented and analyzed in [130] by the rule

$$\beta_k^{\mathrm{LSCD}} = \max\{0, \min\{\beta_k^{\mathrm{LS}}, \beta_k^{\mathrm{CD}}\}\}. \tag{4.21}$$

CG methods can be combined together by introducing one or more parameters. In [32, 33], Dai and Yuan proposed a one-parameter family of CG methods with

$$\beta_k = \frac{\|\mathbf{g}_k\|^2}{\theta_k\|\mathbf{g}_{k-1}\|^2 + (1-\theta_k)\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}, \tag{4.22}$$

where $\theta_k \in [0,1]$ is a parameter. Note that $\beta_k = \beta_k^{\mathrm{FR}}$ in the case $\theta_k = 1$, and $\beta_k = \beta_k^{\mathrm{DY}}$ if $\theta_k = 0$.

Another hybrid method, proposed by Delladji, Belloufi and Sellami in [39], exploits either the PRP scheme or the HZ scheme, as

$$\begin{aligned}\beta_k^{\mathrm{hPRPHZ}} &= \theta_k\beta_k^{\mathrm{PRP}} + (1-\theta_k)\beta_k^{\mathrm{HZ}}\\ &= \theta_k\frac{\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k}{\|\mathbf{g}_{k-1}\|^2} + (1-\theta_k)\frac{1}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}\left(\mathbf{y}_{k-1} - 2\mathbf{d}_{k-1}\frac{\|\mathbf{y}_{k-1}\|^2}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}\right)^{\mathrm{T}}\mathbf{g}_k,\end{aligned} \tag{4.23}$$

in which $\theta_k \in [0,1]$ is called the hybridization parameter. Note that if $\theta_k = 1$ then $\beta_k^{\mathrm{hPRPHZ}} = \beta_k^{\mathrm{PRP}}$, and if $\theta_k = 0$ then $\beta_k^{\mathrm{hPRPHZ}} = \beta_k^{\mathrm{HZ}}$.

Nazareth in [89] proposed a two-parameter family of CGUP parameters using convex combinations of numerators and denominators, as

$$\beta_k = \frac{\nu_k\|\mathbf{g}_k\|^2 + (1-\nu_k)\mathbf{g}_k^{\mathrm{T}}\mathbf{y}_{k-1}}{\theta_k\|\mathbf{g}_{k-1}\|^2 + (1-\theta_k)\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}, \tag{4.24}$$

where $\nu_k, \theta_k \in [0,1]$. This two-parameter family includes FR, DY, PRP, and HS methods as extreme cases.

In [113], the authors proposed hybrid CG methods where the search direction $\mathbf{d}_k := \mathfrak{d}_k$, $k \geq 1$, is improved using one of the rules

$$\mathbf{d}_k := \mathfrak{D}(\beta_k, \mathbf{g}_k, \mathbf{d}_{k-1}) = -\left(1 + \beta_k\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}}{\|\mathbf{g}_k\|^2}\right)\mathbf{g}_k + \beta_k\mathbf{d}_{k-1}, \tag{4.25}$$

$$\mathbf{d}_k := \mathfrak{D}_1(\beta_k, \mathbf{g}_k, \mathbf{d}_{k-1}) = -B_k\mathbf{g}_k + \mathfrak{D}(\beta_k, \mathbf{g}_k, \mathbf{d}_{k-1}), \tag{4.26}$$

and β_k is determined using appropriate combinations of β_k used in Table 2 and/or previously defined hybridizations. In [113], the authors defined a modification of LSCD method, defined in [130] by

$$\begin{aligned}\beta_k^{\mathrm{LSCD}} &= \max\{0, \min\{\beta_k^{\mathrm{LS}}, \beta_k^{\mathrm{CD}}\}\},\\ \mathbf{d}_k &= \mathfrak{d}(\beta_k^{\mathrm{LSCD}}, \mathbf{g}_k, \mathbf{d}_{k-1}).\end{aligned} \tag{4.27}$$

The resulting method is known as the MLSCD method with the search direction

$$\mathbf{d}_k := \mathfrak{D}(\beta_k^{\mathrm{LSCD}}, \mathbf{g}_k, \mathbf{d}_{k-1}). \tag{4.28}$$

In general, the idea is based on the replacement of $\mathbf{d}_k = \mathfrak{d}_k(\beta_k^{\mathrm{LSCD}}, \mathbf{g}_k, \mathbf{d}_{k-1})$ from [130] by $\mathbf{d}_k = \mathfrak{D}(\beta_k^{\mathrm{LSCD}}, \mathbf{g}_k, \mathbf{d}_{k-1})$.

Now we give the general framework of the CG class of methods.

Algorithm 5 Algorithm of CG methods.

Require: A starting point x_0, and real quantities $0 < \epsilon < 1$, $0 < \delta < 1$.

1: Put $k = 0$ and compute $\mathbf{d}_0 = -\mathbf{g}_0$.
2: If
$$\|\mathbf{g}_k\| \leq \epsilon \quad \text{and} \quad \frac{|f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k)|}{1 + |f(\mathbf{x}_k)|} \leq \delta,$$
STOP; else go to Step 3.
3: Compute $\alpha_k \in (0, 1]$ using Algorithm 2.
4: Compute $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$.
5: Calculate $\mathbf{g}_{k+1}$, $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$ and go to Step 6.
6: Calculate β_{k+1}.
7: Compute the search direction $\mathbf{d}_{k+1} = \mathfrak{d}(\beta_{k+1}, \mathbf{g}_{k+1}, \mathbf{d}_k)$ or $\mathbf{d}_{k+1} = \mathfrak{D}(\beta_{k+1}, \mathbf{g}_{k+1}, \mathbf{d}_k)$.
8: Put $k \coloneqq k + 1$, and go to Step 2.

The first iteration in CG methods is a gradient step. Also, it is common to restart the algorithm periodically by taking the gradient step.

4.4. **Broyden-Fletcher-Goldfarb-Shanno conjugate gradient methods.** Known fact is that CG iterates are better than QN methods in terms of the CPU time. Moreover, BFGS updates require greater memory space usage than CG. On the other hand, the QN methods require lesser number of iterations as well as the number of function evaluations. For this purpose, one of the modern trends in defining new CG methods is usage of the BFGS update in defining new rules for defining β_k. A hybrid method which solves the system of nonlinear equations combining the QN method with chaos optimization was discovered in [82]. In [58], the authors defined a combination of a QN and the Cauchy descent method for solving unconstrained optimization problems, which is known as the quasi-Newton SD method. A hybrid direction defined as a combination of the BFGS update of B_k and the CGUP β_k was considered in [10, 67]. The DFP-CG method was originated in [91]. A three-term hybrid BFGS-CG method (termed as H-BFGS-CG1) was proposed in [113] by the search direction

$$\mathbf{d}_k := \begin{cases} -B_k \mathbf{g}_k, & k = 0, \\ \mathfrak{D}_1(\beta_k^{\mathrm{LSCD}}, \mathbf{g}_k, \mathbf{d}_{k-1}), & k \geq 1. \end{cases}$$

In [113], the authors investigated hybrid CG algorithms based on the modified search direction which is defined using one of the following two hybridizations:

$$\mathbf{d}_k := \mathfrak{D}(\beta_k, \mathbf{g}_k, \mathbf{d}_{k-1}) = -\left(1 + \beta_k \frac{\mathbf{g}_k^{\mathrm{T}} \mathbf{d}_{k-1}}{\|\mathbf{g}_k\|^2}\right) \mathbf{g}_k + \beta_k \mathbf{d}_{k-1}, \tag{4.29}$$

$$\mathbf{d}_k := \mathfrak{D}_1(\beta_k, \mathbf{g}_k, \mathbf{d}_{k-1}) = -B_k \mathbf{g}_k + \mathfrak{D}(\beta_k, \mathbf{g}_k, \mathbf{d}_{k-1}), \tag{4.30}$$

as well as on the usage of β_k defined using convenient combinations of the parameters β_k involved in Table 2 and previously defined hybridizations. The matrix B_k in (4.30) is defined as an appropriate Hessian approximation by the BFGS update. A three-term BFGS-CG method, known as H-BFGS-CG1, was defined in [113] using

$$\mathbf{d}_k := \begin{cases} -B_k \mathbf{g}_k, & k = 0, \\ \mathfrak{D}_1(\beta_k^{\mathrm{LSCD}}, \mathbf{g}_k, \mathbf{d}_{k-1}), & k \geq 1. \end{cases}$$

5. **Convergence properties of CG methods.** Below we give some assumptions related to line search procedures.

Assumption 5.1. (1) *The level set* $\mathcal{S}=\{\mathbf{x}\in\mathbb{R}^n|\ f(\mathbf{x})\leq f(\mathbf{x}_0)\}$ *is bounded, where* $\mathbf{x}_0$ *is an initial point of the iterative method* (1.2).

(2) *The objective function* f *is continuous and differentiable in a neighborhood* $\mathcal{N}$ *of* $\mathcal{S}$*, and its gradient* $\mathbf{g}$ *is Lipschitz continuous. So, there exists a positive constant* $L>0$*, satisfying*

$$\|\mathbf{g}(\mathbf{u})-\mathbf{g}(\mathbf{v})\|\leq L\|\mathbf{u}-\mathbf{v}\|,\ \forall u,v\in\mathcal{N}. \tag{5.1}$$

Assumption 5.1 initiates the existence of a positive constants D and γ satisfying

$$\|\mathbf{u}-\mathbf{v}\|\leq D,\ \forall\ \mathbf{u},\mathbf{v}\in\mathcal{N} \tag{5.2}$$

and

$$\|\mathbf{g}(\mathbf{u})\|\leq\gamma,\ \forall\ \mathbf{u}\in\mathcal{N}. \tag{5.3}$$

The proof of Lemma 5.1 is given in [142] and known as the Zoutendijk condition.

Lemma 5.1. [17, 142] *Let Assumption 5.1 be accomplished and the points* $\{\mathbf{x}_k\}$ *be generated by the method* (1.2) *and* (1.10)*. Then it holds*

$$\sum_{k=0}^{\infty}\frac{\|\mathbf{g}_k\|^4}{\|\mathbf{d}_k\|^2}<+\infty. \tag{5.4}$$

5.1. **Properties of CG methods with $\mathbf{y}_{k-1}^T\mathbf{g}_k$ in the numerator of β_k.** In this subsection, we recall properties of the HS, PRP and LS methods. If we look at the chronological development presented in Table 2, a clear observation is that HS, PRP and LS methods involve the expression $\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k$ in the numerator of the parameter β_k. We first mention the Property (*) for β_k given by Gilbert and Nocedal [49]. The Property (*) implies that β_k is bounded and small when the step $\mathbf{s}_{k-1}$ is small.

Property (*) [49] Let a method defined by (1.2) and (1.10) satisfies

$$0<\gamma\leq\|\mathbf{g}_k\|\leq\bar{\gamma}, \tag{5.5}$$

for all $k\geq 0$. Under this assumption we say that the method possesses the Property (*) if there exist constants $b>1$ and $\lambda>0$ such that for all k:

$$|\beta_k|\leq b, \tag{5.6}$$

and

$$\|\mathbf{s}_{k-1}\|\leq\lambda\Rightarrow|\beta_k|\leq\frac{1}{2b}. \tag{5.7}$$

In order to prove that conjugate gradient methods have Property (*), it suffices to show that there exists a constant $c>0$ such that

$$|\beta_k|\leq c\|\mathbf{s}_{k-1}\|\ \text{ for all }\ k, \tag{5.8}$$

under the assumption (5.5). Then, by putting $\lambda=\frac{1}{2bc}$, we have $|\beta_k|\leq\max\{1,2bc\}\equiv b$ and

$$\|\mathbf{s}_{k-1}\|\leq\lambda\Rightarrow|\beta_k|\leq\frac{1}{2b}, \tag{5.9}$$

which confirms the Property (*). It is easily shown that (5.8) holds for the HS, PRP and LS methods, and thus these methods have Property (*).

Next, we give the global convergence theorem of CG methods satisfying Property (*). The proof of Theorem 5.2 is given in [49].

Theorem 5.2. *Consider any conjugate gradient method* (1.2), (1.10) *that satisfies the following conditions:*
(*a*) $\beta_k \geq 0$.
(*b*) *The search directions satisfy the sufficient descent condition* (1.6)
(*c*) *The Zoutendijk condition holds.*
(*d*) *The Property (*) holds.*
If the Lipschitz and Boundedness Assumptions hold, then the iterates are globally convergent.

5.2. **Properties of CG methods involving $\|\mathbf{g}_k\|^2$ in the numerator of β_k.** In this section, we recall properties of the FR, CD and DY methods. If we look at the chronological development presented in Table 2, it is observable that FR, CD and DY methods involve the value $\|\mathbf{g}_k\|^2$ in the numerator of the parameter β_k. If the step-size α_k satisfies the generalized strong Wolfe conditions (2.7) and (2.9), the following properties are obtained.

Proposition 1. *The following statements hold:*
(*a*) *For the* FR *method, if α_k satisfies the generalized strong Wolfe conditions* (2.7) *and* (2.9) *with $\sigma_1 + \sigma_2 < 1$, then*

$$-\frac{1}{1-\sigma_1} \leq \frac{\mathbf{g}_k^T \mathbf{d}_k}{\|\mathbf{g}_k\|^2} \leq -1 + \frac{\sigma_2}{1-\sigma_1}. \tag{5.10}$$

(*b*) *For the* DY *method, if α_k satisfies the generalized strong Wolfe conditions* (2.7) *and* (2.9), *then*

$$-\frac{1}{1-\sigma_1} \leq \frac{\mathbf{g}_k^T \mathbf{d}_k}{\|\mathbf{g}_k\|^2} \leq -\frac{1}{1+\sigma_2}. \tag{5.11}$$

(*c*) *For the* CD *method, if α_k satisfies the generalized strong Wolfe conditions* (2.7) *and* (2.9) *with $\sigma_2 < 1$, then*

$$-1-\sigma_1 \leq \frac{\mathbf{g}_k^T \mathbf{d}_k}{\|\mathbf{g}_k\|^2} \leq -1 + \sigma_2. \tag{5.12}$$

Proposition 1 implies that the FR, CD and DY methods satisfy the sufficient descent condition (1.6), dependent on line searches.

We now give the global convergence properties of the FR and DY methods, which were proven in [2] and [30], respectively.

