Filomat 39:19 (2025), 6533–6557
https://doi.org/10.2298/FIL2519533F

Published by Faculty of Sciences and Mathematics,
University of Niš, Serbia
Available at: http://www.pmf.ni.ac.rs/filomat

On the modified conjugate-descent method and its q-variant for unconstrained optimization problems

Hao Fu[a], Jifeng Cao[a], Yu Cao[a], Tingsong Du[a,b,*]

[a]*Three Gorges Mathematical Research Center, China Three Gorges University, Yichang 443002, P. R. China*
[b]*Department of Mathematics, College of Science, China Three Gorges University,Yichang 443002, P. R. China*

Abstract. Based upon the conjugate-descent (CD) method in conjugate gradient methods (CGMs), we first propose a modified conjugate-descent (MCD) scheme, which adjusts the denominator terms of original CD scheme. Next, we apply q-calculus to the MCD method to derive its q-variant. Regardless of whether the line search technique is exact or not, both methods guarantee descent directions at every iteration. Under appropriate assumptions and the standard Wolfe line search technique, the presented methods are proven to be globally convergent. Numerical experiments demonstrate that the offered methods are efficient.

1. Introduction and preliminaries

Nonlinear CGMs are highly effective for solving unconstrained optimization problems. Due to their simple structure and low memory requirements, CGMs have been extensively studied and widely applied. In the present paper, we focus on nonlinear CGMs for the unconstrained optimization problem (UOP):

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

where $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and its gradient $g(x) = \nabla f(x)$ is computable. Starting from an initial point x_1, the CGM generates a sequence $\{x_k\}$ by the following iteration:

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

where α_k is a step length obtained by an appropriate line search technique, and d_k is the search direction of the CGM computed by

$$d_k = \begin{cases} -g_k, & k = 1, \\ -g_k + \beta_k d_{k-1}, & k \geq 2, \end{cases} \tag{3}$$

2020 *Mathematics Subject Classification*. Primary 49M37; Secondary 65K05, 90C30.
Keywords. Conjugate gradient; Sufficient descent property; Global convergence; Quantum calculus.
Received: 05 January 2023; Revised: 02 April 2025; Accepted: 13 July 2025
Communicated by Predrag Stanimirović
* Corresponding author: Tingsong Du
Email addresses: haofuctgu@163.com (Hao Fu), caojifengctgu@163.com (Jifeng Cao), yucaoctgu@163.com (Yu Cao), tingsongdu@ctgu.edu.cn (Tingsong Du)
ORCID iDs: https://orcid.org/0009-0007-8495-8089 (Hao Fu), https://orcid.org/0009-0003-4142-095X (Jifeng Cao), https://orcid.org/0009-0001-0119-4842 (Yu Cao), https://orcid.org/0000-0002-6072-1788 (Tingsong Du)

with $g_k = g(x_k)$. Here, the scalar β_k is the update parameter, and different choices for β_k lead to distinct CGMs. For a comprehensive overview of these methods, we refer readers to recent works [1, 2, 8, 20, 21] and the references therein.

In the mid-20th century, the problem about how to find an efficient method to solve n simultaneous equations with n unknowns, especially when n is very large, has puzzled a lot of scholars. Admittedly, there is no best method for all problems because the advantage of a method, to some extent, depends on the specific system to be solved. However, many researchers have successfully proposed solutions. In the early 1950s, Hestenes and Stiefel [19] introduced the conjugate gradient methods for linear systems, marking the beginning of its widespread adoption and development. Later, Fletcher and Reeves [15] proposed a quadratically convergent gradient method for finding local minima of multivariate functions. In 1969, Polak and Ribière [38] and Polyak [39] independently developed methods capable of solving linear systems in a finite number of steps. Over a decade later, Fletcher [14] restructured FR formula, leading to the CD formula. Subsequently, Liu and Storey [32] introduced a generalized CGM that accounts for inexact line search effects on conjugacy. Building on Fletcher's work, Dai and Yuan [7] implemented another CGM variant for unconstrained optimization. Here, we mention some classic formulae for β_k include Hestenes–Stiefel (HS), Fletcher–Reeves (FR), Polak–Ribière–Polyak (PRP), Conjugate-Descent (CD), Liu–Storey (LS) and Dai–Yuan (DY), which are given by the coming expressions:

$$\beta_k^{\mathrm{HS}} = \frac{g_k^T (g_k - g_{k-1})}{d_{k-1}^T (g_k - g_{k-1})}, \quad \beta_k^{\mathrm{FR}} = \frac{\|g_k\|^2}{\|g_{k-1}\|^2}, \quad \beta_k^{\mathrm{PRP}} = \frac{g_k^T (g_k - g_{k-1})}{\|g_{k-1}\|^2},$$

$$\beta_k^{\mathrm{CD}} = \frac{\|g_k\|^2}{-g_{k-1}^T d_{k-1}}, \quad \beta_k^{\mathrm{LS}} = \frac{g_k^T (g_k - g_{k-1})}{-g_{k-1}^T d_{k-1}}, \quad \beta_k^{\mathrm{DY}} = \frac{\|g_k\|^2}{d_{k-1}^T (g_k - g_{k-1})},$$

where $\|\cdot\|$ stands for the Euclidean norm.

It is generally believed that the six major CGMs are of vital importance in solving optimization problems. However, they undeniably have several deficiencies. For example, Powell [40] showed that when the objective function is non-convex, the PRP method may fail to converge even under the exact line search technique. Empirical observations suggest that the convergence properties of the HS CGM are not always reliable, while the FR and DY CGMs often perform poorly in numerical experiments. To address these limitations, numerous improvements have been proposed, primarily through two approaches: modification and hybridization.

In terms of modifications, Jiang et al. [24] discussed a modified DY CGM by altering the denominator of the original DY formula, resulting in a sufficient descent CGM for UOPs. Xiao et al. [50] extended the Broyden-Fletcher-Goldfarb-Shanno (BFGS) formula into a limited-memory framework, reducing storage and computational costs while improving performance. Another effective strategy involves adjusting the lower bound on the parameter β_k. Li & Feng [30] applied a similar adjustment to the LS CGM, yielding a descent method with significantly accelerated global convergence. Additional modifications and hybrid approaches are discussed in [13, 43, 47, 54].

Building on these modification strategies, hybrid approaches have emerged as particularly effective solutions. An example is the Jiang-Han-Jian (JHJ) CGM proposed in Ref. [23], where the scalar β_k is defined as:

$$\beta_k^{\mathrm{JHJ}} = \frac{\|g_k\|^2 - \max\{0, \frac{\|g_k\|}{\|d_{k-1}\|} g_k^T d_{k-1}, \frac{\|g_k\|}{\|g_{k-1}\|} g_k^T g_{k-1}\}}{d_{k-1}^T (g_k - g_{k-1})}. \tag{4}$$

It is evident that β_k^{JHJ} is a hybrid of β_k^{DY}, β_k^{MHS}, and β_k^{JMJ}. By taking the maximum value of the numerator, β_k^{JHJ} reduces to either β_k^{DY}, β_k^{MHS} or β_k^{JMJ}. This hybridization combines the strengths of the three methods, resulting in excellent performance in numerical experiments. Similarly, Han et al. [18] gave another hybrid method, which guarantees the sufficient descent property under any line search and demonstrates high efficiency, particularly for high-dimensional problems. Adopting a comparable approach, Fang et al. [12]

developed a modified method that is both robust and computationally efficient. These results confirm that hybrid methods can effectively inherit and enhance the performance of their original counterparts.

Before proceeding, we introduce some essential preliminaries on CGMs. Al-Baali [3] proved that the FR CGM satisfies the following sufficient descent condition under the strong Wolfe line search:

$$g_k^T d_k \leq -c \left\|g_k\right\|^2, \quad \forall\, k \geq 1,\ c > 0. \tag{5}$$

This condition is critical for the convergence analysis of many CGMs using inexact line searches.

On the other hand, in the convergence analysis and implementations of CGMs, the following two line search techniques are always used. The first one is the standard Wolfe line search, that is, to find the step length α_k satisfying that

$$\begin{cases} f\left(x_k + \alpha_k d_k\right) \leq f\left(x_k\right) + \delta \alpha_k g_k^T d_k, \\ g\left(x_k + \alpha_k d_k\right)^T d_k \geq \sigma g_k^T d_k, \end{cases} \tag{6}$$

and the other one is the strong Wolfe line search, to compute α_k satisfying that

$$\begin{cases} f\left(x_k + \alpha_k d_k\right) \leq f\left(x_k\right) + \delta \alpha_k g_k^T d_k, \\ |g\left(x_k + \alpha_k d_k\right)^T d_k| \leq \sigma |g_k^T d_k|, \end{cases} \tag{7}$$

where $0 < \delta < \sigma < 1$.

Many scholars focused on researching CGMs with various line search techniques. With exact line search, that is $g_k^T d_{k-1} = 0$, Hager & Zhang (HZ) [17] proposed the following formula

$$\beta_k^{\mathrm{HZ}} = \frac{1}{d_{k-1}^T (g_k - g_{k-1})} \left(g_k - g_{k-1} - 2 d_{k-1} \frac{\|g_k - g_{k-1}\|^2}{d_{k-1}^T (g_k - g_{k-1})} \right)^T g_k, \tag{8}$$

which reduces to a nonlinear version of the HS conjugate gradient scheme. Remarkably, this method satisfies the sufficient descent condition $g_k^T d_k \leq -\frac{7}{8} \left\|g_k\right\|^2$, regardless of whether the line search is exact or not. Further advancing this field, Alhawarat et al. [4] introduced a competitive modified version of HS CGM, and provided global convergence properties when the strong Wolfe line search or the weak one is employed. To ensure the LS method generates sufficient descent directions at every iteration, Ding [9] applied the descent backtracking line searches. In a parallel development, Mishra et al. [35] proposed a q-BFGS method under a modified Armijo–Wolfe line search involving the q-gradient. Their numerical experiments validated the method's efficacy, highlighting the potential of quantum calculus in optimization. Inspired by these results, we focus on applying quantum calculus to enhance CGMs.

In Sec. 3, we will present a modified Conjugate-Descent (MCD) method based on q-calculus. To lay the groundwork, we first review key concepts from q-calculus theory.

The q-integer $[n]$ is defined as $[n] = \frac{1-q^n}{1-q}$, for $n \in \mathbb{N}$, where $0 < q < 1$. The derivative of x^n with respect to x is given as $[n]x^{n-1}$, while for a function $f : \mathbb{R} \to \mathbb{R}$, the q-derivative is given by the coming expression, see Ref. [22]:

$$D_q f(x) = \begin{cases} \dfrac{f(x) - f(qx)}{x - qx}, & x \neq 0, q \neq 1, \\ \dfrac{\mathrm{d} f(x)}{\mathrm{d} x}, & \text{otherwise}. \end{cases} \tag{9}$$

Notably, the q-derivative is a linear operator for any constants c and d, see Ref. [25], that is

$$D_q \left\{c f_1(x) + d f_2(x)\right\} = c D_q f_1(x) + d D_q f_2(x). \tag{10}$$

For a continuous function $f(x)$ on $[a, b]$, there exists $\hat{q} \in (0, 1)$ and $x \in (a, b)$ such that

$$f(b) - f(a) = \left(D_q f\right)(x)(b - a), \tag{11}$$

for all $q \in (\hat{q}, 1)$, see Ref. [42]. When extending to multivariable functions, the first-order partial q-derivative of the function $f : \mathbb{R}^n \to \mathbb{R}$ with respect to the variable x_i is given by the coming expression, see Ref. [41]:

$$D_{q_i,x_i} f(x) = \begin{cases} \frac{1}{(1-q_i)x_i}[f(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n) \\ \quad -f(x_1, \ldots, x_{i-1}, q_i x_i, x_{i+1}, \ldots, x_n)], & x_i \neq 0, q_i \neq 1, \\ \frac{\partial}{\partial x_i} f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n), & x_i = 0, \\ \frac{\partial}{\partial x_i} f(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n), & q_i = 1, \end{cases} \tag{12}$$

where $i = 1, \ldots, n$. The q-gradient vector of f is the following, see Ref. [45]:

$$\nabla_q f(x)^T = \left[D_{q_1,x_1} f(x) \ldots D_{q_i,x_i} f(x) \ldots D_{q_n,x_n f(x)}\right], \tag{13}$$

where the parameter q is a vector $q = (q_1, \ldots, q_i, \ldots, q_n)^T \in \mathbb{R}^n$. Importantly, the sequence $\{q_{k,i}\}$ is defined by

$$q_{k+1,i} = 1 - \frac{q_{k,i}}{(k+1)^2}, \tag{14}$$

where $k \in \mathbb{N}^+$. As expected, this sequence converges to $(1, \ldots, 1)^T$ when $k \to \infty$, see Ref. [6], reducing the q-gradient to the classical derivative. For simplicity, we denote the q-gradient of f at x_k as $g_k^q = \nabla_q f(x_k)$, and a point x^* is termed a q-critical point if $g_k^q = 0$.

The q-CGMs are iterative methods that generate a sequence of iterates $\{x_k\}$ using the following update rules:

$$x_{k+1} = x_k + \alpha_k d_k^q, \tag{15}$$

and

$$d_k^q = \begin{cases} -g_k^q, & k = 1, \\ -g_k^q + \beta_k^q d_{k-1}^q, & k \geq 2. \end{cases} \tag{16}$$

Correspondingly, the quantum version of the standard Wolfe line search conditions is the following:

$$\begin{cases} f\left(x_k + \alpha_k d_k^q\right) \leq f(x_k) + \delta\alpha_k \left(g_k^q\right)^T d_k^q, \\ g^q\left(x_k + \alpha_k d_k^q\right)^T d_k^q \geq \sigma \left(g_k^q\right)^T d_k, \end{cases} \tag{17}$$

and the other is the strong one,

$$\begin{cases} f\left(x_k + \alpha_k d_k^q\right) \leq f(x_k) + \delta\alpha_k \left(g_k^q\right)^T d_k^q, \\ \left|g^q\left(x_k + \alpha_k d_k^q\right)^T d_k^q\right| \leq \sigma \left|\left(g_k^q\right)^T d_k^q\right|. \end{cases} \tag{18}$$

Recent advances in quantum methods have been made by several scholars. For example, Sana et al. [44] analysed new q-iterative methods using the q-analogue of Taylor's series and coupled system technique, demonstrating their convergence for solving nonlinear equations with high accuracy. It is valuable for their solutions to the nonlinear equations with acceptable accuracy. The framework of quantum calculus, introduced by Ernst [11], provides powerful tools for computation and classification of q-special functions.

Building on this, Gouvêa et al. [16] presented two global optimization methods that do not require ordinary derivatives: a q-analog of the steepest descent method called the q-gradient method and a q-analog of the CGM called the q-CG method, respectively.

The application of q-calculus to CGMs has shown significant promise. Lai and Mishra have contributed extensively to this area. Here, we mention some of their works. And we can refer to [26] for q-steepest descent, to [28] for q-quasi-Newton of multi-objective UOP, to [27] for q-limited memory BFGS, to [29] for q-modified FR methods, to [34] for q-Newton of multi-objective UOP, to [33] for q-PRP and to [36] for modified DY methods and so on. In parallel, CGMs have been widely applied in data and image processing. For instance, Yuan et al. [52] designed a modified CGM with sufficient descent and trust-region properties for image restoration. Later, in Ref. [53], they modified PRP CGM under weak Wolfe-Powell line search for similar applications. For further applications, see the published articles [5, 46, 48, 49, 51] and the bibliographies quoted in them.

In the remainder of this part, we present the general framework of the present paper. In Sec. 2, to improve the CD CGM, we modify the denominator terms of CD by $\max\{-g_{k-1}^T d_{k-1}, \mu_k |g_k^T d_{k-1}|\}$. We design the modified CD method and discuss its sufficient descent property. Furthermore, we prove the global convergence of the proposed method with standard Wolfe line search technique. In Sec. 3, the idea is extended in the context of quantum calculus. In Sec. 4, the numerical experiments comparing with other CGMs are provided. Finally, concluding remarks are made.