Theorem 5.3. *Suppose that Assumption 5.1 holds. Let the sequence $\{\mathbf{x}_k\}$ be generated by the conjugate gradient method of the form* (1.2)-(1.10).
(*a*) *If $\beta_k = \beta_k^{\mathrm{FR}}$ and α_k satisfies the generalized strong Wolfe conditions* (2.7) *and* (2.9) *with $\sigma_1 + \sigma_2 < 1$, then $\{\mathbf{x}_k\}$ converges globally with the limit*

$$\liminf_{k\to\infty} \|\mathbf{g}_k\| = 0. \tag{5.13}$$

(*b*) *If $\beta_k = \beta_k^{\mathrm{DY}}$ and α_k satisfies the Wolfe conditions* (2.7) *and* (2.8), *then $\{\mathbf{d}_k\}$ satisfies the descent condition* (1.4) *and $\{\mathbf{x}_k\}$ converges globally in the sense that* (5.13) *holds.*

Methods surveyed in Subsection 5.2 exhibit strong convergence properties, but they may not be efficient in practice due to appearance of jamming. On the other hand however, methods given in Subsection 5.1 may not be convergent in general, they often perform better than the methods restated in Subsection 5.2. Details about this fact are given in [65]. This clearly implies that combinations of these

methods have been proposed with the aim of exploiting attractive features of each family of methods.

5.3. **Convergence of DL methods.** The following assumptions will be commonly used in the subsequent convergence analysis of DHSDL, DLSDL, MHSDL and MLSDL methods.

It is supposed that the conditions in Assumption 5.1 hold. Assumption 5.1 initiates the existence of positive constants D and γ satisfying (5.2) and (5.3).

By the uniform convexity of f, there exists a constant $\theta > 0$ such that

$$(\mathbf{g}(\mathbf{u}) - \mathbf{g}(\mathbf{v}))^{\mathrm{T}}(\mathbf{u} - \mathbf{v}) \geq \theta \|\mathbf{u} - \mathbf{v}\|^2, \text{ for all } \mathbf{u}, \mathbf{v} \in S, \tag{5.14}$$

or equivalently,

$$f(\mathbf{u}) \geq f(\mathbf{v}) + \mathbf{g}(\mathbf{v})^{\mathrm{T}}(\mathbf{u} - \mathbf{v}) + \frac{\theta}{2}\|\mathbf{u} - \mathbf{v}\|^2, \text{ for all } \mathbf{u}, \mathbf{v} \in S. \tag{5.15}$$

It follows from (5.14) and (5.15) that

$$\mathbf{s}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1} \geq \theta \|\mathbf{s}_{k-1}\|^2 \tag{5.16}$$

and

$$f_{k-1} - f_k \geq -\mathbf{g}_k^T \mathbf{s}_{k-1} + \frac{\theta}{2}\|\mathbf{s}_{k-1}\|^2. \tag{5.17}$$

By (5.1) and (5.16), we have

$$\theta \|\mathbf{s}_{k-1}\|^2 \leq \mathbf{s}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1} \leq L\|\mathbf{s}_{k-1}\|^2, \tag{5.18}$$

where the inequalities imply $\theta \leq L$.

Further, (5.18) implies

$$\mathbf{s}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1} = \alpha_{k-1}\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1} > 0. \tag{5.19}$$

From $\alpha_{k-1} > 0$ and (5.19), it follows that

$$\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1} > 0. \tag{5.20}$$

In order to improve presentation, an arbitrary method defined by (1.2) and (1.10) will be denoted by $\mathcal{M}(\alpha_k, \beta_k)$. In our current investigation, it will be assumed $\beta_k \in \left\{\beta_k^{\mathrm{DHSDL}}, \beta_k^{\mathrm{DLSDL}}, \beta_k^{\mathrm{MHSDL}}, \beta_k^{\mathrm{MLSDL}}\right\}$. It is assumed that α_k satisfies the backtracking condition. Further, we will use the notation $\beta_k' \in \left\{\beta_k^{\mathrm{DHS}}, \beta_k^{\mathrm{DLS}}, \beta_k^{\mathrm{MHS}}, \beta_k^{\mathrm{MLS}}\right\}$. Clearly,

$$\beta_k = \beta_k' - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}. \tag{5.21}$$

Lemma 5.4. [17, 142] *Let Assumption 5.1 be accomplished and the points $\{\mathbf{x}_k\}$ be generated by the method $\mathcal{M}(\alpha_k, \beta_k)$. Then (5.4) holds*

Lemma 5.5. *The parameters β_k' in $\mathcal{M}(\alpha_k, \beta_k)$ satisfy*

$$0 \leq \beta_k' \leq \frac{\|\mathbf{g}_k\|^2}{\lambda|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}|}, \ \lambda \geq 1 \tag{5.22}$$

in each iterative step k.

Proof. In the case $\beta_k' \in \left\{\beta_k^{\mathrm{DHS}}, \beta_k^{\mathrm{DLS}}\right\}$ the inequalities

$$0 \leq \beta_k^{\mathrm{DHS}}, \quad \beta_k^{\mathrm{DLS}} \leq \frac{\|\mathbf{g}_k\|^2}{\mu|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}|}$$

are known from [29].

For $\beta'_k \equiv \beta_k^{\mathrm{MHS}}$, it is possible to verify

$$0 \leq \beta_k^{\mathrm{MHS}} = \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \leq \frac{\|\mathbf{g}_k\|^2}{\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}\right|}.$$

For $\beta'_k \equiv \beta_k^{\mathrm{MLS}}$, we have

$$0 \leq \beta_k^{\mathrm{MLS}} = \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{-\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}} \leq \frac{\|\mathbf{g}_k\|^2}{\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}\right|}.$$

Since $\mu > 1$ the proof is completed. □

Lemma 5.6. *The iterations $\mathcal{M}(\alpha_k, \beta_k)$ satisfy*

$$\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_k \leq -c\|\mathbf{g}_k\|^2 \tag{5.23}$$

for some $0 \leq c \leq 1$, all $k \geq 0$ and arbitrary β_k.

Proof. The inequality (5.23) will be verified by induction. In the initial situation $k = 0$, one obtains $\mathbf{g}_0^{\mathrm{T}}\mathbf{d}_0 = -\|\mathbf{g}_0\|^2$. Since $c \leq 1$, obviously (5.23) is satisfied in the basic case. Suppose that (5.23) is valid for some $k \geq 1$. By taking the inner product of the left and right hand side in (1.10) with the vector $\mathbf{g}_k^{\mathrm{T}}$, it can be obtained

$$\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_k = -\|\mathbf{g}_k\|^2 + \beta_k \mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}. \tag{5.24}$$

An application of (5.21) and $\mathbf{s}_{k-1} = \alpha_{k-1}\mathbf{d}_{k-1}$ leads to further conclusions:

$$\begin{aligned}
\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_k &= -\|\mathbf{g}_k\|^2 + \left(\beta'_k - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}\right)\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1} \\
&= -\|\mathbf{g}_k\|^2 + \beta'_k\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1} - t\frac{\alpha_{k-1}\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1} \\
&= -\|\mathbf{g}_k\|^2 + \beta'_k\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1} - t\frac{\alpha_{k-1}(\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1})^2}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}.
\end{aligned}$$

From (5.20), $t > 0$ and $\alpha_{k-1} > 0$, one obtains

$$t\frac{\alpha_{k-1}(\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1})^2}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} > 0. \tag{5.25}$$

Now (5.25) in conjunction with (5.22) implies

$$\begin{aligned}
\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_k &\leq -\|\mathbf{g}_k\|^2 + \beta'_k\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1} \\
&\leq -\|\mathbf{g}_k\|^2 + \frac{\|\mathbf{g}_k\|^2}{\lambda|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}|}|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}| \\
&\leq -\|\mathbf{g}_k\|^2 + \frac{\|\mathbf{g}_k\|^2}{\lambda} \\
&= -\left(1 - \frac{1}{\lambda}\right)\|\mathbf{g}_k\|^2.
\end{aligned}$$

In view of $\lambda \geq 1$, the inequality (5.23) is satisfied for $c = \left(1 - \frac{1}{\lambda}\right)$ and arbitrary $k \geq 0$. □

Lemma 5.7. *The parameter $\beta_k \in \{\beta_k^{\mathrm{DHSDL}}, \beta_k^{\mathrm{DLSDL}}, \beta_k^{\mathrm{MHSDL}}\}$ satisfies*

$$\beta_k \leq \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right| - t\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}. \tag{5.26}$$

Proof. For $\beta_k \equiv \beta_k^{\mathrm{DHSDL}}$, in view of (5.20), it follows that

$$\begin{aligned}\beta_k^{\mathrm{DHSDL}} &= \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\mu\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}\right| + \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \\ &\leq \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \\ &= \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right| - t\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}.\end{aligned} \tag{5.27}$$

As $\mu > 1$ we conclude

$$\begin{aligned}\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1} &= \mathbf{d}_{k-1}^{\mathrm{T}}(\mathbf{g}_k - \mathbf{g}_{k-1}) \\ &= \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_k - \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1} \\ &\leq \left|\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_k\right| - \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1} \\ &\leq \mu\left|\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_k\right| - \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}.\end{aligned} \tag{5.28}$$

Using inequalities (5.20) and (5.28) in $\beta_k \equiv \beta_k^{\mathrm{DLSDL}}$ one obtains

$$\begin{aligned}\beta_k^{\mathrm{DLSDL}} &= \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\mu\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}\right| - \mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \\ &\leq \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \\ &= \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right| - t\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}.\end{aligned} \tag{5.29}$$

The following conclusion is valid in the case $\beta_k \equiv \beta_k^{\mathrm{MHSDL}}$:

$$\begin{aligned}\beta_k^{\mathrm{MHSDL}} &= \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \\ &= \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right| - t\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}.\end{aligned} \tag{5.30}$$

Therefore, the inequality (5.26) follows from (5.27), (5.29) and (5.30). □

The global convergence of the proposed methods is confirmed by Theorem 5.8.

Theorem 5.8. *Assume that Assumption 5.1 is true and f is a uniformly convex function. Then the sequence $\{\mathbf{x}_k\}$ generated by the $\mathcal{M}(\alpha_k, \beta_k)$ method fulfils*

$$\liminf_{k\to\infty} \|\mathbf{g}_k\| = 0. \tag{5.31}$$

Proof. Assume that (5.31) is not true. This implies the existence of a constant $c_1 > 0$ such that

$$\|\mathbf{g}_k\| \geq c_1, \quad \text{for all } k. \tag{5.32}$$

Squaring both sides of (1.10) implies

$$\|\mathbf{d}_k\|^2 = \|\mathbf{g}_k\|^2 - 2\beta_k\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1} + (\beta_k)^2\|\mathbf{d}_{k-1}\|^2. \tag{5.33}$$

Taking into account (5.22), one can obtain

$$\begin{aligned} -2\beta_k \mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1} &= -2\left(\beta_k' - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}\right)\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1} \\ &= -2\left(\beta_k'\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1} - t\frac{\alpha_k(\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1})^2}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}\right). \end{aligned} \tag{5.34}$$

Now from (5.25), with respect to $t\frac{\alpha_{k-1}(\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1})^2}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} > 0$, the following inequalities hold:

$$\begin{aligned} -2\beta_k\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1} &\le 2|\beta_k'||\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}| \\ &\le 2\frac{\|\mathbf{g}_k\|^2}{\lambda|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}|}|\mathbf{g}_k^{\mathrm{T}}\mathbf{d}_{k-1}| \\ &\le 2\frac{\|\mathbf{g}_k\|^2}{\lambda}. \end{aligned} \tag{5.35}$$

Case 1. In the cases $\beta_k \in \{\beta_k^{\mathrm{DHSDL}}, \beta_k^{\mathrm{DLSDL}}, \beta_k^{\mathrm{MHSDL}}\}$ from Lemma 5.7, we conclude

$$\begin{aligned} \beta_k &\le \frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right| - t\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}} \\ &\le \left|\frac{\mathbf{g}_k^T\mathbf{g}_k - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right| - t\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}\right| \\ &\le \frac{\left|\mathbf{g}_k^T\left(\mathbf{g}_k - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_{k-1} - t\mathbf{s}_{k-1}\right)\right|}{\theta\alpha_{k-1}\|\mathbf{d}_{k-1}\|^2} \\ &= \frac{\left|\mathbf{g}_k^T\left(\mathbf{g}_k - \mathbf{g}_{k-1} + \mathbf{g}_{k-1} - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_{k-1} - t\mathbf{s}_{k-1}\right)\right|}{\theta\alpha_{k-1}\|\mathbf{d}_{k-1}\|^2} \\ &\le \frac{\|\mathbf{g}_k\|\left(\|\mathbf{g}_k - \mathbf{g}_{k-1}\| + \left\|\mathbf{g}_{k-1}\left(1 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\right)\right\| + t\|\mathbf{s}_{k-1}\|\right)}{\theta\alpha_{k-1}\|\mathbf{d}_{k-1}\|^2} \\ &\le \frac{\|\mathbf{g}_k\|\left(\|\mathbf{g}_k - \mathbf{g}_{k-1}\| + \left|1 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\right|\|\mathbf{g}_{k-1}\| + t\|\mathbf{s}_{k-1}\|\right)}{\theta\alpha_{k-1}\|\mathbf{d}_{k-1}\|^2} \\ &\le \frac{\|\mathbf{g}_k\|\left(\|\mathbf{g}_k - \mathbf{g}_{k-1}\| + |\|\mathbf{g}_{k-1}\| - \|\mathbf{g}_k\|| + t\|\mathbf{s}_{k-1}\|\right)}{\theta\alpha_{k-1}\|\mathbf{d}_{k-1}\|^2} \\ &\le \frac{\|\mathbf{g}_k\|\left(\|\mathbf{g}_k - \mathbf{g}_{k-1}\| + \|\mathbf{g}_{k-1} - \mathbf{g}_k\| + t\|\mathbf{s}_{k-1}\|\right)}{\theta\alpha_{k-1}\|\mathbf{d}_{k-1}\|^2} \end{aligned} \tag{5.36}$$

So,

$$\begin{aligned} \beta_k &\le \frac{\|\mathbf{g}_k\|\left(2\|\mathbf{g}_k - \mathbf{g}_{k-1}\| + t\|\mathbf{s}_{k-1}\|\right)}{\theta\alpha_{k-1}\|\mathbf{d}_{k-1}\|^2} \\ &\le \frac{\|\mathbf{g}_k\|\left(2L\|\mathbf{s}_{k-1}\| + t\|\mathbf{s}_{k-1}\|\right)}{\theta\alpha_{k-1}\|\mathbf{d}_{k-1}\|^2} \\ &\le \frac{(2L+t)\|\mathbf{g}_k\|\|\mathbf{s}_{k-1}\|}{\theta\alpha_{k-1}\|\mathbf{d}_{k-1}\|^2} \end{aligned} \tag{5.37}$$

$$= \frac{(2L+t)\,\|\mathbf{g}_k\|\,\alpha_{k-1}\,\|\mathbf{d}_{k-1}\|}{\theta\alpha_{k-1}\|\mathbf{d}_{k-1}\|^2}$$
$$= \frac{(2L+t)\,\|\mathbf{g}_k\|}{\theta\,\|\mathbf{d}_{k-1}\|}.$$

Using (5.35) and (5.37) in (5.33), we obtain

$$\begin{aligned}
\|\mathbf{d}_k\|^2 &\le \|\mathbf{g}_k\|^2 + 2\frac{\|\mathbf{g}_k\|^2}{\lambda} + \frac{(2L+t)^2\,\|\mathbf{g}_k\|^2}{\theta^2\|\mathbf{d}_{k-1}\|^2}\|\mathbf{d}_{k-1}\|^2 \\
&\le \|\mathbf{g}_k\|^2 + 2\frac{\|\mathbf{g}_k\|^2}{\lambda} + \frac{(2L+t)^2}{\theta^2}\|\mathbf{g}_k\|^2 \\
&\le \left(1 + \frac{2}{\lambda} + \frac{(2L+t)^2}{\theta^2}\right)\|\mathbf{g}_k\|^2 \\
&\le \left(\frac{\lambda+2}{\lambda} + \frac{(2L+t)^2}{\theta^2}\right)\|\mathbf{g}_k\|^2 \\
&\le \frac{(\lambda+2)\theta^2 + \lambda\,(2L+t)^2}{\lambda\theta^2}\|\mathbf{g}_k\|^2.
\end{aligned} \tag{5.38}$$

Next, dividing both sides of (5.38) by $\|\mathbf{g}_k\|^4$ and using (5.32), it can be concluded

$$\begin{aligned}
\frac{\|\mathbf{d}_k\|^2}{\|\mathbf{g}_k\|^4} &\le \frac{(\lambda+2)\theta^2 + \lambda\,(2L+t)^2}{\lambda\theta^2}\cdot\frac{1}{c_1^2} \\
\frac{\|\mathbf{g}_k\|^4}{\|\mathbf{d}_k\|^2} &\ge \frac{\lambda\theta^2\cdot c_1^2}{(\lambda+2)\theta^2 + \lambda\,(2L+t)^2}.
\end{aligned} \tag{5.39}$$

The inequalities in (5.39) imply

$$\sum_{k=0}^{\infty}\frac{\|\mathbf{g}_k\|^4}{\|\mathbf{d}_k\|^2} \ge \sum_{k=0}^{\infty}\frac{\lambda\theta^2\cdot c_1^2}{(\lambda+2)\theta^2 + \lambda\,(2L+t)^2} = \infty. \tag{5.40}$$

Therefore, $\|\mathbf{g}_k\| \ge c_1$ causes a contradiction to (5.4). Consequently, (5.31) is confirmed for **Case 1**.

Case 2. In the cases $\beta_k \equiv \beta_k^{\mathrm{MLSDL}}$, applying Lemma 5.6 and Assumption 5.1, we have

$$\begin{aligned}
\beta_k^{\mathrm{MLSDL}} &\le \left|\frac{\|\mathbf{g}_k\|^2 - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{-\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}} - t\frac{\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}}{\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}}\right| \\
&\le \left|\frac{\mathbf{g}_k^T\mathbf{g}_k - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{g}_{k-1}\right|}{-\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}}\right| + t\frac{\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}\right|}{\left|\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}\right|} \\
&\le \frac{\left|\mathbf{g}_k^T\left(\mathbf{g}_k - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_{k-1}\right)\right|}{\left|-\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}\right|} + t\frac{\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}\right|}{\left|\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}\right|} \\
&= \frac{\left|\mathbf{g}_k^T\left(\mathbf{g}_k - \mathbf{g}_{k-1} + \mathbf{g}_{k-1} - \frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_{k-1}\right)\right|}{\left|-\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}\right|} + t\frac{\left|\mathbf{g}_k^{\mathrm{T}}\mathbf{s}_{k-1}\right|}{\left|\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}\right|}
\end{aligned}$$

$$
\begin{aligned}
&\leq \frac{\|\mathbf{g}_k\|\left(\|\mathbf{g}_k-\mathbf{g}_{k-1}\|+\left\|\mathbf{g}_{k-1}-\frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\mathbf{g}_{k-1}\right\|\right)}{\left|-\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}\right|}+t\frac{\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|}{\left|\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}\right|}\\
&\leq \frac{\|\mathbf{g}_k\|\left(\|\mathbf{g}_k-\mathbf{g}_{k-1}\|+\left\|\mathbf{g}_{k-1}(1-\frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|})\right\|\right)}{\left|-\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}\right|}+t\frac{\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|}{\left|\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}\right|}\\
&\leq \frac{\|\mathbf{g}_k\|\left(\|\mathbf{g}_k-\mathbf{g}_{k-1}\|+\left|1-\frac{\|\mathbf{g}_k\|}{\|\mathbf{g}_{k-1}\|}\right|\|\mathbf{g}_{k-1}\|\right)}{\left|-\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}\right|}+t\frac{\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|}{\left|\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}\right|}\\
&\leq \frac{\|\mathbf{g}_k\|\left(\|\mathbf{g}_k-\mathbf{g}_{k-1}\|+|\|\mathbf{g}_{k-1}\|-\|\mathbf{g}_k\||\right)}{\left|-\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}\right|}+t\frac{\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|}{\left|\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}\right|}\\
&\leq \frac{\|\mathbf{g}_k\|\left(\|\mathbf{g}_k-\mathbf{g}_{k-1}\|+\|\mathbf{g}_k-\mathbf{g}_{k-1}\|\right)}{\left|-\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{g}_{k-1}\right|}+t\frac{\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|}{\left|\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}\right|}\\
&\leq \frac{2\cdot\|\mathbf{g}_k\|\,\|\mathbf{g}_k-\mathbf{g}_{k-1}\|}{c\|\mathbf{g}_{k-1}\|^2}+t\frac{\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|}{\left|\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}\right|}\\
&\leq \frac{2L\cdot\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|}{c\|\mathbf{g}_{k-1}\|^2}+t\frac{\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|}{\left|\mathbf{d}_{k-1}^{\mathrm{T}}\mathbf{y}_{k-1}\right|}.
\end{aligned}
\tag{5.41}
$$

From (5.18), (5.19) and (5.41), we conclude

$$
\begin{aligned}
\beta_k^{\mathrm{MLSDL}} &\leq \frac{2L\cdot\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|}{c\|\mathbf{g}_{k-1}\|^2}+t\frac{\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|}{\theta\|\mathbf{s}_{k-1}\|^2}\\
&= \frac{2L\cdot\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|}{c\|\mathbf{g}_{k-1}\|^2}+\frac{t\,\|\mathbf{g}_k\|}{\theta\,\|\mathbf{s}_{k-1}\|}\\
&\leq \frac{2L\cdot\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|}{c\cdot c_1^2}+\frac{t\,\|\mathbf{g}_k\|}{\theta\,\|\mathbf{s}_{k-1}\|}\\
&\leq \frac{2L\theta\cdot\|\mathbf{g}_k\|\,\|\mathbf{s}_{k-1}\|^2+c\cdot c_1^2\cdot t\,\|\mathbf{g}_k\|}{c\cdot c_1^2\cdot\theta\,\|\mathbf{s}_{k-1}\|}\\
&\leq \frac{\left(2L\theta\cdot D^2+c\cdot c_1^2\cdot t\right)\|\mathbf{g}_k\|}{c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\,\|\mathbf{d}_{k-1}\|}.
\end{aligned}
\tag{5.42}
$$

Replacement of (5.35) and (5.42) in (5.33) leads to

$$
\begin{aligned}
\|\mathbf{d}_k\|^2 &\leq \|\mathbf{g}_k\|^2+2\frac{\|\mathbf{g}_k\|^2}{\lambda}+\frac{\left(2L\theta\cdot D^2+c\cdot c_1^2\cdot t\right)^2\|\mathbf{g}_k\|^2}{\left(c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\right)^2\|\mathbf{d}_{k-1}\|^2}\|\mathbf{d}_{k-1}\|^2\\
&= \|\mathbf{g}_k\|^2+2\frac{\|\mathbf{g}_k\|^2}{\lambda}+\frac{\left(2L\theta\cdot D^2+c\cdot c_1^2\cdot t\right)^2}{\left(c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\right)^2}\|\mathbf{g}_k\|^2\\
&= \left(1+\frac{2}{\lambda}+\frac{\left(2L\theta\cdot D^2+c\cdot c_1^2\cdot t\right)^2}{\left(c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\right)^2}\right)\|\mathbf{g}_k\|^2\\
&= \left(\frac{\lambda+2}{\lambda}+\frac{\left(2L\theta\cdot D^2+c\cdot c_1^2\cdot t\right)^2}{\left(c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\right)^2}\right)\|\mathbf{g}_k\|^2\\
&= \frac{(\lambda+2)\left(c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\right)^2+\lambda\left(2L\theta\cdot D^2+c\cdot c_1^2\cdot t\right)^2}{\lambda\left(c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\right)^2}\|\mathbf{g}_k\|^2.
\end{aligned}
\tag{5.43}
$$

Next, dividing both sides of (5.43) by $\|\mathbf{g}_k\|^4$ and using (5.32), it can be concluded

$$\begin{aligned}\frac{\|\mathbf{d}_k\|^2}{\|\mathbf{g}_k\|^4} &\leq \frac{(\lambda+2)\left(c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\right)^2+\lambda\left(2L\theta\cdot D^2+c\cdot c_1^2\cdot t\right)^2}{\lambda\left(c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\right)^2}\cdot\frac{1}{c_1^2}\\ \frac{\|\mathbf{g}_k\|^4}{\|\mathbf{d}_k\|^2} &\geq \frac{\lambda\left(c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\right)^2\cdot c_1^2}{(\lambda+2)\left(c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\right)^2+\lambda\left(2L\theta\cdot D^2+c\cdot c_1^2\cdot t\right)^2}.\end{aligned} \tag{5.44}$$

The inequalities in (5.44) imply

$$\sum_{k=0}^{\infty}\frac{\|\mathbf{g}_k\|^4}{\|\mathbf{d}_k\|^2}\geq\sum_{k=0}^{\infty}\frac{\lambda\left(c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\right)^2\cdot c_1^2}{(\lambda+2)\left(c\cdot c_1^2\cdot\theta\cdot\alpha_{k-1}\right)^2+\lambda\left(2L\theta\cdot D^2+c\cdot c_1^2\cdot t\right)^2}=\infty. \tag{5.45}$$

Therefore, $\|\mathbf{g}_k\|\geq c_1$ causes a contradiction to (5.4). Consequently, (5.31) is confirmed for **Case 2**. The proof is complete. □

6. **Numerical experiments.** The code used in the testing experiments is written in the software Matlab R2017a, and executed on the personal computer Workstation Intel Core i3 2.0 GHz, 8GB of RAM memory, and Windows 10 operating system. Three important criteria: the number of iterations (IT), number of function evaluations (FE) and CPU time (CPU) in all tested methods are analyzed.

The numerical experiments are performed on contains functions presented in [3], where much of the problems are taken over from CUTEr collection [14]. Each test function is tested 10 times with a gradually increasing values of the dimension by the rule $n = 10$, 50, 100, 200, 300, 500, 700, 800, 1000 and 1500.

Strong Wolfe line search use the following choice of parameters for all algorithms $\sigma_1 = 0.0001$ and $\sigma_2 = 0.5$.

We utilized the performance profile given in [43] to compare numerical results (IT, FE and CPU) for all tested methods. The upper curve of the selected performance profile corresponds to the method that shows the best performance.

6.1. **Numerical experiments on QN methods with constant diagonal Hessian approximation.** BFGS, DFP, SR1 updates in QN methods with respect to different ILS strategies are compared in [40]. The main conclusion is that the BFGS method is superior to the others. Continuing such research, we compare the numerical performances obtained from AGD, MSM and SM methods, i.e, gradient methods with acceleration parameter. The numerical experiment contains 25 test functions proposed in [3]. For each of tested functions, we performed 12 numerical experiments with 100, 200, 300, 500, 1000, 2000, 3000, 5000, 7000, 8000, 10000, and 15000 variables. Tested algorithms are based on the same implementation of the backtracking line search (Algorithm 2), which we set $\omega = 0.0001$ and $\varphi = 0.8$

The uniform stopping criteria in this numerical experiments are

$$\|\mathbf{g}_k\|\leq 10^{-6}\quad\text{and}\quad\frac{|f_{k+1}-f_k|}{1+|f_k|}\leq 10^{-16}.$$

Summary numerical data generated by AGD, MSM and SM method, tried on 25 test functions, are arranged in Table 4.

Table 4 contains numerical results corresponding to IT, FE and CPU criteria for the AGD, MSM and SM methods. Figures 1, 2, 3 illustrate the performance profiles corresponding to the results in Table 4 corresponding to the criterion IT, GE and CPU, respectively.

TABLE 4. Summary numerical results of the AGD, MSM and SM methods with respect to IT, FE and CPU.

	IT profile			FE profile			CPU time		
Test function	**AGD**	**MSM**	**SM**	**AGD**	**MSM**	**SM**	**AGD**	**MSM**	**SM**
Perturbed Quadratic	353897	34828	59908	13916515	200106	337910	6756.047	116.281	185.641
Raydan 1	22620	26046	14918	431804	311260	81412	158.359	31.906	36.078
Diagonal 3	120416	7030	12827	4264718	38158	69906	5527.844	52.609	102.875
Generalized Tridiagonal 1	670	346	325	9334	1191	1094	11.344	1.469	1.203
Extended Tridiagonal 1	3564	1370	4206	14292	10989	35621	55.891	29.047	90.281
Extended TET	443	156	156	3794	528	528	3.219	0.516	0.594
Diagonal 4	120	96	96	1332	636	636	0.781	0.203	0.141
Extended Himmelblau	396	260	196	6897	976	668	1.953	0.297	0.188
Perturbed quadratic diagonal	2542050	37454	44903	94921578	341299	460028	44978.750	139.625	185.266
Quadratic QF1	366183	36169	62927	13310016	208286	352975	12602.563	81.531	138.172
Extended quadratic penalty QP1	210	369	271	2613	2196	2326	1.266	1.000	0.797
Extended quadratic penalty QP2	395887	1674	3489	9852040	11491	25905	3558.734	3.516	6.547
Quadratic QF2	100286	32727	64076	3989239	183142	353935	1582.766	73.438	132.703
Extended quadratic exponential EP1	48	100	73	990	894	661	0.750	0.688	0.438
Extended Tridiagonal 2	1657	659	543	8166	2866	2728	3.719	1.047	1.031
ARWHEAD (CUTE)	5667	430	270	214284	5322	3919	95.641	1.969	1.359
Almost Perturbed Quadratic	356094	33652	60789	14003318	194876	338797	13337.125	73.047	133.516
LIARWHD (CUTE)	1054019	3029	18691	47476667	27974	180457	27221.516	9.250	82.016
ENGVAL1 (CUTE)	743	461	375	6882	2285	2702	3.906	1.047	1.188
QUARTC (CUTE)	171	217	290	402	494	640	2.469	1.844	2.313
Generalized Quartic	187	181	189	849	493	507	0.797	0.281	0.188
Diagonal 7	72	147	108	333	504	335	0.625	0.547	0.375
Diagonal 8	60	120	118	304	383	711	0.438	0.469	0.797
Full Hessian FH3	45	63	63	1352	566	631	1.438	0.391	0.391
Diagonal 9	329768	10540	13619	13144711	68189	89287	6353.172	43.609	38.672

From Table 4, we conclude that the AGD, MSM and SM methods have successfully solved all test functions.