2. Method frame and convergence analysis

In this section, based upon the preceding discussion, we present our modified framework that eliminates the need for fixed line search. Our study focuses on enhancing the numerical performance of the CD method while preserving its simplicity and desirable properties. To achieve this, we take a modification to the denominator terms of the CD formula in the coming expression:

$$\beta_k^{\mathrm{MCD}} = \frac{\|g_k\|^2}{\max\{-g_{k-1}^T d_{k-1}, \mu_k |g_k^T d_{k-1}|\}}, \tag{19}$$

where $\mu_k = k^\ell + 1$ with $\ell > 1$. Compared to the classical CD method, the denominator term $-g_{k-1}^T d_{k-1}$ is replaced by $\max\{-g_{k-1}^T d_{k-1}, \mu_k |g_k^T d_{k-1}|\}$. This motivation stems from the published articles [24, 31, 36], particularly the scheme developed by Jiang & Jian [24], who proposed a modified Dai–Yuan conjugate gradient method (MDY CGM). Our approach adapts a similar but distinct strategy by introducing a variable parameter $\mu_k = k^\ell + 1$, which differentiates it from existing schemes. For brevity, we refer to the iteration method defined by (2), (3), and (19) as the MCD method. Notably, the MCD method reduces to the classical CD method under exact line search conditions (i.e., when $g_k^T d_{k-1} = 0$).

The Algorithm 1 is the process of the proposed MCD method.

The following lemma demonstrates that the search direction generated by Algorithm 1 satisfies the sufficient descent condition, regardless of the line search strategy used.

Lemma 2.1. *If the objective function $f(x)$ is continuously differentiable, then the search direction d_k in Algorithm 1 satisfies*

$$g_k^T d_k \leq -\left(1 - \frac{1}{\mu_k}\right) \|g_k\|^2, \mu_k > 1, \tag{20}$$

that is, the search direction d_k is sufficient descent. Moreover $\beta_k^{\mathrm{MCD}} > 0$.

Proof. We prove (20) by induction. For $k = 1$, $g_1^T d_1 = -\|g_1\|^2 < -\left(1 - \frac{1}{2}\right)\|g_k\|^2$. Suppose that (20) is satisfied for $k-1$. Now we prove that (20) holds true for k.

Algorithm 1 MCD Conjugate Gradient Method

Input: Objective function $f : \mathbb{R}^n \to \mathbb{R}, \epsilon$ is a tolerance for convergence. Choose an initial point $x_1 \in \mathbb{R}^n$, maximum number of iteration it_max.

$d_1 \leftarrow -g_1$.
Initialize $k = 1$.
while $k \leq it\_max$ **do**
  if $\|g_k\| \leq \epsilon$ **then**
    Stop.
  else
    Determine α_k by (6).
    Let $x_{k+1} = x_k + \alpha_k d_k$.
    Compute $g_{k+1} = g(x_{k+1})$.
    if $\|g_{k+1}\| \leq \epsilon$ **then**
      Stop.
    else
      Compute $\beta_{k+1} = \beta_{k+1}^{\text{MCD}}$.
    end if
    Let $d_{k+1} = -g_{k+1} + \beta_{k+1} d_k$.
    $k = k + 1$.
  end if
end while

Output: Minimum x^* and minimum function value $f(x^*)$.

If $g_k^T d_{k-1} = 0$, then we have that $\beta_k^{\text{MCD}} = \frac{\|g_k\|^2}{-g_{k-1}^T d_{k-1}}$,

$$
\begin{aligned}
g_k^T d_k &= g_k^T \left(-g_k + \beta_k^{\text{MCD}} d_{k-1}\right) \\
&= -\|g_k\|^2 + \beta_k^{\text{MCD}} g_k^T d_{k-1} \\
&= -\|g_k\|^2 < -\left(1 - \frac{1}{\mu_k}\right)\|g_k\|^2.
\end{aligned} \tag{21}
$$

If $g_k^T d_{k-1} \neq 0$, then it is easy to know that

$$
0 < \beta_k^{\text{MCD}} = \frac{\|g_k\|^2}{\max\{-g_{k-1}^T d_{k-1}, \mu_k \left|g_k^T d_{k-1}\right|\}} \leq \frac{\|g_k\|^2}{\mu_k \left|g_k^T d_{k-1}\right|}. \tag{22}
$$

Furthermore, we can get that

$$
\begin{aligned}
g_k^T d_k &= g_k^T \left(-g_k + \beta_k^{\text{MCD}} d_{k-1}\right) \\
&= -\|g_k\|^2 + \beta_k^{\text{MCD}} g_k^T d_{k-1} \\
&\leq -\|g_k\|^2 + \beta_k^{\text{MCD}} \left|g_k^T d_{k-1}\right| \\
&\leq -\|g_k\|^2 + \frac{\|g_k\|^2}{\mu_k} \\
&= -\left(1 - \frac{1}{\mu_k}\right)\|g_k\|^2.
\end{aligned} \tag{23}
$$

This ends the proof.

To analyze the global convergence of Algorithm 1, the following assumptions are indispensable conditions.

Assumption 2.2. *The level set* $\Lambda = \{x \in \mathbb{R}^n | f(x) \leq f(x_1)\}$ *is bounded.*

Assumption 2.3. *In a neighborhood* Λ_1 *of* Λ*, the function* $f(x)$ *is continuously differentiable and its gradient is Lipschitz continuous, namely, there exists a constant* $L > 0$ *such that* $\|g(x) - g(y)\| \leq L\|x - y\|, \forall x, y \in \Lambda_1$.

Lemma 2.4. *Postulate that Assumptions 2.2 and 2.3 both hold, and step length* α_k *satisfies the Wolfe line search conditions* (6)*. Then, we have that*

(i) $0 < \beta_k^{\text{MCD}} \leq \left[1 + \frac{1}{(k-1)^{\ell}}\right] \frac{\|g_k\|^2}{\|g_{k-1}\|^2}, \forall\, k \geq 2,$

(ii) $-\beta_k^{\text{MCD}} g_k^T d_{k-1} \leq \sigma \|g_k\|^2.$

Proof. Taking advantage of Lemma 2.1, we can get that

$$-g_{k-1}^T d_{k-1} \geq \frac{\mu_{k-1} - 1}{\mu_{k-1}} \|g_{k-1}\|^2, \tag{24}$$

that is,

$$\frac{1}{-g_{k-1}^T d_{k-1}} \leq \frac{\mu_{k-1}}{\mu_{k-1} - 1} \cdot \frac{1}{\|g_{k-1}\|^2}. \tag{25}$$

Combining with (19), we have that

$$\begin{aligned} 0 < \beta_k^{\text{MCD}} &\leq \frac{\|g_k\|^2}{-g_{k-1}^T d_{k-1}} \\ &\leq \left[1 + \frac{1}{(k-1)^{\ell}}\right] \frac{\|g_k\|^2}{\|g_{k-1}\|^2}, \end{aligned} \tag{26}$$

which concludes the proof of (i).

Now we consider (ii), if $g_k^T d_{k-1} > 0$, then it is obvious that $-\beta_k^{\text{MCD}} g_k^T d_{k-1} < 0$.

If $g_k^T d_{k-1} \leq 0$, connecting with Wolfe line search conditions, then we have that

$$\begin{aligned} -\beta_k^{\text{MCD}} g_k^T d_{k-1} &\leq \frac{\|g_k\|^2}{-g_{k-1}^T d_{k-1}} \left(-g_k^T d_{k-1}\right) \\ &\leq \frac{\|g_k\|^2}{-g_{k-1}^T d_{k-1}} \left(-\sigma g_{k-1}^T d_{k-1}\right) \\ &= \sigma \|g_k\|^2, \end{aligned} \tag{27}$$

which ends the proof.

To prove the global convergence of the MCD Algorithm 1, we need the following lemma.