Figure 1 presents the performance profiles of the IT of the AGD, MSM and SM methods. In this figure, it is observable that MSM method is best in 52.00% of the test functions compared with: AGD (24.00%) and SM (32.00%).

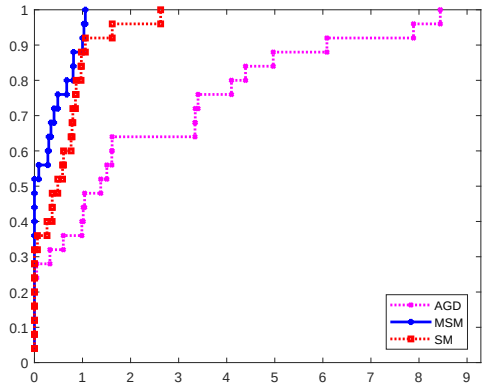

FIGURE 1. IT performance profile for AGD, MSM and SM methods.

From Figure 1, it is observable that the graph of the MSM method comes first to the top, which means that the MSM is superior compared to other considered methods with respect to the IT profile.

Figure 2 presents the performance profiles of the FE of the AGD, MSM and SM methods. It is observable that MSM method is best in 64.00% of tested functions compared with: AGD (12.00%) and SM (32.00%). In view of Figure 2, the MSM graph first comes to the top, which means that the MSM is winer with respect to the FE profile.

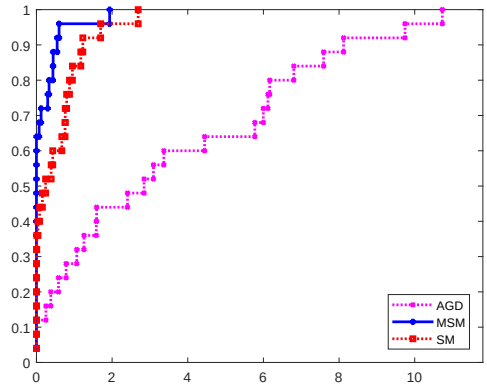

FIGURE 2. FE performance profile for AGD, MSM and SM methods.

Figure 3 presents the performance profiles of the CPU of the AGD, MSM and SM methods. It is obvious that MSM is winer in 56.00% of the test functions with respect to: AGD (4.00%) and SM (44.00%). Figure 3 demonstrates that the graph of the MSM method first comes to the highest level, which signifies that the MSM is winer with respect to the CPU.

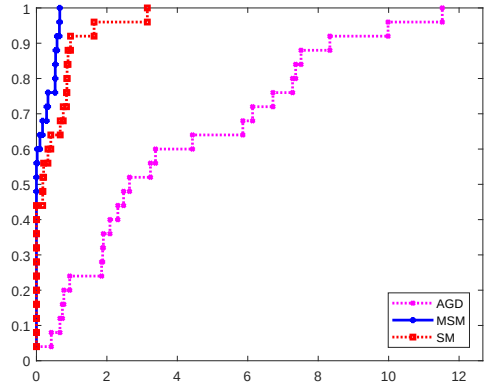

FIGURE 3. CPU time performance profile for AGD, MSM and SM methods.

From the previous analysis of the results shown in Table 4 and Figures 1-3, we can conclude that the MSM iterates are most efficient in terms of all three basic metrics: IT, FE and CPU. The MSM method has the smallest IT, FE and the CPU time compared to the other two methods on the most test functions.

6.2. **Numerical experiments on the CG methods with $\mathbf{y}_{k-1}^T\mathbf{g}_k$ in the numerator of β_k.** The uniform stopping criterion during testing CG methods is

$$\|\mathbf{g}_k\| \leq \epsilon,$$

where $\epsilon = 10^{-6}$ or when the number of function evaluations becomes greater than 1000000.

In this subsection, we compare the numerical results obtained from HS, PRP and LS methods, i.e., methods with $\mathbf{y}_{k-1}^{\mathrm{T}}\mathbf{g}_k$ in the numerator of β_k. The numerical experiment is based on 26 test functions. Summary numerical results for HS, PRP and LS method, tried on 26 test functions, are presented in Table 5.

Table 5 shows the numerical results (IT, FE and CPU) for the HS, PRP and LS methods.

TABLE 5. Summary numerical results of the HS, PRP and LS methods with respect to the IT, FE and CPU.

	IT profile			FE profile			CPU time		
Test function	**HS**	**PRP**	**LS**	**HS**	**PRP**	**LS**	**HS**	**PRP**	**LS**
Perturbed Quadratic	1157	1157	6662	3481	3481	19996	0.234	0.719	1.438
Raydan 2	NaN	174	40	NaN	373	120	NaN	0.094	0.078
Diagonal 2	NaN	1721	5007	NaN	6594	15498	NaN	1.313	2.891
Extended Tridiagonal 1	NaN	170	17079	NaN	560	54812	NaN	0.422	13.641
Diagonal 4	NaN	70	1927	NaN	180	5739	NaN	0.078	0.391
Diagonal 5	NaN	154	30	NaN	338	90	NaN	0.172	0.078
Extended Himmelblau	160	120	241	820	600	1043	0.172	0.125	0.172
Full Hessian FH2	5096	5686	348414	15294	17065	1045123	83.891	80.625	5081.875
Perturbed quadratic diagonal	1472	1120	21667	4419	3363	65057	0.438	0.391	2.547
Quadratic QF1	1158	1158	5612	3484	3484	16813	0.281	0.313	1.047
Extended quadratic penalty QP2	NaN	533	NaN	NaN	5395	NaN	NaN	0.781	NaN
Quadratic QF2	2056	2311	NaN	9168	9862	NaN	0.969	0.859	NaN
Extended quadratic exponential EP1	NaN	NaN	70	NaN	NaN	350	NaN	NaN	0.141
TRIDIA (CUTE)	6835	6744	NaN	20521	20248	NaN	1.438	1.094	NaN
Almost Perturbed Quadratic	1158	1158	5996	3484	3484	17998	0.281	0.328	1.063
LIARWHD (CUTE)	NaN	408	11498	NaN	4571	50814	NaN	0.438	2.969
POWER (CUTE)	7781	7789	190882	23353	23377	572656	1.422	1.219	14.609
NONSCOMP (CUTE)	4545	3647	NaN	15128	12433	NaN	0.875	0.656	NaN
QUARTC (CUTE)	NaN	165	155	NaN	1347	1466	NaN	0.781	0.766
Diagonal 6	NaN	174	137	NaN	373	442	NaN	0.109	0.125
DIXON3DQ (CUTE)	NaN	12595	12039	NaN	37714	36091	NaN	1.641	2.859
BIGGSB1 (CUTE)	NaN	11454	11517	NaN	34293	34530	NaN	1.969	2.141
Generalized Quartic	NaN	134	139	NaN	458	445	NaN	0.125	0.094
Diagonal 7	NaN	51	80	NaN	142	240	NaN	0.063	0.109
Diagonal 8	NaN	70	80	NaN	180	180	NaN	0.063	0.125
FLETCHCR (CUTE)	18292	19084	20354	178305	170266	171992	8.859	6.203	7.484

Figures 4, 5 and 6 plot the performance profiles for the results in Table 5 with respect to IT, FE and CPU criterion, respectively.

Figure 4 presents the performance profiles of the IT correspondig to the HS, PRP and LS methods. In this figure, it is observable that PRP method is best in 61.54% of the test functions compared with: HS (26.92%) and LS (23.08%). From Figure 4, it is observable that the graph of the PRP method comes first to the top, which signifies that the PRP outperforms other considered methods with respect to the IT criterion.

Figure 5 presents the performance profiles of the FE of the HS, PRP and LS methods. It is observable that PRP method is best in 69.23% of the test functions compared with: HS (23.08%) and LS (23.08%). From Figure 5, it is observed that the PRP graph first comes to the top, which signifies that the PRP is the best with respect to the FE.

Figure 6 presents the performance profiles of the CPU of the HS, PRP and LS methods. It is obvious that PRP is best in 69.23% of the test functions compared with: HS (11.54%) and LS (19.23%). Figure 6 demonstrates that the graph of the PRP method first comes to the top, which signifies that the PRP is the best with respect to the CPU.

From the previous analysis of the results shown in Table 5 and Figures 4-6, we can conclude that the PRP method achieved the best and most efficient results in terms of all three basic metrics: IT, FE and CPU.

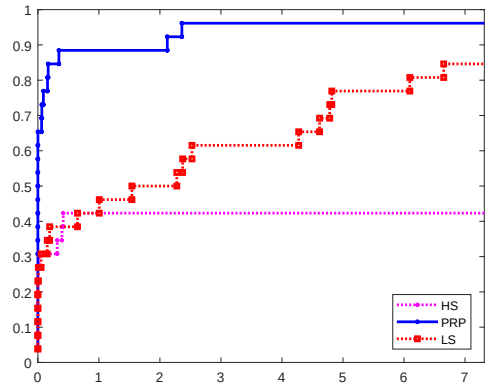

FIGURE 4. IT performance profile for HS, PRP and LS methods.

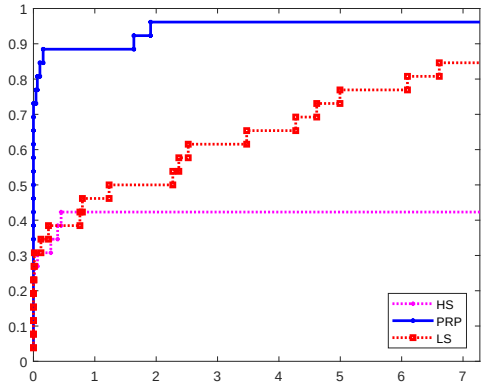

FIGURE 5. FE performance profile for HS, PRP and LS methods.

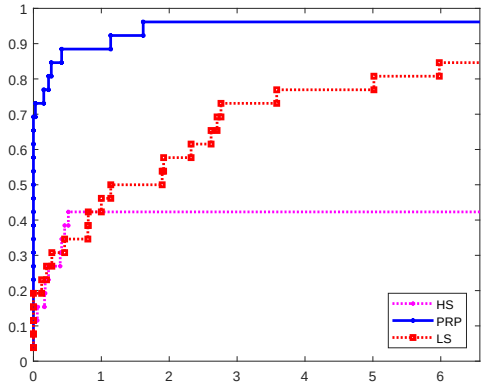

FIGURE 6. CPU time performance profile for HS, PRP and LS methods.

6.3. **Numerical experiments on CG methods with $\|\mathbf{g}_k\|^2$ in the numerator of β_k.** In this subsection, we compare the numerical results obtained from DY, FR and CD methods, i.e, methods with $\|\mathbf{g}_k\|^2$ in the numerator of β_k. The numerical experiment contains 25 test functions. Summary numerical results for DY, FR and CD method, tried on 25 test functions, are presented in Table 6.

Table 6 contains numerical results (IT, FE and CPU) for the DY, FR and CD methods. Figures 7, 8 and 9 plot the performance profiles for the results in Table 6 with respect to profiles IT, FE and CPU, respectively.

TABLE 6. Summary numerical results of the DY, FR and CD methods with respect to IT, FE and CPU.

	IT profile			FE profile			CPU time		
Test function	**DY**	**FR**	**CD**	**DY**	**FR**	**CD**	**DY**	**FR**	**CD**
Perturbed Quadratic	1157	1157	1157	3481	3481	3481	0.469	0.609	0.531
Raydan 2	86	40	40	192	100	100	0.063	0.016	0.016
Diagonal 2	1636	3440	2058	4774	7982	8063	0.922	1.563	1.297
Extended Tridiagonal 1	2081	690	1140	4639	2022	2984	1.703	1.141	1.578
Diagonal 4	70	70	70	200	200	200	0.047	0.031	0.016
Diagonal 5	40	124	155	100	258	320	0.109	0.141	0.125
Extended Himmelblau	383	339	207	1669	1467	961	0.219	0.172	0.172
Full Hessian FH2	4682	4868	4794	14054	14610	14390	65.938	66.469	65.922
Perturbed quadratic diagonal	1036	1084	1276	3114	3258	3834	0.406	0.422	0.422
Quadratic QF1	1158	1158	1158	3484	3484	3484	0.297	0.297	0.328
Quadratic QF2	NaN	NaN	2349	NaN	NaN	10073	NaN	NaN	1.531
Extended quadratic exponential EP1	NaN	60	60	NaN	310	310	NaN	0.109	0.125
Almost Perturbed Quadratic	1158	1158	1158	3484	3484	3484	0.422	0.453	0.391
LIARWHD (CUTE)	2812	1202	1255	12366	7834	7379	0.938	1.000	1.109
POWER (CUTE)	7779	7781	7782	23347	23353	23356	1.078	1.500	1.328
NONSCOMP (CUTE)	2558	13483	10901	49960	43268	33413	1.203	1.406	1.422
QUARTC (CUTE)	134	94	95	1132	901	916	0.688	0.672	0.563
Diagonal 6	86	40	40	192	100	100	0.047	0.063	0.063
DIXON3DQ (CUTE)	16047	18776	19376	48172	56369	58176	2.266	2.516	2.734
BIGGSB1 (CUTE)	15274	17835	18374	45853	53546	55170	2.875	2.922	2.484
Generalized Quartic	142	214	173	497	712	589	0.078	0.172	0.109
Diagonal 7	50	50	50	160	160	160	0.063	0.047	0.094
Diagonal 8	50	40	40	160	130	130	0.109	0.125	0.063
Full Hessian FH3	43	43	43	139	139	139	0.063	0.109	0.109
FLETCHCR (CUTE)	NaN	NaN	26793	NaN	NaN	240237	NaN	NaN	10.203

From Figure 7, it is observable that the graph of the CD method comes first to the top, which signifies that the CD outperforms other considered methods with respect to the IT.

Figure 8 presents the performance profiles of the FE of the DY, FR and CD methods. From Figure 8, it is observed that the CD graph first comes to the highest level, which means that the CD possesses best performances with respect to the criterion FE.

Figure 9 presents the performance profiles of the CPU of the DY, FR and CD methods. Figure 9 demonstrates that the graph of the CD method first achieves the top level,so that the CD is winer with respect to the CPU.

From the previous analysis of the results shown in Table 6 and Figures 7-9, it is clear that the CD method achieved most efficient results in terms of all three basic metrics: IT, FE and CPU.