Lemma 2.5. *[7] Postulate that Assumptions 2.2 and 2.3 both hold. Let* $\{x_k\}$ *be generated by Algorithm 1, where the step length* α_k *satisfies the Wolfe line search* (6)*. Then, we have that*

$$\sum_{k=1}^{\infty} \frac{\left(g_k^T d_k\right)^2}{\|d_k\|^2} < +\infty. \tag{28}$$

Theorem 2.6. *Postulate that Assumptions 2.2 and 2.3 both hold. Let* $\{x_k\}$ *be generated by Algorithm 1, where the step length* α_k *satisfies the Wolfe line search* (6)*. Then, we have that*

$$\liminf_{k \to \infty} \|g_k\| = 0. \tag{29}$$

Proof. By contradiction, it is assumed that the conclusion is not true. Then, there exists a constant $\gamma > 0$ such that $\|g_k\|^2 \geq \gamma$, $\forall k \geq 2$. From (3), it follows that $d_k = -g_k + \beta_k^{\mathrm{MCD}} d_{k-1}$, together with Lemma 2.4 implies that

$$\begin{aligned}\|d_k\|^2 &= \|g_k\|^2 - 2\beta_k^{\mathrm{MCD}} g_k^T d_{k-1} + \left(\beta_k^{\mathrm{MCD}}\right)^2 \|d_{k-1}\|^2 \\ &\leq \|g_k\|^2 + 2\sigma\|g_k\|^2 + \left[1 + \frac{1}{(k-1)^\ell}\right]^2 \frac{\|g_k\|^4}{\|g_{k-1}\|^4}\|d_{k-1}\|^2 \\ &= (1+2\sigma)\|g_k\|^2 + \left[1 + \frac{1}{(k-1)^\ell}\right]^2 \frac{\|g_k\|^4}{\|g_{k-1}\|^4}\|d_{k-1}\|^2. \end{aligned} \tag{30}$$

Dividing both sides of (30) by $\|g_k\|^4$, we get that

$$\frac{\|d_k\|^2}{\|g_k\|^4} \leq \left[1 + \frac{1}{(k-1)^\ell}\right]^2 \frac{\|d_{k-1}\|^2}{\|g_{k-1}\|^4} + (1+2\sigma)\|g_k\|^{-2}. \tag{31}$$

For the sake of convenience, let us define $\theta_k = \frac{\|d_k\|^2}{\|g_k\|^4}$, and we have that

$$\begin{aligned}\theta_k &\leq \left[1 + \frac{1}{(k-1)^\ell}\right]^2 \theta_{k-1} + (1+2\sigma)\|g_k\|^{-2} \\ &\leq \prod_{n=2}^{k}\left[1 + \frac{1}{(n-1)^\ell}\right]^2 \theta_1 + (1+2\sigma)\left[\|g_k\|^{-2} + \sum_{n=1}^{k-2}\prod_{m=n+1}^{k-1}\left[1 + \frac{1}{m^\ell}\right]^2 \cdot \|g_{n+1}\|^{-2}\right]. \end{aligned} \tag{32}$$

Let $\lambda_k = \prod_{n=2}^{k}\left[1 + \frac{1}{(n-1)^\ell}\right]^2$, it is easy to see that λ_k and $\ln\lambda_k = 2\sum_{n=2}^{k}\ln\left[1 + \frac{1}{(n-1)^\ell}\right]$ have the same convergence and divergence properties. We now prove $\ln\lambda_k = 2\sum_{n=2}^{k}\ln\left[1 + \frac{1}{(n-1)^\ell}\right]$ is convergent. Owing to the fact that we have

$$\lim_{n\to\infty}\frac{\ln\left[1 + \frac{1}{(n-1)^\ell}\right]}{\frac{1}{(n-1)^\ell}} = 1, \tag{33}$$

thus one can know that $\sum_{n=2}^{\infty}\ln\left[1 + \frac{1}{(n-1)^\ell}\right]$ and $\sum_{n=2}^{\infty}\frac{1}{(n-1)^\ell}$ have the same convergence property. Obviously, $\sum_{n=2}^{\infty}\frac{1}{(n-1)^\ell}$ is convergent($\ell > 1$), which means that $\sum_{n=2}^{\infty}\ln\left[1 + \frac{1}{(n-1)^\ell}\right]$ is convergent. Then, it is easy to know that $\ln\lambda_k$ is also convergent. Further, the sequence $\{\lambda_k\}$ increases monotonically and has a limit.

Let $\lim_{k\to\infty}\lambda_k = \lambda$, we can know that $1 < \lambda_k < \lambda$ $(k \geq 2)$. Hence, we have that

$$\begin{aligned}\theta_k &\leq \lambda\theta_1 + (1+2\sigma)\lambda \cdot \sum_{n=1}^{k-1}\|g_{n+1}\|^{-2} \\ &\leq \lambda\theta_1 + (1+2\sigma)\lambda \cdot \frac{k-1}{\gamma}. \end{aligned} \tag{34}$$

From (20), we get that

$$-g_k^T d_k \geq \left(1 - \frac{1}{\mu_k}\right)\|g_k\|^2 \geq \frac{1}{2}\|g_k\|^2, \tag{35}$$

that is,

$$\left(g_k^T d_k\right)^2 \geq \frac{1}{4}\|g_k\|^4. \tag{36}$$

Together with (34), we can find out that

$$\begin{aligned}\sum_{k=1}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} &\geq \sum_{k=1}^{\infty} \frac{1}{4} \cdot \frac{\|g_k\|^4}{\|d_k\|^2} = \frac{1}{4}\sum_{k=1}^{\infty} \frac{1}{\theta_k} \\ &\geq \frac{1}{4}\sum_{k=1}^{\infty} \frac{1}{\frac{(1+2\sigma)\lambda}{\gamma}(k-1) + \lambda\theta_1} = +\infty.\end{aligned} \tag{37}$$

It contradicts (28), and therefore the desired result holds true. This ends the proof.

3. A q-variant of modified conjugate-descent method

From the above analysis, it is evident that the MCD method imposes strong requirements (e.g., derivability and differentiability) on the objective function. To address this limitation, we propose a q-variant of Algorithm 1. Departing from conventional gradient-based approaches, Mishra et al. [36] extended the MDY method using quantum calculus. Adopting a similar approach, we generalize the modified Conjugate-Descent CGM via the following update rule:

$$\beta_k^{q-\mathrm{MCD}} = \frac{\|g_k^q\|^2}{\max\left\{-\left(g_{k-1}^q\right)^T d_{k-1}^q, \mu_k \left|\left(g_k^q\right)^T d_{k-1}^q\right|\right\}}. \tag{38}$$

For brevity, we refer to the method defined by (15), (16) and (38) as the q-MCD method. If exact line search is used, that is, the line search ensures that $\left(g_k^q\right)^T d_{k-1}^q = 0$, then the $\beta_k^{q-\mathrm{MCD}}$ reduces to the $\beta_k^{q-\mathrm{CD}}$, namely

$$\beta_k^{q-\mathrm{CD}} = \frac{\|g_k^q\|^2}{-\left(g_{k-1}^q\right)^T d_{k-1}^q}. \tag{39}$$

The Algorithm 2 is the process of the q-MCD method.

Lemma 3.1. *If the objective function $f(x)$ is continuously differentiable, then the search direction d_k^q in Algorithm 2 satisfies*

$$\left(g_k^q\right)^T d_k^q \leq -\left(1 - \frac{1}{\mu_k}\right)\|g_k^q\|^2, \tag{40}$$

that is, the search direction d_k^q is sufficient descent. Moreover $\beta_k^{q-\mathrm{MCD}} > 0$.

Proof. We prove (40) by induction. For $k = 1$, $\left(g_1^q\right)^T d_1^q = -\|g_1^q\|^2 < -\left(1 - \frac{1}{2}\right)\|g_k^q\|^2$. It is assumed that (40) is satisfied for $k-1$. Now we prove that (40) holds true for k.

If $\left(g_k^q\right)^T d_{k-1}^q = 0$, then we have that $\beta_k^{q-\mathrm{MCD}} = \frac{\|g_k^q\|^2}{-\left(g_{k-1}^q\right)^T d_{k-1}^q}$,

$$\begin{aligned}\left(g_k^q\right)^T d_k^q &= \left(g_k^q\right)^T \left(-g_k^q + \beta_k^{q-\mathrm{MCD}} d_{k-1}^q\right) \\ &= -\|g_k^q\|^2 + \beta_k^{q-\mathrm{MCD}} \left(g_k^q\right)^T d_{k-1}^q \\ &= -\|g_k^q\|^2 < -\left(1 - \frac{1}{\mu_k}\right)\|g_k^q\|^2.\end{aligned} \tag{41}$$

Algorithm 2 q–MCD Conjugate Gradient Method

Input: Objective function $f : \mathbb{R}^n \to \mathbb{R}, \epsilon$ is a tolerance for convergence. Choose an initial point $x_1 \in \mathbb{R}^n$, $q_1^1 \in (0, 1)$, the dimension of the given vector q is n, maximum number of iteration is *it_max*.