6.4. **Numerical experiments on the hybrid conjugate gradient methods.** This subsection analyses numerical results obtained by running a MATLAB implementation with predefined conditions given at the beginning section. The following

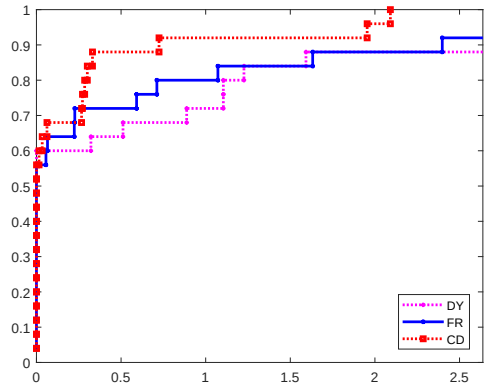

FIGURE 7. IT performance profile for DY, FR and CD methods.

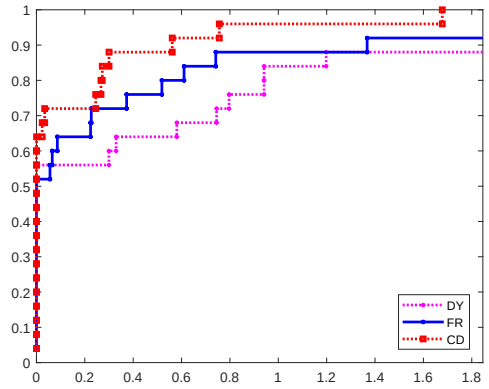

FIGURE 8. FE performance profile for DY, FR and CD methods.

ten hybrid CG methods in the form of (1.2) and (1.10), which differ only in the choice of the CG parameter β_k, are tested:

- HCG1: The CG method with β_k defined by (4.15).
- HCG2: The CG method with β_k defined by (4.16).
- HCG3: The CG method with β_k defined by (4.17).
- HCG4: The CG method with β_k defined by (4.18).
- HCG5: The CG method with β_k defined by (4.19) in which $c = \frac{1-\sigma}{1+\sigma}$.
- HCG6: The CG method with β_k defined by (4.20).
- HCG7: The CG method with the parameter β_k defined by (4.27).
- HCG8: The CG method with the parameter β_k defined by (4.22) in which $\theta_k \in [0, 1]$.
- HCG9: The CG method with the parameter β_k defined by (4.23) in which $\theta_k \in [0, 1]$.
- HCG10: The CG method with the parameter β_k defined by (4.24) in which $\nu_k, \theta_k \in [0, 1]$.

The numerical experiment contains 25 test functions.

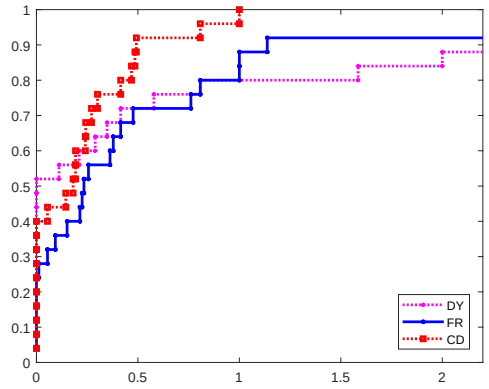

FIGURE 9. CPU time performance profile for DY, FR and CD methods.

Summary numerical results for hybrid CG methods, tried on 25 test functions, with respect to IT, FE and CPU profiles are presented in Table 7.

TABLE 7. Summary numerical results of the hybrid CG methods HCG1–HCG10 with respect to IT.

Test function	HCG1	HCG2	HCG3	HCG4	HCG5	HCG6	HCG7	HCG8	HCG9	HCG10
Perturbed Quadratic	1157	1157	1157	1157	1157	1157	1157	1157	1157	1157
Raydan 2	40	40	40	57	78	81	40	69	NaN	126
Diagonal 2	1584	1581	1542	1488	1500	2110	2193	1843	1475	1453
Extended Tridiagonal 1	805	623	754	2110	2160	10129	1167	966	NaN	270
Diagonal 4	60	60	70	60	70	70	60	70	NaN	113
Diagonal 5	124	39	98	39	120	109	39	141	154	130
Extended Himmelblau	145	139	111	161	181	207	159	381	109	108
Full Hessian FH2	5036	5036	5036	4820	4820	4800	4994	4789	5163	5705
Perturbed quadratic diagonal	1228	1214	1266	934	1093	987	996	1016	NaN	2679
Quadratic QF1	1158	1158	1158	1158	1158	1158	1158	1158	NaN	1158
Quadratic QF2	2125	2098	2174	1995	1991	2425	2378	NaN	2204	2034
TRIDIA (CUTE)	NaN	NaN	NaN	6210	6210	5594	NaN	NaN	6748	7345
Almost Perturbed Quadratic	1158	1158	1158	1158	1158	1158	1158	1158	1158	1158
LIARWHD (CUTE)	1367	817	1592	1024	1831	1774	531	2152	NaN	573
POWER (CUTE)	7782	7782	7782	7779	7779	7802	7781	7780	NaN	7781
NONSCOMP (CUTE)	10092	10746	8896	10466	9972	13390	11029	3520	3988	11411
QUARTC (CUTE)	94	160	145	150	126	95	160	114	165	154
Diagonal 6	40	40	40	57	78	81	40	69	NaN	126
DIXON3DQ (CUTE)	12182	5160	11257	5160	11977	14302	5160	17080	NaN	12264
BIGGSB1 (CUTE)	10664	5160	10479	5160	11082	13600	5160	16192	NaN	11151
Generalized Quartic	129	107	110	107	142	153	107	123	131	145
Diagonal 7	50	NaN	40	NaN	40	50	NaN	50	51	40
Diagonal 8	40	40	40	50	NaN	50	40	NaN	NaN	40
Full Hessian FH3	43	42	42	42	42	43	42	43	NaN	NaN
FLETCHCR (CUTE)	17821	17632	18568	17272	17446	26794	24865	NaN	17315	20813

Figure 10 plot corresponding performance profiles IT for the results included in Table 7, in three columns denoted by IT.

From Figure 10, it is observable that the graph of the HCG6 method comes first to the top, which signifies that the HCG6 outperforms other considered methods with respect to the IT. However, if we look in more detail Figure 10, we can see that the HCG6 method does not have the best results, it has the best results in only (16%), while the HCG7 method has the best results in (52%) on the number of functions tested. The reason for such behavior lies in the fact that the HCG6 method is the only one that has successfully solved all the test problems.

The numerical results of the hybrid CG methods with respect to the FE are arranged in Table 8.

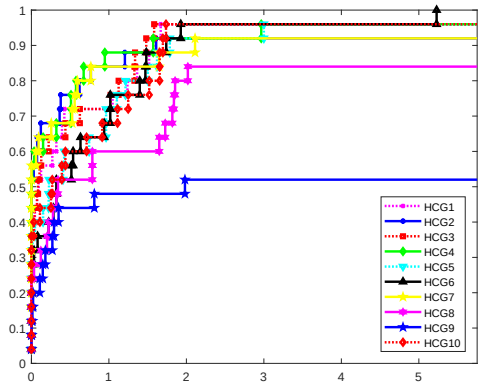

FIGURE 10. IT performance profile for hybrid CG methods HCG1–HCG10.

TABLE 8. Summary numerical results of the hybrid CG methods HCG1–HCG10 with respect to FE.

Test function	HCG1	HCG2	HCG3	HCG4	HCG5	HCG6	HCG7	HCG8	HCG9	HCG10
Perturbed Quadratic	3481	3481	3481	3481	3481	3481	3481	3481	3481	3481
Raydan 2	100	100	100	134	176	182	100	158	NaN	282
Diagonal 2	6136	6217	6006	5923	5944	8281	8594	4822	5711	5636
Extended Tridiagonal 1	2369	1991	2275	4678	4924	22418	3119	2661	NaN	869
Diagonal 4	170	170	200	170	200	200	170	200	NaN	339
Diagonal 5	258	88	206	88	270	228	88	292	338	270
Extended Himmelblau	855	687	583	763	813	961	757	1613	567	594
Full Hessian FH2	15115	15115	15115	14467	14467	14407	14989	14374	15495	17122
Perturbed quadratic diagonal	3686	3647	3805	2805	3282	2967	2993	3053	NaN	8044
Quadratic QF1	3484	3484	3484	3484	3484	3484	3484	3484	NaN	3484
Quadratic QF2	9455	9202	9501	9016	9054	10229	10086	NaN	9531	9085
TRIDIA (CUTE)	NaN	NaN	NaN	18640	18640	16792	NaN	NaN	20260	22051
Almost Perturbed Quadratic	3484	3484	3484	3484	3484	3484	3484	3484	3484	3484
LIARWHD (CUTE)	7712	5931	8275	6165	8113	9395	5854	10305	NaN	4848
POWER (CUTE)	23356	23356	23356	23347	23347	23416	23353	23350	NaN	23353
NONSCOMP (CUTE)	31355	33211	27801	32705	31458	40807	34013	23411	13367	35106
QUARTC (CUTE)	901	1254	1261	1224	1224	916	1254	1041	1347	1305
Diagonal 6	100	100	100	134	176	182	100	158	NaN	282
DIXON3DQ (CUTE)	36508	15534	33759	15534	35926	42952	15534	51284	NaN	36796
BIGGSB1 (CUTE)	31960	15534	31427	15534	33247	40846	15534	48620	NaN	33469
Generalized Quartic	457	371	370	371	481	529	371	439	446	467
Diagonal 7	160	NaN	130	NaN	130	160	NaN	160	142	13
Diagonal 8	130	130	130	160	NaN	160	130	NaN	NaN	130
Full Hessian FH3	139	136	136	136	136	139	136	139	NaN	NaN
FLETCHCR (CUTE)	166463	165774	168739	175309	175845	240240	184939	NaN	174406	215687

Figure 11 plots the performance profiles for the results in Table 8, in three columns denoted by FE.

From Figure 11, it is observable that the graph of the HCG6 method comes first to the top, which signifies that the HCG6 outperforms other considered methods with respect to the FE. However, if we look in more detail Figure 11, we can see an identical situation as in Figure 10 that the HCG6 method does not have the best results, it has the best results in only (16%), while the HCG2 method has the best results in (48%) on the number of functions tested.

Table 9 contains numerical results of the hybrid CG methods with respect to the CPU.

Figure 12 plots the performance profiles for the results in Tables 9.

From Figure 12, it is observable that the graph of the HCG6 method comes first to the top, which signifies that the HCG6 outperforms other considered methods

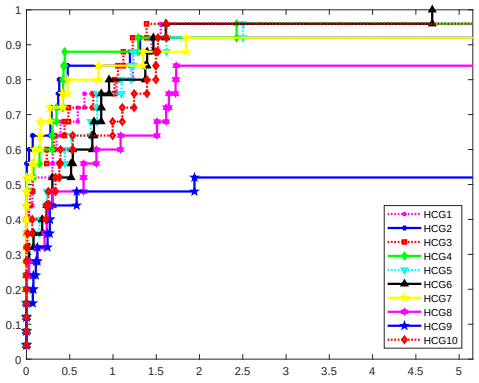

FIGURE 11. FE performance profile for hybrid CG methods HCG1–HCG10).

TABLE 9. Summary numerical results of the hybrid CG methods HCG1–HCG10 with respect to the CPU (sec).

Test function	HCG1	HCG2	HCG3	HCG4	HCG5	HCG6	HCG7	HCG8	HCG9	HCG10
Perturbed Quadratic	0.656	0.516	0.781	0.719	0.594	0.438	0.719	0.688	0.844	0.688
Raydan 2	0.031	0.063	0.078	0.078	0.078	0.078	0.078	0.078	NaN	0.078
Diagonal 2	1.453	1.328	1.656	1.172	1.438	1.797	1.813	1.266	1.250	1.141
Extended Tridiagonal 1	1.016	1.125	1.359	2.250	2.375	7.578	1.672	1.375	NaN	0.922
Diagonal 4	0.031	0.031	0.031	0.078	0.078	0.047	0.109	0.094	NaN	0.094
Diagonal 5	0.141	0.063	0.156	0.094	0.094	0.125	0.109	0.078	0.219	0.156
Extended Himmelblau	0.172	0.172	0.109	0.141	0.172	0.141	0.125	0.141	0.172	0.125
Full Hessian FH2	83.125	91.938	86.984	85.766	94.484	78.281	77.141	74.500	80.969	82.469
Perturbed quadratic diagonal	0.406	0.609	0.641	0.375	0.563	0.359	0.328	0.344	NaN	0.734
Quadratic QF1	0.359	0.438	0.422	0.422	0.406	0.391	0.484	0.422	NaN	0.281
Quadratic QF2	1.047	1.313	1.203	1.156	1.063	1.156	1.000	NaN	1.094	1.047
TRIDIA (CUTE)	NaN	NaN	NaN	1.688	1.391	1.859	NaN	NaN	1.875	1.391
Almost Perturbed Quadratic	0.406	0.438	0.516	0.594	0.250	0.359	0.406	0.578	0.641	0.422
LIARWHD (CUTE)	0.938	0.828	1.203	0.797	1.125	1.172	0.938	1.203	NaN	0.594
POWER (CUTE)	1.563	1.672	1.750	1.609	1.625	1.578	1.625	1.188	NaN	1.453
NONSCOMP (CUTE)	1.547	1.484	1.063	1.766	1.422	1.719	1.516	1.063	1.203	1.703
QUARTC (CUTE)	0.750	1.000	0.969	0.969	0.875	0.797	0.938	0.703	1.266	0.93
Diagonal 6	0.078	0.078	0.078	0.094	0.063	0.016	0.016	0.125	NaN	0.109
DIXON3DQ (CUTE)	2.047	1.453	2.016	1.484	2.359	2.234	1.406	2.297	NaN	2.078
BIGGSB1 (CUTE)	1.875	2.047	2.359	1.750	2.250	2.391	1.422	2.672	NaN	2.422
Generalized Quartic	0.063	0.125	0.141	0.156	0.125	0.094	0.078	0.109	0.172	0.109
Diagonal 7	0.063	NaN	0.016	NaN	0.109	0.063	NaN	0.063	0.063	0.063
Diagonal 8	0.078	0.125	0.078	0.031	NaN	0.063	0.109	NaN	NaN	0.078
Full Hessian FH3	0.063	0.047	0.109	0.047	0.031	0.063	0.047	0.109	NaN	NaN
FLETCHCR (CUTE)	5.656	6.750	7.922	9.484	6.484	8.766	7.281	NaN	6.906	7.547

with respect to the CPU. However, if we look in more detail Figure 12, we can see an identical situation as in the figures 10 and 11 that the HCG6 method does not have the best results, it has the best results in only (8%), while the HCG7 and HCG10 methods has the best results in (20%) on the number of functions tested.

6.5. **Numerical experiments on the modified Dai-Liao methods.** The numerical experiments presented in this subsection investigate the influence of the scalar size in the modified Dai-Liao methods. The previously mentioned variants of the DL method use a fixed value of the parameter t. It can also be seen that the scalar t in all the above papers are greater than 0 and is less than 1. Analyzing the results from [18, 26, 131, 139], we conclude that the scalar t was defined by a fixed value of 0.1 in numerical experiments. Also, numerical experience related the fixed

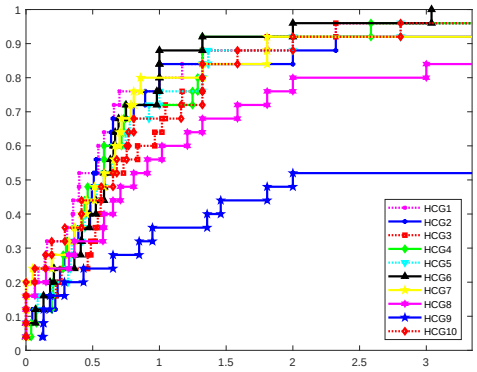

FIGURE 12. CPU time performance profile for hybrid CG methods HCG1–HCG10.

valued $t = 1$ was reported in [26]. Common numerical experience is that different choice of t initiate totally different numerical experience.

That is why we come to the next question. What is the 'best' value of $t \in (0, 1)$ from the computational point of view? Because of that, our intention is to investigate numerically and theoretically behavior of different variants of the DL conjugate gradient framework with respect to various values t. For this reason, we started this research with the aim to find answer to the aforementioned question. Our strategy is to select several values of the parameter t within the interval $(0, 1)$ and to compare the obtained results based on different criteria. In that way, we will get the answer to the question: whether it is better to take values closer to zero or closer to one.

6.5.1. *Motivations and the corresponding algorithm.* As we have already indicated in the previous section, the aim of this subsection is to answer to the question: what is the 'best' value of $t \in (0, 1)$ in DL CG computational scheme? Our plan is to examine numerically the influence of the scalar t in the DL class of iterations and determine some rules for its appropriate choice and, if possible, find the best value. The detailed research plan is to find the answer to two challenging questions:

- Does and how much the values of the scalar t affect each of the methods DHSDL, DLSDL, MHSDL and MLSDL, which are observed individually, with respect to IT, FE and the CPU time (CPU)?

- Does the choice of t favor one (or some) of the considered methods?

To give an answer to these questions, we would have to test all the methods under the same conditions. During testing, we will compare all the considered methods with the same values of required scalars. The Algorithm 2, i.e. the backtracking line search, determines the step-size α_k in (1.2).

Algorithm 6 gives the corresponding general framework for DHSDL, DLSDL, MHSDL and MLSDL methods.

Values of t used during the testing of the observed methods are given in the Table 10. Each particular value of the scalar t is marked with one of the labels $T1$ to $T6$, corresponding to a joined value of t. Five of the six given values are fixed during testing of the observed methods. Only the value of t labeled by $T1$ is variable during the iterations, where the value in kth iteration is denoted by t_k.

Algorithm 6 Modified Dai-Liao conjugate gradient methods.

Require: A starting point $\mathbf{x}_0$, real numbers $0 < \epsilon < 1$, $0 < \delta < 1$, $\mu > 1$ and $t > 0$.

1: Set $k=0$ and compute $\mathbf{d}_0 = -\mathbf{g}_0$.
2: If
$$\|\mathbf{g}_k\| \leq \epsilon \quad \text{and} \quad \frac{|f_{k+1} - f_k|}{1 + |f_k|} \leq \delta$$
STOP;
else perform Step 3.
3: Determine $\alpha_k \in (0, 1)$ using backtracking in Algorithm 2.
4: Compute $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$.
5: Calculate $\mathbf{g}_{k+1}$, $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$, $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$.
6: Calculate β_k by (4.9) or (4.10) or (4.11) or (4.12).
7: Compute $\mathbf{d}_k = -\mathbf{g}_k + \beta_k \mathbf{d}_{k-1}$.
8: $k := k + 1$, and go to Step 2.

In this case, the value t_k is obtained from backtracking line search, i.e., $t_k = \alpha_k$ inherits a new value obtained from Algorithm 2 in each iteration.

The reason why we decided to define values t_k from the backtracking line search algorithm is:
- $t_k \in (0, 1)$;
- t_k also affects the iterative steps when computing the next value for $\mathbf{x}_k$.

TABLE 10. Labels and values of scalar t in the DHSDL, DLSDL, MHSDL and MLSDL methods.

Label	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**
Value of the scalar t	$t_k = \alpha_k$	0.05	0.1	0.2	0.5	0.9

The convergence of the above methods has already considered in the mentioned references. Our goal is to give a unified analysis of the convergence of proposed DL methods. The aim of our research is to investigate the influence of the scalar t in the DL iterates.

Each particular value of the scalar t is marked with one of the labels $T1$ to $T6$, corresponding to a joined value of t. Five of the six given values are fixed during testing of the observed methods. Only the value of t labeled by $T1$ is variable during the iterations, where the value in kth iteration is denoted by t_k. In this case, the value t_k is obtained from backtracking line search, i.e., $t_k = \alpha_k$ inherits a new value obtained from Algorithm 2 in each iteration.

The convergence of the above methods has already considered in the mentioned references. Our goal is to give a unified analysis of the convergence of the proposed DL methods. The aim of our research is to investigate the influence of the scalar t in the DL iterates.

Arranged numerical results are generated by testing and comparing DHSDL, DLSDL, MHSDL and MLSDL methods on 6 different values t, denoted by $T1$-$T6$. Three important criteria (IT, CPU, FE) in all tested methods are analyzed. Numerical report is based on 22 test functions proposed in [3]. For each of tested functions, we performed 10 numerical experiments with 100, 500, 1000, 3000, 5000, 7000, 8000,

10000, 15000 and 20000 variables. Summary results for DHSDL, DLSDL, MHSDL and MLSDL methods, tried on 22 tests, are presented.

The uniform stopping criteria are (refer to previous)

$$\|\mathbf{g}_k\| \leq 10^{-6} \quad \text{and} \quad \frac{|f_{k+1} - f_k|}{1 + |f_k|} \leq 10^{-16}.$$

The backtracking parameters for all algorithms are $\omega = 0.0001$ and $\varphi = 0.8$.

During the testing of the DHSDL and DLSDL methods, the constant parameter $\mu = 1.2$ was used in each iteration.

6.5.2. *Numerical eexperiments on DHSDL method.* Figure 13 indicates the performance profiles of the IT criterion with respect to the DHSDL method depending on the scalar value t. This figure exhibits that the DHSDL method successfully solved all the problems for all values of the scalar t. Also, the DHSDL-T2 method is finest in 45.5% of testings with respect to DHSDL-T1 (18.2%), DHSDL-T3 (9.1%), DHSDL-T4 (4.5%), DHSDL-T5 (13.6%) and DHSDL-T6 (13.6%). It can be noticed that the graph of DHSDL-T2 reaches the top first, which signifies that DHSDL-T2 is the best with respect to IT.

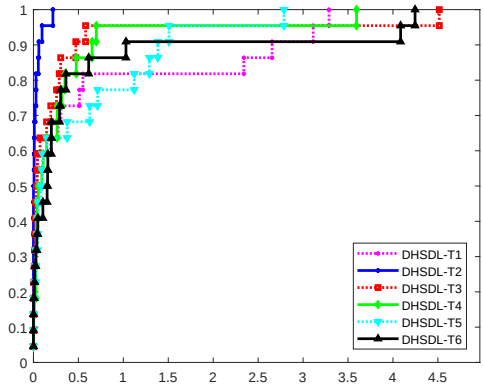

FIGURE 13. Performance profiles of DHSDL (T1,T2,T3,T4,T5,T6) method based on IT.

Figure 14 shows the performance profile given by FE of the DHSDL solver with respect to t. Evidently, DHSDL solves all test problems for all values of the scalar t, and the DHSDL-T2 method is winer in 50.0% of the test problems compared to the DHSDL-T1 (18.2%), DHSDL-T3 (9.1%), DHSDL-T4 (0%), DHSDL-T5 (13.6%) and DHSDL-T6(9.1%). From Figure 14, it is notifiable that the DHSDL-T2 is the best with respect to FE.

Figure 15 illustrates the CPU criterion spanned by the DHSDL method depending on the metrics t. Again, the DHSDL method is able to solve all the tested problems for all values of the scalar t. Further, the DHSDL-T6 method is superior in 40.9% of tests with respect to the DHSDL-T1 (4.5%), DHSDL-T2 (27.3%), DHSDL-T3 (9.1%), DHSDL-T4 (4.5%) and DHSDL-T5 (13.6%). We also observed that at the beginning, DHSDL-T2 does not perform well but as the number of problems increases, its graph crosses other graphs and achieves the top fist, which means that the DHSDL-T2 is dominant with respect to CPU.

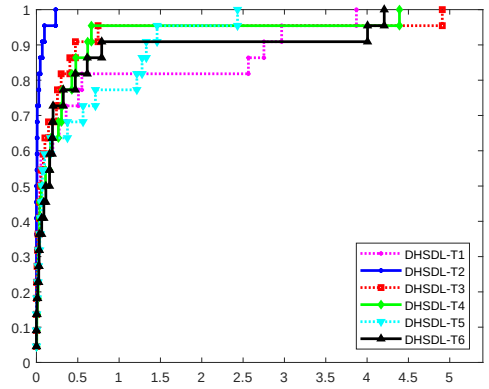

FIGURE 14. Performance profiles of DHSDL (T1,T2,T3,T4,T5,T6) method based on FE.

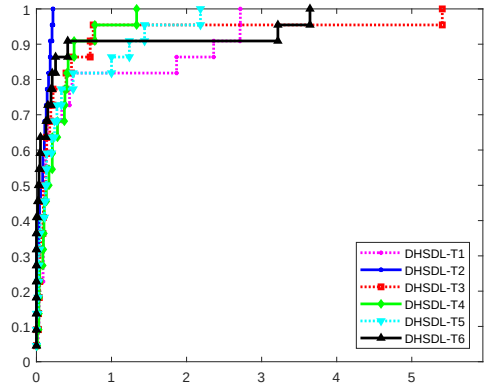

FIGURE 15. Performance profile of DHSDL (T1,T2,T3,T4,T5,T6) method based on CPU time.

The graphs displayed in figures 13–15 show that the DHSDL method has achieved superior results for $t \equiv T2 = 0.05$.

6.5.3. *Numerical experiments on DLSDL method.* Figure 16 illustrates the IT criterion in the DLSDL method depending on the scalar value t. It is observable that DLSDL successfully solves all the problems for all values of the scalar t. Moreover, the DLSDL-T2 method is winner in 36.4% of the tests compared to DLSDL-T1 (22.7%), DLSDL-T3 (13.6%), DLSDL-T4 (9.1%), DLSDL-T5 (9.1%) and DLSDL-T6 (13.6%). Figure 16 (left) exhibits that the graph DLSDL-T2 achieves the top first, which means that DLSDL-T2 outperforms all the other methods with respect to IT.

Figure 17 shows the performance profiles of the criterion FE corresponding to the DLSDL method and the scalar value t. In this Figure, it is observable that DLSDL for all values of the scalar t successfully solves all tests, and DLSDL-T2 is

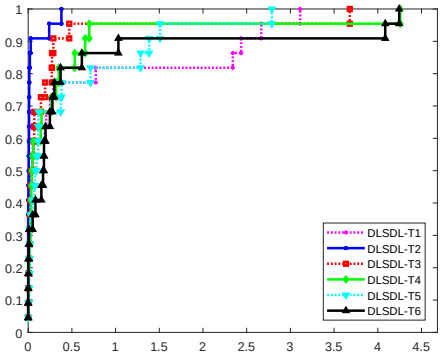

FIGURE 16. Performance profiles of DLSDL (T1,T2,T3,T4,T5,T6) method based on IT.

best in 36.4% of the test functions in comparison to DLSDL-T1 (18.2%), DLSDL-T3 (4.5%), DLSDL-T4 (18.2%), DLSDL-T5 (13.6%) and DLSDL-T6 (9.1%). From Figure 17, it is observed that DLSDL-T2 is the best with respect to FE.

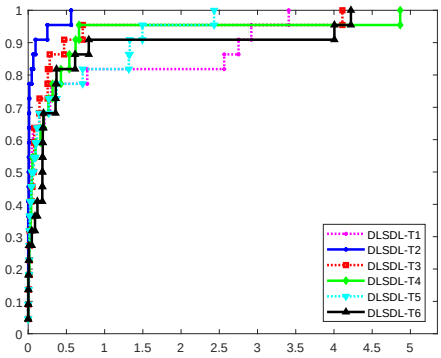

FIGURE 17. Performance profiles of DLSDL (T1,T2,T3,T4,T5,T6) method based on FE.

Figure 18 shows the performance profiles of the CPU time of the DLSDL method depending on t. The graphs in this figure indicate that DLSDL method solved all the problems for all values of t, and the DLSDL-T4 method is superior in 31.8% of the test problems compared to DLSDL-T1 (13.6%), DLSDL-T2 (22.7%), DLSDL-T3 (18.2%), DLSDL-T5 (9.1%) and DLSDL-T6 (4.6%). We also observed that at the beginning, DLSDL-T2 does not perform well; but as the number of problems increases, its graph crosses other graphs and comes to the top which signifies that, with respect to CPU, the DLSDL-T2 is the best.

Based on figures (16–18) analysis, we come to the conclusion that the DLSDL method has achieved best responses for $t \equiv T2 = 0.05$.

6.5.4. *Analysis of average values.* In Subsection 6.5.1, we have defined two questions. Our aim in this subsection is to give answers. In order to answer properly to the first question, in Tables 11, 12, 13 are collected average values for all three considered criteria.

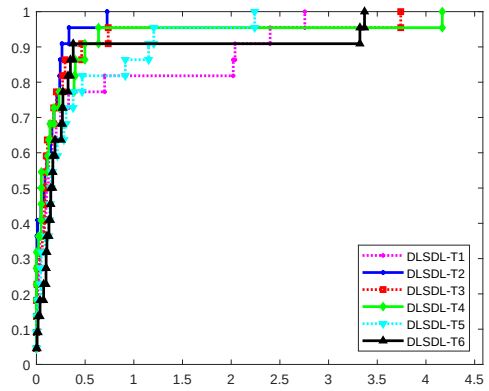

FIGURE 18. Performance profile of DLSDL (T1,T2,T3,T4,T5,T6) method based on CPU.

TABLE 11. Average IT values for 22 test functions tested on 10 numerical experiments.