$d_1^q \leftarrow -g_1^q$.
Initialize $k = 1$.
while $k \le it\_max$ **do**
  if $\left\|g_k^q\right\| \le \epsilon$ **then**
    Stop.
  else
    Determine α_k by (17).
    Let $x_{k+1} = x_k + \alpha_k d_k^q$.
    for $i = 1, ..., n$ **do**
      $q_{k+1,i} = 1 - \frac{q_{k,i}}{(k+1)^2}$.
    end for
    Compute $g_{k+1}^q = g^q(x_{k+1})$.
    if $\left\|g_{k+1}^q\right\| \le \epsilon$ **then**
      Stop.
    else
      Compute $\beta_{k+1}^q = \beta_{k+1}^{q-\mathrm{MCD}}$.
    end if
    Let $d_{k+1}^q = -g_{k+1}^q + \beta_{k+1}^q d_k^q$.
    $k = k + 1$.
  end if
end while

Output: Minimum x^* and minimum function value $f(x^*)$.

If $\left(g_k^q\right)^T d_{k-1}^q \neq 0$, then it is easy to know that

$$0 < \beta_k^{q-\mathrm{MCD}} = \frac{\|g_k^q\|^2}{\max\left\{-\left(g_{k-1}^q\right)^T d_{k-1}^q, \mu_k \left|\left(g_k^q\right)^T d_{k-1}^q\right|\right\}} \le \frac{\|g_k^q\|^2}{\mu_k \left|\left(g_k^q\right)^T d_{k-1}^q\right|}. \tag{42}$$

Furthermore, we can get that

$$\begin{aligned}
\left(g_k^q\right)^T d_k^q &= \left(g_k^q\right)^T \left(-g_k^q + \beta_k^{q-\mathrm{MCD}} d_{k-1}^q\right) \\
&= -\|g_k^q\|^2 + \beta_k^{q-\mathrm{MCD}} \left(g_k^q\right)^T d_{k-1}^q \\
&\le -\|g_k^q\|^2 + \beta_k^{q-\mathrm{MCD}} \left|\left(g_k^q\right)^T d_{k-1}^q\right| \\
&\le -\|g_k^q\|^2 + \frac{\|g_k^q\|^2}{\mu_k \left|\left(g_k^q\right)^T d_{k-1}^q\right|} \cdot \left|\left(g_k^q\right)^T d_{k-1}^q\right| \\
&= -\left(1 - \frac{1}{\mu_k}\right)\|g_k^q\|^2.
\end{aligned} \tag{43}$$

This ends the proof.

To analyze the global convergence of Algorithm 2, the following assumption is necessary.

Assumption 3.2. *In a neighborhood Λ_2 of Λ, the function $f(x)$ is continuously differentiable and its q-gradient is Lipschitz continuous, namely, there exists a constant $L > 0$ such that $\|g^q(x) - g^q(y)\| \le L\|x - y\|, \forall x, y \in \Lambda_2$.*

Lemma 3.3. *Postulate that Assumptions 2.2 and 3.2 both hold, and step length α_k satisfies the Wolfe line search conditions* (17)*. Then, we have that*

(*i*) $0 < \beta_k^{q-\mathrm{MCD}} \leq \left[1 + \frac{1}{(k-1)^{\ell}}\right] \frac{\|g_k^q\|^2}{\|g_{k-1}^q\|^2}, \forall\, k \geq 2,$

(*ii*) $-\beta_k^{q-\mathrm{MCD}} \left(g_k^q\right)^T d_{k-1}^q \leq \sigma \|g_k^q\|^2.$

Proof. From (40), we can get that

$$-\left(g_k^q\right)^T d_k^q \geq \left(1 - \frac{1}{\mu_k}\right) \|g_k^q\|^2, \tag{44}$$

that is

$$\frac{1}{-\left(g_k^q\right)^T d_k^q} \leq \frac{\mu_k}{\mu_k - 1} \cdot \frac{1}{\|g_k^q\|^2}, \tag{45}$$

i.e.

$$\frac{1}{-\left(g_{k-1}^q\right)^T d_{k-1}^q} \leq \frac{\mu_{k-1}}{\mu_{k-1} - 1} \cdot \frac{1}{\|g_{k-1}^q\|^2}. \tag{46}$$

Combining with (38), we deduce that

$$\begin{aligned} 0 < \beta_k^{q-\mathrm{MCD}} &\leq \frac{\|g_k^q\|^2}{-\left(g_{k-1}^q\right)^T d_{k-1}^q} \\ &\leq \frac{\mu_{k-1}}{\mu_{k-1} - 1} \cdot \frac{1}{\|g_{k-1}^q\|^2} \cdot \|g_k^q\|^2 \\ &\leq \left[1 + \frac{1}{(k-1)^{\ell}}\right] \frac{\|g_k^q\|^2}{\|g_{k-1}^q\|^2}, \end{aligned} \tag{47}$$

which ends the proof of the result (*i*).

Now we prove the result (*ii*), if $\left(g_k^q\right)^T d_{k-1}^q > 0$, then it is obvious that $-\beta_k^{q-\mathrm{MCD}} \left(g_k^q\right)^T d_{k-1}^q < 0 \leq \sigma \|g_k^q\|^2$. If $\left(g_k^q\right)^T d_{k-1}^q \leq 0$, connecting with Wolfe line search conditions (17), then we derive that

$$\begin{aligned} -\beta_k^{q-\mathrm{MCD}} \left(g_k^q\right)^T d_{k-1}^q &\leq \frac{\|g_k^q\|^2}{-\left(g_{k-1}^q\right)^T d_{k-1}^q} \left[-\left(g_k^q\right)^T d_{k-1}^q\right] \\ &\leq \frac{\|g_k^q\|^2}{-\left(g_{k-1}^q\right)^T d_{k-1}^q} \left[-\sigma \left(g_{k-1}^q\right)^T d_{k-1}^q\right] \\ &= \sigma \|g_k^q\|^2. \end{aligned} \tag{48}$$

This concludes the proof.

Lemma 3.4. *Postulate that Assumptions 2.2 and 3.2 both hold. Let $\{x_k\}$ be generated by Algorithm 2, where the step length α_k satisfies the Wolfe line search* (17)*. Then, we have that*

$$\sum_{k=1}^{\infty} \frac{\left[\left(g_k^q\right)^T d_k^q\right]^2}{\left\|d_k^q\right\|^2} < +\infty. \tag{49}$$

Proof. In views of (17) and Assumption 3.2, we have that

$$(\sigma - 1)\left(g_k^q\right)^T d_k^q \le \left(d_k^q\right)^T (g_{k+1}^q - g_k^q) \le \alpha_k L \|d_k^q\|^2, \tag{50}$$

from which, we know that

$$\alpha_k \ge \frac{\sigma - 1}{L} \cdot \frac{\left(g_k^q\right)^T d_k^q}{\|d_k^q\|^2}. \tag{51}$$

Then, we have that

$$\begin{aligned} f_k - f_{k+1} &\ge -\delta\alpha_k \left(g_k^q\right)^T d_k^q \\ &\ge -\frac{\delta(\sigma-1)}{L} \cdot \frac{\left(g_k^q\right)^T d_k^q}{\|d_k^q\|^2} \cdot \left(g_k^q\right)^T d_k^q \\ &= \frac{\delta(1-\sigma)}{L} \cdot \frac{\left[\left(g_k^q\right)^T d_k^q\right]^2}{\|d_k^q\|^2}. \end{aligned} \tag{52}$$

Summing up both sides of the inequality (52) from $k = 1$ to ∞, we deduce that

$$\sum_{k=1}^{\infty}(f_k - f_{k+1}) = f_1 - \lim_{k\to\infty} f_k \ge \sum_{k=1}^{\infty} \frac{\delta(1-\sigma)}{L} \cdot \frac{\left[\left(g_k^q\right)^T d_k^q\right]^2}{\|d_k^q\|^2}, \tag{53}$$

along with the Assumption 2.2 leads to

$$\sum_{k=1}^{\infty} \frac{\left[\left(g_k^q\right)^T d_k^q\right]^2}{\|d_k^q\|^2} < +\infty. \tag{54}$$

This ends the proof.