Method	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**
DHSDL	32980.14	31281.32	33640.45	32942.36	34448.32	33872.36
DLSDL	30694.00	28701.14	31048.32	30594.77	31926.59	31573.05
MHSDL	29289.73	27653.64	29660.00	29713.50	30491.18	30197.27
MLSDL	25398.82	22941.77	24758.27	24250.68	25722.64	25032.64

TABLE 12. Average values of FE for 22 test functions tested on 10 numerical experiments.

Method	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**
DHSDL	1228585.50	1191960.55	1252957.09	1238044.36	1271176.59	1255710.45
DLSDL	1131421.41	1083535.14	1149482.41	1134315.00	1167030.14	1158554.77
MHSDL	1089700.41	1036710.32	1089777.64	1091985.41	1105299.91	1101380.18
MLSDL	904217.14	845017.55	891669.50	879473.14	913165.68	895652.36

TABLE 13. Average CPU time for 22 test functions tested on 10 numerical experiments.

Method	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**
DHSDL	902.06	894.73	917.77	930.56	911.28	870.93
DLSDL	816.08	790.63	804.69	816.28	803.84	809.67
MHSDL	770.78	751.65	728.61	749.70	712.64	720.57
MLSDL	573.14	587.41	581.50	576.32	582.62	580.96

After the numerical testing of the compared methods and the individual analysis for each method, we can now give a global conclusion of the behavior of the observed methods. The first conclusion is that the value of the scalar t significantly affects on each of the DHSDL, DLSDL, MHSDL and MLSDL methods with respect to all three criteria. Further, a common conclusion is that all the methods give the best performance profiles for the scalar value $t = 0.05$. We can also give an answer to another question. According to the previous analysis, a particular selection of the

scalar value t can give priority to one of the observed method. All methods do not behave identically for the same values of the scalar t. If we observe Table 11 which contains the average number of iterations, we can notice that the difference between the smallest and the biggest average result of the observed method is in the range of 10.1% to 12.1% in relation to the minimum obtained value. Table 12 shows the average results related FE. An individual comparison of considered methods leads to the conclusion that the difference between the smallest and the biggest average results is in the range of 6.6% to 8.9% in relation to the minimum obtained value. This brings us to the same conclusion once again, that is, the value of the scalar t affects the methods in a given percentage. Also, if we observe Table 13 with the average CPU time, we can notice that the difference between the smallest and the biggest average CPU time observed method is in the range of 2.5% to 10.6% in relation to the minimum value.

7. **Conclusion.** Overview of QN methods, CG methods and their classification are presented. Section 5 investigates convergence properties of CG methods, following the CG classes in accordance with the presented taxonomy of basic CG methods. Numerical experiments compare main classes of QN and CG methods. More precisely, main QN methods with constant diagonal Hessian approximation are compared as well as two classes of basic CG methods, hybrid CG methods, and finally some variants of modified Dai-Liao methods.

The problem of defining further improvements of the CGUP β_k is still open. Moreover, new CG methods could be defined using appropriate updates of the parameter t. Another research stream includes various hybridizations of so far proposed CG methods. On the other hand, there are open possibilities for defining new updates of the matrices B_k and H_k, used in defining QN methods. Continuing research on some composite definitions of β_k based on CG and BFGS updates, it is possible to discover new three-term (or even different) CG variants.

One of prospective fields for further research includes a generalization of the discrete-time approach to continuous-time approach, considered in [69]. Another possibility for further research is extension of gradient methods to tensor case. This possibility was exploited in [77] on solving $\mathcal{M}$-tensor equations.

Moreover, an application of low rank updates used in optimization can be transferred to appropriate numerical methods for computing generalized inverses. Applications of rank-one update formula are investigated in [115, 116].

Acknowledgments. Predrag Stanimirović and Dijana Mosić are supported by the Ministry of Education, Science and Technological Development, Grants No. 174013 and 174007. Haifeng Ma is supported by the National Natural Science Foundation of China under grant 11971136 and the bilateral project between China and and Poland (no. 37-18).

REFERENCES

[1] J. Abaffy, A new reprojection of the conjugate directions, *Numer. Algebra Control Optim.*, **9** (2019), 157–171.

[2] M. Al-Baali, Descent property and global convergence of the Fletcher-Reeves method with inexact line search, *IMA J. Numer. Anal.*, **5** (1985), 121–124.

[3] N. Andrei, An unconstrained optimization test functions collection, *Adv. Model. Optim.*, **10** (2008), 147–161.

[4] N. Andrei, An acceleration of gradient descent algorithm with backtracking for unconstrained optimization, *Numer. Algorithms*, **42** (2006), 63–73.

[5] N. Andrei, A Dai-Liao conjugate gradient algorithm with clustering of eigenvalues, *Numer. Algorithms*, **77** (2018), 1273–1282.
[6] N. Andrei, Relaxed gradient descent and a new gradient descent methods for unconstrained optimization, Visited August 19, (2018).
[7] N. Andrei, *Nonlinear Conjugate Gradient Methods for Unconstrained Optimization*, 1^{st} edition, Springer International Publishing, 2020.
[8] L. Armijo, Minimization of functions having Lipschitz continuous first partial derivatives, *Pacific J. Math.*, **16** (1966), 1–3.
[9] S. Babaie-Kafaki and R. Ghanbari, The Dai-Liao nonlinear conjugate gradient method with optimal parameter choices, *European J. Oper. Res.*, **234** (2014), 625–630.
[10] B. Baluch, Z. Salleh, A. Alhawarat and U. A. M. Roslan, A new modified three-term conjugate gradient method with sufficient descent property and its global convergence, *J. Math.*, **2017** (2017), Article ID 2715854, 12 pages.
[11] J. Barzilai and J. M. Borwein, Two-point step-size gradient method, *IMA J. Numer. Anal.*, **8** (1988), 141–148.
[12] M. Bastani and D. K. Salkuyeh, On the GSOR iteration method for image restoration, *Numerical Algebra, Control and Optimization*, (2020).
[13] A. E. J. Bogaers, S. Kok, B. D. Reddy and T. Franz, An evaluation of quasi-Newton methods for application to FSI problems involving free surface flow and solid body contact, *Computers & Structures*, **173** (2016), 71–83.
[14] I. Bongartz, A. R. Conn, N. Gould and Ph. L. Toint, CUTE: Constrained and unconstrained testing environments, *ACM Trans. Math. Softw.*, **21** (1995), 123–160.
[15] C. Brezinski, A classification of quasi-Newton methods, *Numer. Algorithms*, **33** (2003), 123–135.
[16] J. Cao and J. Wu, A conjugate gradient algorithm and its applications in image restoration, *Appl. Numer. Math.*, **152** (2020), 243–252.
[17] W. Cheng, A two-term PRP-based descent method, *Numer. Funct. Anal. Optim.*, **28** (2007), 1217–1230.
[18] Y. Cheng, Q. Mou, X. Pan and S. Yao, A sufficient descent conjugate gradient method and its global convergence, *Optim. Methods Softw.*, **31** (2016), 577–590.
[19] A. I. Cohen, Stepsize analysis for descent methods, *J. Optim. Theory Appl.*, **33** (1981), 187–205.
[20] A. R. Conn, N. I. M. Gould and Ph. L. Toint, Convergence of quasi-Newton matrices generated by the symmetric rank one update, *Math. Programming*, **50** (1991), 177–195.
[21] Y.-H. Dai, *Nonlinear Conjugate Gradient Methods*, Wiley Encyclopedia of Operations Research and Management Science, (2011).
[22] Y.-H. Dai, *Alternate Step Gradient Method*, Report AMSS–2001–041, Academy of Mathematics and Systems Sciences, Chinese Academy of Sciences, 2001.
[23] Y. Dai, A nonmonotone conjugate gradient algorithm for unconstrained optimization, *J. Syst. Sci. Complex.*, **15** (2002), 139–145.
[24] Y.-H. Dai and R. Fletcher, *On the Asymptotic Behaviour of some New Gradient Methods*, Numerical Analysis Report, NA/212, Dept. of Math. University of Dundee, Scotland, UK, 2003.
[25] Y.-H. Dai and C.-X. Kou, A nonlinear conjugate gradient algorithm with an optimal property and an improved wolfe line search, *SIAM. J. Optim.*, **23** (2013), 296–320.
[26] Y.-H. Dai and L.-Z. Liao, New conjugacy conditions and related nonlinear conjugate gradient methods, *Appl. Math. Optim.*, **43** (2001), 87–101.
[27] Y.-H. Dai and L.-Z. Liao, R-linear convergence of the Barzilai and Borwein gradient method, *IMA J. Numer. Anal.*, **22** (2002), 1–10.
[28] Y.-H. Dai and Q. Ni, Testing different conjugate gradient methods for large-scale unconstrained optimization, *J. Comput. Math.*, **21** (2003), 311–320.
[29] Z. Dai and F. Wen, Another improved Wei–Yao–Liu nonlinear conjugate gradient method with sufficient descent property, *Appl. Math. Comput.*, **218** (2012), 7421–7430.
[30] Y.-H. Dai and Y. Yuan, A nonlinear conjugate gradient method with a strong global convergence property, *SIAM J. Optim.*, **10** (1999), 177–182.
[31] Y.-H. Dai and Y. Yuan, An efficient hybrid conjugate gradient method for unconstrained optimization, *Ann. Oper. Res.*, **103** (2001), 33–47.

[32] Y. H. Dai and Y. Yuan, *A class of Globally Convergent Conjugate Gradient Methods*, Research report ICM-98-030, Institute of Computational Mathematics and Scientific/Engineering Computing, Chinese Academy of Sciences, 1998.
[33] Y. Dai and Y. Yuan, A class of globally convergent conjugate gradient methods, *Sci. China Ser. A*, **46** (2003), 251–261.
[34] Y. Dai, J. Yuan and Y.-X. Yuan, Modified two-point step-size gradient methods for unconstrained optimization, *Comput. Optim. Appl.*, **22** (2002), 103–109.
[35] Y.-H. Dai and Y.-X. Yuan, Alternate minimization gradient method, *IMA J. Numer. Anal.*, **23** (2003), 377–393.
[36] Y.-H. Dai and Y.-X. Yuan, Analysis of monotone gradient methods, *J. Ind. Manag. Optim.*, **1** (2005), 181–192.
[37] Y.-H. Dai and H. Zhang, *An Adaptive Two-Point Step-size gradient method*, Research report, Institute of Computational Mathematics and Scientific/Engineering Computing, Chinese Academy of Sciences, 2001.
[38] J. W. Daniel, The conjugate gradient method for linear and nonlinear operator equations, *SIAM J. Numer. Anal.*, **4** (1967), 10–26.
[39] S. Delladji, M. Belloufi and B. Sellami, Behavior of the combination of PRP and HZ methods for unconstrained optimization, *Numerical Algebra, Control and Optimization*, (2020).
[40] Y. Ding, E. Lushi and Q. Li, Investigation of quasi-Newton methods for unconstrained optimization, *International Journal of Computer Application*, **29** (2010), 48–58.
[41] S. S. Djordjević, Two modifications of the method of the multiplicative parameters in descent gradient methods, *Appl. Math, Comput.*, **218** (2012), 8672–8683.
[42] S. S. Djordjević, *Unconstrained Optimization Methods: Conjugate Gradient Methods and Trust-Region Methods*, Book Chapter, 2019.
[43] E. D. Dolan and J. J. Moré, Benchmarking optimization software with performance profiles, *Math. Program.*, **91** (2002), 201–213.
[44] M. S. Engelman, G. Strang and K.-J. Bathe, The application of quasi-Nnewton methods in fluid mechanics, *Internat. J. Numer. Methods Engrg.*, **17** (1981), 707–718.
[45] D. K. Faddeev and I. S. Sominskiĭ, *Collection of Problems on Higher Algebra. Gostekhizdat*, 2^{nd} edition, Moscow, 1949.
[46] A. G. Farizawani, M. Puteh, Y. Marina and A. Rivaie, A review of artificial neural network learning rule based on multiple variant of conjugate gradient approaches, *Journal of Physics: Conference Series*, **1529** (2020).
[47] R. Fletcher, *Practical Methods of Optimization, Unconstrained Optimization*, 1^{st} edition, Wiley, New York, 1987.
[48] R. Fletcher and C. M. Reeves, Function minimization by conjugate gradients, *Comput. J.*, **7** (1964), 149–154.
[49] J. C. Gilbert and J. Nocedal, Global convergence properties of conjugate gradient methods for optimization, *SIAM J. Optim.*, **2** (1992), 21–42.
[50] A. A. Goldstein, On steepest descent, *J. SIAM Control Ser. A*, **3** (1965), 147–151.
[51] L. Grippo, F. Lampariello and S. Lucidi, A nonmonotone line search technique for Newton's method, *SIAM J. Numer. ANAL.*, **23** (1986), 707–716.
[52] L. Grippo, F. Lampariello and S. Lucidi, A class of nonmonotone stability methods in unconstrained optimization, *Numer. Math.*, **59** (1991), 779–805.
[53] L. Grippo, F. Lampariello and S. Lucidi, A truncated Newton method with nonmonotone line search for unconstrained optimization, *J. Optim. Theory Appl.*, **60** (1989), 401–419.
[54] L. Grippo and M. Sciandrone, Nonmonotone globalization techniques for the Barzilai-Borwein gradient method, *Comput. Optim. Appl.*, **23** (2002), 143–169.
[55] W. W. Hager and H. Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search, *SIAM J. Optim.*, **16** (2005), 170–192.
[56] W. W. Hager and H. Zhang, Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent, *ACM Trans. Math. Software*, **32** (2006), 113–137.
[57] W. W. Hager and H. Zhang, A survey of nonlinear conjugate gradient methods, *Pac. J. Optim.*, **2** (2006), 35–58.
[58] L. Han and M. Neumann, Combining quasi-Newton and Cauchy directions, *Int. J. Appl. Math.*, **12** (2003), 167–191.
[59] B. A. Hassan, A new type of quasi-Newton updating formulas based on the new quasi-Newton equation, *Numer. Algebra Control Optim.*, **10** (2020), 227–235.