Theorem 3.5. *Postulate that Assumptions 2.2 and 3.2 both hold. Let $\{x_k\}$ be generated by Algorithm 2, where the step length α_k satisfies the Wolfe line search* (17)*. Then, we have that*

$$\liminf_{k\to\infty} \|g_k^q\| = 0. \tag{55}$$

Proof. By contradiction, it is assumed that the conclusion is not true. Then, there exists a constant $c > 0$ such that $\|g_k^q\|^2 \ge c$, $\forall k \ge 2$. From (16), it follows that $d_k^q = -g_k^q + \beta_k^{q-\mathrm{MCD}} d_{k-1}^q$, together with Lemma 3.3 implies that

$$\begin{aligned} \|d_k^q\|^2 &= \|g_k^q\|^2 - 2\beta_k^{q-\mathrm{MCD}}\left(g_k^q\right)^T d_{k-1}^q + \left(\beta_k^{q-\mathrm{MCD}}\right)^2 \|d_{k-1}^q\|^2 \\ &\le \|g_k^q\|^2 + 2\sigma\|g_k^q\|^2 + \left[1 + \frac{1}{(k-1)^\ell}\right]^2 \frac{\|g_k^q\|^4}{\|g_{k-1}^q\|^4}\|d_{k-1}^q\|^2 \\ &= (1+2\sigma)\|g_k^q\|^2 + \left[1 + \frac{1}{(k-1)^\ell}\right]^2 \frac{\|g_k^q\|^4}{\|g_{k-1}^q\|^4}\|d_{k-1}^q\|^2. \end{aligned} \tag{56}$$

Dividing both sides of (56) by $\|g_k^q\|^4$, we get that

$$\frac{\|d_k^q\|^2}{\|g_k^q\|^4} \le \left[1 + \frac{1}{(k-1)^\ell}\right]^2 \frac{\|d_{k-1}^q\|^2}{\|g_{k-1}^q\|^4} + (1+2\sigma)\|g_k^q\|^{-2}. \tag{57}$$

For the sake of convenience, let us define $\eta_k = \frac{\|d_k^q\|^2}{\|g_k^q\|^4}$, and we have that

$$\begin{aligned}\eta_k &\le \left[1 + \frac{1}{(k-1)^\ell}\right]^2 \eta_{k-1} + (1+2\sigma)\|g_k^q\|^{-2} \\ &\le \prod_{n=2}^{k}\left[1 + \frac{1}{(n-1)^\ell}\right]^2 \eta_1 + (1+2\sigma)\left[\|g_k^q\|^{-2} + \sum_{n=1}^{k-2}\prod_{m=n+1}^{k-1}\left[1+\frac{1}{m^\ell}\right]^2 \cdot \|g_{n+1}^q\|^{-2}\right]. \end{aligned} \tag{58}$$

Let $\varphi_k = \prod_{n=2}^{k}\left[1 + \frac{1}{(n-1)^\ell}\right]^2$, it is easy to see that φ_k and $\ln\varphi_k = 2\sum_{n=2}^{k}\ln\left[1 + \frac{1}{(n-1)^\ell}\right]$ have the same convergence and divergence properties. Since $\lim_{k\to\infty}\ln\varphi_k = 2\sum_{n=2}^{\infty}\ln\left[1 + \frac{1}{(n-1)^\ell}\right]$ is convergent ($\ell > 1$), it is simple to know that the sequence $\{\varphi_k\}$ increases monotonically and has a limit.

Let $\lim_{k\to\infty}\varphi_k = \varphi$, we can know that $1 < \varphi_k < \varphi$ ($k \ge 2$). Thus, we have that

$$\begin{aligned}\eta_k &\le \varphi\eta_1 + (1+2\sigma)\varphi \cdot \sum_{n=1}^{k-1}\|g_{n+1}^q\|^{-2} \\ &\le \varphi\eta_1 + (1+2\sigma)\varphi \cdot \frac{k-1}{c}.\end{aligned} \tag{59}$$

From (40), we get that

$$-\left(g_k^q\right)^T d_k^q \ge \left(1 - \frac{1}{\mu_k}\right)\|g_k^q\|^2 \ge \frac{1}{2}\|g_k^q\|^2, \tag{60}$$

that is

$$\left[\left(g_k^q\right)^T d_k^q\right]^2 \ge \frac{1}{4}\|g_k^q\|^4, \tag{61}$$

together with (59), we can know that

$$\begin{aligned}\sum_{k=1}^{\infty}\frac{\left[\left(g_k^q\right)^T d_k^q\right]^2}{\|d_k^q\|^2} &\ge \sum_{k=1}^{\infty}\frac{1}{4}\cdot\frac{\|g_k^q\|^4}{\|d_k^q\|^2} = \frac{1}{4}\sum_{k=1}^{\infty}\frac{1}{\eta_k} \\ &\ge \frac{1}{4}\sum_{k=1}^{\infty}\frac{1}{\frac{(1+2\sigma)\varphi}{c}(k-1) + \varphi\eta_1} = +\infty.\end{aligned} \tag{62}$$

It contradicts (49), therefore, the desired result holds true. This ends the proof.

4. Numerical results

In this section, we evaluate the computational performance of the proposed MCD and q-MCD CGMs by solving a set of benchmark problems provided by Moré et al. in Ref. [37]. We compare their performance against the classical CD method. In all CGMs, the step size α_k is computed using the Wolfe line search conditions (6) or (17). The algorithms were implemented in MATLAB R2021a (version 9.10) and executed on a Dell workstation equipped with a 1.80 GHz Intel Core i7 processor, 8.00 GB RAM, and Windows 10.

Given the global convergence guarantees of the MCD and q-MCD methods, we initialize parameters randomly within the feasible domain of each test function. The algorithmic parameters are set as follows: $\delta = 0.1, \sigma = 0.2, \ell = 1.1$ with $q_{1,i} = 0.9$ ($i = 1, ..., n$). The proposed methods stop if one of the following conditions is satisfied: (1) $\|g_k\| \leq 10^{-6}$ or $\|g_k^q\| \leq 10^{-6}$; (2) the number of iteration $Nit > 1000$. Cases where condition (2) is triggered are marked as failures (denoted by "F"). The numerical results, including iterations (N_{it}), gradient evaluations (NG), function evaluations (NF), and computation time (TR in seconds), are summarized in Table 7.

Example 4.1. *Consider the Rose function* $f : \mathbb{R}^2 \to \mathbb{R}$ *be defined as*

$$f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2.$$

We optimize this function employing the MCD Algorithm 1 and q-MCD Algorithm 2. It is clearly seen that there is a unique global minimum $(1,1)^T$, and we choose the initial point $(2,1)^T$ for three methods. We have shown the numerical results of CD in Table 1, numerical results of MCD in Table 2 and numerical results of q-MCD in Table 3, respectively.

Under identical conditions, the MCD method achieves the approximate global minimum in fewer iterations compared to both the CD and q-MCD methods for Example 4.1. While the q-MCD requires more iterations than MCD, it relaxes the differentiability requirements on the objective function, thereby broadening its applicability to non-smooth or quantum-calculus-based problems.

Table 1: Detailed Computation for Rose Function on CD CGM

k-th	Step length	x^T	$f(x)$	g_k^T
1	0.00040070	(1.03752085, 1.24041944)	2.69002126	(2402, −600)
2	0.00082468	(1.09173700, 1.21383459)	0.05657356	(−67.97384245, 4.38898098)
4	0.00068271	(1.09971979, 1.21003510)	0.00998648	(−1.16503272, 0.62006208)
5	0.00095090	(1.09981833, 1.20990298)	0.00997286	(−0.08714040, 0.13029682)
6	0.06789796	(1.09739940, 1.20298203)	0.00965653	(0.06650617, 0.06052385)
7	0.01749912	(1.03607094, 1.07047236)	0.00218358	(0.76694546, −0.26068296)
8	0.00200149	(1.01330534, 1.02810884)	0.00035157	(1.30326012, −0.59412835)
9	0.00126407	(1.01184619, 1.02386256)	0.00014042	(−0.50887404, 0.26422673)
10	0.00334235	(1.01180520, 1.02383623)	0.00014011	(0.01160773, 0.00597158)
11	0.27145378	(1.00736070, 1.01430075)	7.67268E−05	(−0.01138559, 0.01729384)
12	0.00223658	(1.00242306, 1.00490839)	6.18934E−06	(0.20605364, −0.09496709)
13	0.00124619	(1.00242347, 1.00485300)	5.87323E−06	(−0.01776971, 0.01128058)
14	0.00202580	(1.00241383, 1.00484888)	5.85028E−06	(0.00477500, 0.00003589)
15	0.01320632	(1.00240048, 1.00479494)	5.77615E−06	(−0.00134402, 0.00307841)
16	0.03142235	(1.00182862, 1.00376776)	4.49262E−06	(0.00952225, −0.00235500)
17	0.00439114	(1.00033877, 1.00068763)	1.24704E−07	(−0.03929362, 0.02143623)
18	0.00111587	(1.00033915, 1.00067855)	1.15026E−07	(−0.00331084, 0.00199352)
19	0.00185633	(1.00033800, 1.00067816)	1.14661E−07	(0.00062618, 2.6055E−05)
20	0.02277209	(1.00033453, 1.00066662)	1.12553E−07	(−0.00013903, 0.00040738)
21	0.03628880	(1.00018055, 1.00037726)	5.86059E−08	(0.00168572, −0.00050816)
22	0.00206577	(1.00005885, 1.00011824)	3.49212E−09	(−0.00609084, 0.00322539)
23	0.00131323	(1.00005894, 1.00011802)	3.47574E−09	(−9.82719E−05, 0.00010798)
24	0.01392860	(1.00005824, 1.00011720)	3.44234E−09	(6.37464E−05, 2.70651E−05)
25	0.04652736	(1.00004264, 1.00008304)	2.32159E−09	(−0.00016716, 0.00014181)
26	0.00401569	(1.00000601, 1.00001327)	1.95131E−10	(0.00098259, −0.00044863)

(continued)

k-th	Step length	x^T	$f(x)$	g_k^T
27	0.00130651	(1.00000353, 1.00000700)	$1.27792E{-}11$	(−0.00049248, 0.00025224)
28	0.00112854	(1.00000349, 1.00000699)	$1.22003E{-}11$	($2.98252E{-}05$, $-1.13834E{-}05$)
29	0.00438311	(1.00000349, 1.00000698)	$1.21675E{-}11$	($-6.49752E{-}07$, $3.81254E{-}06$)
30	0.16089944	(1.00000248, 1.00000512)	$8.66747E{-}12$	($6.59326E{-}06$, $1.9155E{-}07$)
31	0.00382559	(1.00000027, 1.00000050)	$1.55079E{-}13$	($-5.84748E{-}05$, $3.17177E{-}05$)
32	0.00113617	(1.00000023, 1.00000046)	$5.2092E{-}14$	($1.21143E{-}05$, $-5.7902E{-}06$)

Table 2: Detailed Computation for Rose Function on MCD CGM

k-th	Step length	x^T	$f(x)$	g_k^T
1	0.00040070	(1.03752085, 1.24041944)	2.69002126	(2402, −600)
2	0.00082468	(1.09173700, 1.21383459)	0.05657356	(−67.97384245, 32.79398386)
3	0.00066326	(1.09882066, 1.21050716)	0.01072672	(−9.39975188, 4.38898098)
4	0.00068271	(1.09971979, 1.21003510)	0.00998648	(−1.16503272, 0.62006208)
5	0.00095090	(1.09981833, 1.20990298)	0.00997286	(−0.08714040, 0.13029682)
6	0.06789796	(1.09739940, 1.20298203)	0.00965653	(0.06650617, 0.06052385)
7	0.01749912	(1.03607094, 1.07047236)	0.00218358	(0.76694546, −0.26068296)
8	0.00200149	(1.01330534, 1.02810884)	0.00035157	(1.30326012, −0.59412835)
9	0.00126407	(1.01184619, 1.02386256)	0.00014042	(−0.50887404, 0.26422673)
10	0.00324483	(1.01180676, 1.02383807)	0.00014012	(0.01160773, 0.00597158)
11	0.32898405	(1.00662053, 1.01279684)	$6.76505E{-}05$	(−0.01084604, 0.01702873)
12	0.00182782	(1.00253926, 1.00510265)	$6.47912E{-}06$	(0.20975270, −0.09760959)
13	0.00154881	(1.00254201, 1.00509648)	$6.46541E{-}06$	(−0.00201586, 0.00353821)
14	0.15719244	(1.00225694, 1.00460265)	$5.79413E{-}06$	(0.00267995, 0.00119899)
15	0.00908605	(1.00037520, 1.00073389)	$1.68517E{-}07$	(−0.02903689, 0.01673760)
16	0.00110923	(1.00036025, 1.00072375)	$1.30756E{-}07$	(0.00741493, −0.00333101)
17	0.00143524	(1.00036082, 1.00072273)	$1.30285E{-}07$	(−0.00052958, 0.00062481)
18	0.03776112	(1.00035147, 1.00070930)	$1.27416E{-}07$	(0.00034007, 0.00019072)
19	0.04001791	(1.00011292, 1.00021388)	$2.71009E{-}08$	(−0.00179059, 0.00124633)
20	0.00172308	(1.00005405, 1.00011370)	$6.0515E{-}09$	(0.00501800, −0.00239581)
21	0.00107369	(1.00005530, 1.00011073)	$3.05984E{-}09$	(−0.00212983, 0.00111890)
22	0.00265055	(1.00005515, 1.00011066)	$3.05435E{-}09$	($5.89033E{-}05$, $2.58476E{-}05$)
23	1	(0.99999909, 0.99999434)	$1.4749E{-}09$	($-3.49038E{-}05$, $7.25947E{-}05$)
24	0.00102930	(0.99999586, 0.99999171)	$1.71328E{-}11$	(0.00153392, −0.00076787)
25	0.02170468	(0.99999591, 0.99999178)	$1.69779E{-}11$	($-2.38125E{-}06$, $-2.94594E{-}06$)
26	0.11579993	(1.00000000, 1.00000000)	$1.14969E{-}16$	($1.33298E{-}05$, $-1.07501E{-}05$)