[60] M. R. Hestenes and E. Stiefel, Methods of conjugate gradients for solving linear systems, *J. Res. Nat. Bur. Standards*, **49** (1952), 409–436.
[61] Y. F. Hu and C. Storey, Global convergence result for conjugate gradient methods, *J. Optim. Theory Appl.*, **71** (1991), 399–405.
[62] S. Ishikawa, Fixed points by a new iteration method, *Proc. Am. Math. Soc.*, **44** (1974), 147–150.
[63] 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*, (2019).
[64] Z. Jia, Applications of the conjugate gradient method in optimal surface parameterizations, *Int. J. Comput. Math.*, **87** (2010), 1032–1039.
[65] J. Jian, L. Han and X. Jiang, A hybrid conjugate gradient method with descent property for unconstrained optimization, *Appl. Math. Model.*, **39** (2015), 1281–1290.
[66] S. H. Khan, A Picard-Mann hybrid iterative process, *Fixed Point Theory Appl.*, **2013** (2013), Article number: 69, 10 pp.
[67] Z. Khanaiah and G. Hmod, Novel hybrid algorithm in solving unconstrained optimizations problems, *International Journal of Novel Research in Physics Chemistry & Mathematics*, **4** (2017), 36–42.
[68] N. Kontrec and M. Petrović, Implementation of gradient methods for optimization of underage costs in aviation industry, *University Thought, Publication in Natural Sciences*, **6** (2016), 71–74.
[69] J. Kwon and P. Mertikopoulos, A continuous-time approach to online optimization, *J. Dyn. Games*, **4** (2017), 125–148.
[70] M. S. Lee, B. S. Goh, H. G. Harno and K. H. Lim, On a two phase approximate greatest descent method for nonlinear optimization with equality constraints, *Numer. Algebra Control Optim.*, **8** (2018), 315–326.
[71] D.-H. Li and M. Fukushima, A modified BFGS method and its global convergence in nonconvex minimization, *J. Comput. Appl. Math.*, **129** (2001), 15–35.
[72] X. Li and Q. Ruan, A modified PRP conjugate gradient algorithm with trust region for optimization problems, *Numer. Funct. Anal. Optim.*, **32** (2011), 496–506.
[73] X. Li, C. Shen and L.-H. Zhang, A projected preconditioned conjugate gradient method for the linear response eigenvalue problem, *Numer. Algebra Control Optim.*, **8** (2018), 389–412.
[74] D.-H. Li and X.-L. Wang, A modified Fletcher-Reeves-type derivative-free method for symmetric nonlinear equations, *Numer. Algebra Control Optim.*, **1** (2011), 71–82.
[75] K. H. Lim, H. H. Tan and H. G. Harno, Approximate greatest descent in neural network optimization, *Numer. Algebra Control Optim.*, **8** (2018), 327–336.
[76] J. Liu, S. Du and Y. Chen, A sufficient descent nonlinear conjugate gradient method for solving M-tensor equations, *J. Comput. Appl. Math.*, **371** (2020), 112709, 11 pp.
[77] J. K. Liu and S. J. Li, A projection method for convex constrained monotone nonlinear equations with applications, *Comput. Math. Appl.*, **70** (2015), 2442–2453.
[78] Y. Liu and C. Storey, Efficient generalized conjugate gradient algorithms, part 1: Theory, *J. Optim. Theory Appl.*, **69** (1991), 129–137.
[79] Q. Liu and X. Zou, A risk minimization problem for finite horizon semi-Markov decision processes with loss rates, *J. Dyn. Games*, **5** (2018), 143–163.
[80] I. E. Livieris and P. Pintelas, A descent Dai-Liao conjugate gradient method based on a modified secant equation and its global convergence, *ISRN Computational Mathematics*, **2012** (2012), Article ID 435495.
[81] M. Lotfi and S. M. Hosseini, An efficient Dai-Liao type conjugate gradient method by reformulating the CG parameter in the search direction equation, *J. Comput. Appl. Math.*, **371** (2020), 112708, 15 pp.
[82] Y.-Z. Luo, G.-J. Tang and L.-N. Zhou, Hybrid approach for solving systems of nonlinear equations using chaos optimization and quasi-Newton method, *Applied Soft Computing*, **8** (2008), 1068–1073.
[83] W. R. Mann, Mean value methods in iterations, *Proc. Amer. Math. Soc.*, **4** (1953), 506–510.
[84] M. Miladinović, P. Stanimirović and S. Miljković, Scalar correction method for solving large scale unconstrained minimization problems, *J. Optim. Theory Appl.*, **151** (2011), 304–320.
[85] S. K. Mishra and B. Ram, *Introduction to Unconstrained Optimization with R*, 1^{st} edition, Springer Singapore, Springer Nature Singapore Pte Ltd., 2019.

[86] I. S. Mohammed, M. Mamat, I. Abdulkarim and F. S. Bt. Mohamad, A survey on recent modifications of conjugate gradient methods, *Proceedings of the UniSZA Research Conference 2015 (URC 5)*, Universiti Sultan Zainal Abidin, 14–16 April 2015.
[87] H. Mohammad, M. Y. Waziri and S. A. Santos, A brief survey of methods for solving nonlinear least-squares problems, *Numer. Algebra Control Optim.*, **9** (2019), 1–13.
[88] Y. Narushima and H. Yabe, A survey of sufficient descent conjugate gradient methods for unconstrained optimization, *SUT J. Math.*, **50** (2014), 167–203.
[89] J. L. Nazareth, *Conjugate-Gradient Methods*, In: C. Floudas and P. Pardalos (eds), Encyclopedia of Optimization, 2^{nd} edition, Springer, Boston, 2009.
[90] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer, New York, 1999.
[91] W. F. H. W. Osman, M. A. H. Ibrahim and M. Mamat, Hybrid DFP-CG method for solving unconstrained optimization problems, *Journal of Physics: Conf. Series*, **890** (2017), 012033.
[92] S. Panić, M. J. Petrović and M. Mihajlov-Carević, Initial improvement of the hybrid accelerated gradient descent process, *Bull. Aust. Math. Soc.*, **98** (2018), 331–338.
[93] A. Perry, A modified conjugate gradient algorithm, *Oper. Res.*, **26** (1978), 1073–1078.
[94] M. J. Petrović, An accelerated double step-size method in unconstrained optimization, *Applied Math. Comput.*, **250** (2015), 309–319.
[95] M. J. Petrović, N. Kontrec and S. Panić, Determination of accelerated factors in gradient descent iterations based on Taylor's series, *University Thought, Publication in Natural Sciences*, **7** (2017), 41–45.
[96] M. J. Petrović, V. Rakočević, N. Kontrec, S. Panić and D. Ilić, Hybridization of accelerated gradient descent method, *Numer. Algorithms*, **79** (2018), 769–786.
[97] M. J. Petrović, V. Rakočević, D. Valjarević and D. Ilić, A note on hybridization process applied on transformed double step size model, *Numerical Algorithms*, **85** (2020), 449–465.
[98] M. J. Petrović and P. S. Stanimirović, Accelerated double direction method for solving unconstrained optimization problems, *Math. Probl. Eng.*, **2014** (2014), Article ID 965104, 8 pages.
[99] M. J. Petrović, P. S. Stanimirović, N. Kontrec and J. Mladenović, Hybrid modification of accelerated double direction method, *Math. Probl. Eng.*, **2018** (2018), Article ID 1523267, 8 pages.
[100] M. R. Peyghami, H. Ahmadzadeh and A. Fazli, A new class of efficient and globally convergent conjugate gradient methods in the Dai-Liao family, *Optim. Methods Softw.*, **30** (2015), 843–863.
[101] E. Picard, Memoire sur la theorie des equations aux derivees partielles et la methode des approximations successives, *J. Math. Pures Appl.*, **6** (1890), 145–210.
[102] E. Polak and G. Ribière, Note sur la convergence des méthodes de directions conjuguées, *Rev. Française Imformat Recherche Opértionelle*, **3** (1969), 35–43.
[103] B. T. Polyak, The conjugate gradient method in extreme problems, *USSR Comput. Math. and Math. Phys.*, **9** (1969), 94–112.
[104] M. J. D. Powell, Algorithms for nonlinear constraints that use Lagrangian functions, *Math. Programming*, **14** (1978), 224–248.
[105] M. Raydan, On the Barzilai and Borwein choice of steplength for the gradient method, *IMA J. Numer. Anal.*, **13** (1993), 321–326.
[106] M. Raydan, The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem, *SIAM J. Optim.*, **7** (1997), 26–33.
[107] M. Raydan and B. F. Svaiter, Relaxed steepest descent and Cauchy-Barzilai-Borwein method, *Comput. Optim. Appl.*, **21** (2002), 155–167.
[108] N. Shapiee, M. Rivaie, M. Mamat and P. L. Ghazali, A new family of conjugate gradient coefficient with application, *International Journal of Engineering & Technology*, **7** (2018), 36–43.
[109] Z.-J. Shi, Convergence of line search methods for unconstrained optimization, *Appl. Math. Comput.*, **157** (2004), 393–405.
[110] S. Shoid, N. Shapiee, N. Zull, N. H. A. Ghani, N. S. Mohamed, M. Rivaie and M. Mamat, The application of new conjugate gradient methods in estimating data, *International Journal of Engineering & Technology*, **7** (2018), 25–27.
[111] H. S. Sim, W. J. Leong, C. Y. Chen and S. N. I. Ibrahim, Multi-step spectral gradient methods with modified weak secant relation for large scale unconstrained optimization, *Numer. Algebra Control Optim.*, **8** (2018), 377–387.

[112] P. S. Stanimirović, B. Ivanov, S. Djordjević and I. Brajević, New hybrid conjugate gradient and Broyden-Fletcher-Goldfarb-Shanno conjugate gradient methods, *J. Optim. Theory Appl.*, **178** (2018), 860–884.
[113] P. S. Stanimirović, V. N. Katsikis and D. Pappas, Computation of $\{2,4\}$ and $\{2,3\}$-inverses based on rank-one updates, *Linear Multilinear Algebra*, **66** (2018), 147–166.
[114] P. S. Stanimirović, V. N. Katsikis and D. Pappas, Computing $\{2,4\}$ and $\{2,3\}$-inverses by using the Sherman-Morrison formula, *Appl. Math. Comput.*, **273** (2015), 584–603.
[115] P. S. Stanimirović and M. B. Miladinović, Accelerated gradient descent methods with line search, *Numer. Algor.*, **54** (2010), 503–520.
[116] P. S. Stanimirović, G. V. Milovanović, M. J. Petrović and N. Z. Kontrec, A transformation of accelerated double step-size method for unconstrained optimization, *Math. Probl. Eng.*, **2015** (2015), Article ID 283679, 8 pages.
[117] W. Sun, J. Han and J. Sun, Global convergence of nonmonotone descent methods for unconstrained optimization problems, *J. Comp. Appl. Math.*, **146** (2002), 89–98.
[118] Z. Sun and T. Sugie, Identification of Hessian matrix in distributed gradient based multi agent coordination control systems, *Numer. Algebra Control Optim.*, **9** (2019), 297–318.
[119] W. Sun and Y.-X. Yuan, *Optimization Theory and Methods: Nonlinear Programming*, 1^{st} edition, Springer, New York, 2006.
[120] Ph. L. Toint, Non–monotone trust–region algorithm for nonlinear optimization subject to convex constraints, *Math. Prog.*, **77** (1997), 69–94.
[121] D. Touati-Ahmed and C. Storey, Efficient hybrid conjugate gradient techniques, *J. Optim. Theory Appl.*, **64** (1990), 379–397.
[122] M. N. Vrahatis, G. S. Androulakis, J. N. Lambrinos and G. D. Magoulas, A class of gradient unconstrained minimization algorithms with adaptive step-size, *J. Comp. Appl. Math.*, **114** (2000), 367–386.
[123] Z. Wei, G. Li and L. Qi, New quasi-Newton methods for unconstrained optimization problems, *Appl. Math. Comput.*, **175** (2006), 1156–1188.
[124] Z. Wei, G. Li and L. Qi, New nonlinear conjugate gradient formulas for large-scale unconstrained optimization problems, *Appl. Math. Comput.*, **179** (2006), 407–430.
[125] Z. Wei, S. Yao and L. Liu, The convergence properties of some new conjugate gradient methods, *Appl. Math. Comput.*, **183** (2006), 1341–1350.
[126] P. Wolfe, Convergence conditions for ascent methods, *SIAM Rev.*, **11** (1969), 226–235.
[127] M. Xi, W. Sun and J. Chen, Survey of derivative-free optimization, *Numer. Algebra Control Optim.*, **10** (2020), 537–555.
[128] Y. Xiao and H. Zhu, A conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing, *J. Math. Anal. Appl.*, **405** (2013), 310–319.
[129] H. Yabe and M. Takano, Global convergence properties of nonlinear conjugate gradient methods with modified secant condition, *Comput. Optim. Appl.*, **28** (2004), 203–225.
[130] X. Yang, Z. Luo and X. Dai, A global convergence of LS-CD hybrid conjugate gradient method, *Adv. Numer. Anal.*, **2013** (2013), Article ID 517452, 5 pp.
[131] S. Yao, X. Lu and Z. Wei, A conjugate gradient method with global convergence for large-scale unconstrained optimization problems, *J. Appl. Math.*, **2013** (2013), Article ID 730454, 9 pp.
[132] G. Yuan and Z. Wei, Convergence analysis of a modified BFGS method on convex minimizations, *Comp. Optim. Appl.*, **47** (2010), 237–255.
[133] S. Yao, Z. Wei and H. Huang, A notes about WYL's conjugate gradient method and its applications, *Appl. Math. Comput.*, **191** (2007), 381–388.
[134] Y. Yuan, A new stepsize for the steepest descent method, *J. Comput. Math.*, **24** (2006), 149–156.
[135] G. Yuan, T. Li and W. Hu, A conjugate gradient algorithm for large-scale nonlinear equations and image restoration problems, *Appl. Numer. Math.*, **147** (2020), 129–141.
[136] G. Yuan, T. Li and W. Hu, A conjugate gradient algorithm and its application in large-scale optimization problems and image restoration, *J. Inequal. Appl.*, **2019** (2019), Article number: 247, 25 pp.
[137] G. Yuan, Z. Wei and Y. Wu, Modified limited memory BFGS method with nonmonotone line search for unconstrained optimization, *J. Korean Math. Soc.*, **47** (2010), 767–788.
[138] L. Zhang, An improved Wei-Yao-Liu nonlinear conjugate gradient method for optimization computation, *Appl. Math. Comput.*, **215** (2009), 2269–2274.

[139] Y. Zheng and B. Zheng, Two new Dai-Liao-type conjugate gradient methods for unconstrained optimization problems, *J. Optim. Theory Appl.*, **175** (2017), 502–509.
[140] H. Zhong, G. Chen and X. Guo, Semi-local convergence of the Newton-HSS method under the center Lipschitz condition, *Numer. Algebra Control Optim.*, **9** (2019), 85–99.
[141] W. Zhou and L. Zhang, A nonlinear conjugate gradient method based on the MBFGS secant condition, *Optim. Methods Softw.*, **21** (2006), 707–714.
[142] G. Zoutendijk, Nonlinear programming, computational methods, In: J. Abadie (eds.): *Integer and Nonlinear Programming*, North-Holland, Amsterdam, (1970), 37–86.
[143] N. Zull, N. 'Aini, M. Rivaie and M. Mamat, A new gradient method for solving linear regression model, *International Journal of Recent Technology and Engineering*, **7** (2019), 624–630.

Received October 2020; revised November 2020.

E-mail address: pecko@pmf.ni.ac.rs
E-mail address: ivanov.branislav@gmail.com
E-mail address: haifengma@aliyun.com
E-mail address: dijana@pmf.ni.ac.rs