Table 3: Detailed Computation for Rose Function on q-MCD CGM

k-th	Step length	x^T	$f(x)$	$\left(g_k^q\right)^T$	q_k^T
1	0.00026210	(1.40188397, 1.15726101)	65.45076257	(2282, -600)	(0.1, 0.1)
2	0.00055473	(1.11443665, 1.25958586)	0.04413100	(431.24762201, −161.6035295)	(0.975, 0.975)
3	0.00120657	(1.11930001, 1.25536028)	0.01487145	(−4.09035032, 3.52336486)	(0.89166667, 0.89166667)
4	0.00066621	(1.11998138, 1.25494025)	0.01442940	(−0.87898678, 0.50555412)	(0.94427083, 0.94427083)
5	0.00268711	(1.12002987, 1.25460858)	0.01440918	(−0.00663080, 0.11639285)	(0.96222917, 0.96222917)
6	0.45212088	(1.05776031, 1.11615740)	0.00406496	(0.17835663, 0.02833483)	(0.97327141, 0.97327141)
7	0.00166629	(1.04334431, 1.08958385)	0.00198206	(1.23610447, −0.53989260)	(0.98013732, 0.98013732)
8	0.00132552	(1.04321171, 1.08825906)	0.00186735	(−0.33186410, 0.20329922)	(0.98468535, 0.98468535)
9	0.00206074	(1.04299465, 1.08815170)	0.00185839	(0.09947932, −0.00632032)	(0.98784339, 0.98784339)
10	0.00346767	(1.04300153, 1.08786206)	0.00184914	(−0.04394533, 0.06276986)	(0.99012157, 0.99012157)
11	0.01205798	(1.04204395, 1.08656969)	0.00181869	(0.08191227, 0.00197306)	(0.99181718, 0.99181718)
12	0.00186014	(1.04197575, 1.08567954)	0.00176208	(−0.21209515, 0.14282182)	(0.99311238, 0.99311238)
13	0.00260331	(1.04170531, 1.08549857)	0.00175149	(0.09803858, −0.00678756)	(0.99412360, 0.99412360)
14	0.00185990	(1.04175280, 1.08532432)	0.00174386	(−0.06134983, 0.06972255)	(0.99492794, 0.99492794)
15	0.01003830	(1.04132245, 1.08483042)	0.00173039	(0.05216708, 0.01508557)	(0.99557810, 0.99557810)
16	0.00387407	(1.04104130, 1.08362191)	0.00168649	(−0.11594204, 0.09559567)	(0.99611102, 0.99611102)
17	0.00441614	(1.04015691, 1.08265090)	0.00166507	(0.14236076, −0.02901505)	(0.99655325, 0.99655325)
18	0.00098068	(1.04032293, 1.08245361)	0.00162924	(−0.22054082, 0.14490159)	(0.99692422, 0.99692422)
19	0.00748258	(1.04029590, 1.08216816)	0.00162398	(0.00511404, 0.03636477)	(0.99723844, 0.99723844)
20	0.20076340	(1.01492810, 1.02878882)	0.00038932	(0.10028421, −0.00947939)	(0.99750690, 0.99750690)
21	0.00117382	(1.01262104, 1.02560290)	0.00016335	(0.55292653, −0.25804467)	(0.99773808, 0.99773808)
22	0.00122496	(1.01266013, 1.02551229)	0.00016038	(−0.05628058, 0.04030354)	(0.99793856, 0.99793856)
23	0.00575377	(1.01259591, 1.02545829)	0.00015982	(0.01246950, 0.00635223)	(0.99811354, 0.99811354)
24	0.08969722	(1.01019563, 1.02011526)	0.00011839	(−0.01843691, 0.02156524)	(0.99826716, 0.99826716)
25	0.00128850	(1.00966286, 1.01952565)	$9.45063E{-}05$	(0.17377539, −0.07598971)	(0.99840277, 0.99840277)
26	0.00145815	(1.00968088, 1.01947632)	$9.37629E{-}05$	(−0.02367151, 0.02131133)	(0.99852307, 0.99852307)
27	0.00698498	(1.00961607, 1.01941519)	$9.32893E{-}05$	(0.01095350, 0.00416715)	(0.99863028, 0.99863028)
28	0.04129612	(1.00858832, 1.01702303)	$7.89292E{-}05$	(−0.01732156, 0.01811615)	(0.99872624, 0.99872624)
29	0.00228711	(1.00806400, 1.01648605)	$7.36148E{-}05$	(0.10884438, −0.04547473)	(0.99881245, 0.99881245)
30	0.00096054	(1.00815402, 1.01642166)	$6.67102E{-}05$	(−0.10195444, 0.05860624)	(0.99889021, 0.99889021)
31	0.00204156	(1.00816001, 1.01640206)	$6.66096E{-}05$	(−0.00268574, 0.00942568)	(0.99896057, 0.99896057)
32	1.00000000	(1.00138923, 1.00244392)	$1.32512E{-}05$	(0.01008738, 0.00309281)	(0.99902445, 0.99902445)
33	0.00117170	(1.00073488, 1.00150589)	$6.66762E{-}07$	(0.13748349, −0.06729405)	(0.99908262, 0.99908262)
34	0.00103441	(1.00074623, 1.00149585)	$5.57663E{-}07$	(−0.01277281, 0.00711952)	(0.99913574, 0.99913574)
35	0.38128011	(1.00062050, 1.00127081)	$4.71559E{-}07$	(0.00035329, 0.00056942)	(0.99918438, 0.99918438)
36	0.00566253	(1.00008006, 1.00016336)	$7.46248E{-}09$	(−0.01052794, 0.00588336)	(0.99922902, 0.99922902)
37	0.00101487	(1.00008103, 1.00016234)	$6.57347E{-}09$	(−0.00113785, 0.00064919)	(0.99927010, 0.99927010)
38	0.01390879	(1.00008032, 1.00016155)	$6.53263E{-}09$	($5.46242E{-}05$, $5.37367E{-}05$)	(0.99930798, 0.99930798)
39	0.17036167	(1.00000573, 1.00000981)	$3.01626E{-}10$	(−0.00020086, 0.00018080)	(0.99934299, 0.99934299)
40	0.00110133	(1.00000135, 1.00000276)	$2.25241E{-}12$	(0.00066707, −0.00032792)	(0.99937541, 0.99937541)

(continued)

k-th	Step length	x^T	$f(x)$	$\left(g_k^q\right)^T$	q_k^T
41	0.00105332	(1.00000137, 1.00000274)	1.868E−12	(−2.38082E−05, 1.3255E−05)	(0.99940549, 0.99940549)
42	0.00863207	(1.00000136, 1.00000273)	1.86043E−12	(1.00478E−06, 8.63942E−07)	(0.99943344, 0.99943344)
43	0.26476293	(1.00000015, 1.00000026)	1.55091E−13	(−2.47763E−06, 2.59736E−06)	(0.99945947, 0.99945947)
44	0.00112455	(1.00000002, 1.00000005)	5.69295E−16	(1.48783E−05, −7.29243E−06)	(0.99948375, 0.99948375)

Example 4.2. *Consider the Beale function* $f : \mathbb{R}^2 \to \mathbb{R}$ *be defined as*

$$f(x_1, x_2) = \left[1.5 - x_1 (1 - x_2)\right]^2 + \left[2.25 - x_1 \left(1 - x_2^2\right)\right]^2 + \left[2.625 - x_1 \left(1 - x_2^3\right)\right]^2.$$

We optimize the given function using both the MCD (Algorithm 1) and q-MCD (Algorithm 2) methods. The function has a unique global minimum at $(3, 0.5)^T$, and we initialize all three methods (CD, MCD, and q-MCD) from the starting point $(3, 1)^T$. We have shown the numerical results of CD in Table 4, numerical results of MCD in Table 5 and numerical results of q-MCD in Table 6, respectively. Under identical initial conditions and convergence thresholds, our analysis reveals that:

(a). The proposed MCD method achieves the approximate global minimum in fewer iterations compared to both CD and q-MCD for Example 4.2.

(b). While CD and q-MCD demonstrate nearly identical convergence rates, the q-MCD method offers significantly broader applicability than both CD and MCD.

Table 4: Detailed Computation for Beale Function on CD CGM

k-th	Step length	x^T	$f(x)$	g_k^T
1	0.00686801	(3, 0.42823838)	0.10571383	(0, 83.25)
2	0.02267706	(2.98119025, 0.48896238)	0.00095108	(0.82946170, −2.77878951)
3	0.01667700	(2.97995444, 0.49409517)	8.36296E−05	(0.06563739, −0.28044875)
4	0.01756258	(2.97987055, 0.49489278)	6.64502E−05	(0.00346781, −0.03997943)
5	0.11424006	(2.98052276, 0.49539973)	6.36731E−05	(−0.00582642, −0.00332252)
6	0.50477631	(2.99944380, 0.50029594)	4.39114E−06	(−0.00947982, 0.01206724)
7	0.01994262	(3.00088639, 0.50024348)	1.38671E−07	(−0.00513879, 0.02001960)
8	0.01974344	(3.00089021, 0.50022166)	1.26743E−07	(1.14388E−05, 0.00109737)
9	0.53021806	(3.00075192, 0.50016005)	1.06441E−07	(0.00027312, 4.61588E−05)
10	0.11733807	(2.99997999, 0.49998718)	1.48674E−09	(0.00054184, −0.00121628)
11	0.01903899	(2.99996864, 0.49999189)	1.59938E−10	(8.3426E−05, −0.00036226)
12	0.02826561	(2.99996878, 0.49999234)	1.56133E−10	(−6.28154E−06, −1.51586E−05)
13	2	(2.99999553, 0.50000042)	5.73214E−11	(−1.08789E−05, 3.56838E−06)
14	0.02034745	(3.00000703, 0.50000235)	1.63017E−11	(−1.8952E−05, 7.06645E−05)
15	0.01122611	(3.00000834, 0.50000224)	1.18428E−11	(−4.644E−06, 2.78088E−05)
16	0.01152950	(3.00000844, 0.50000214)	1.1457E−11	(6.46429E−07, 8.15445E−06)
17	0.01940471	(3.00000842, 0.50000207)	1.13649E−11	(2.1559E−06, 2.20281E−06)
18	0.07338426	(3.00000816, 0.50000189)	1.10433E−11	(2.87346E−06, −7.12516E−07)
19	0.10023553	(3.00000549, 0.50000097)	8.35483E−12	(4.11334E−06, −6.06016E−06)

(continued)

k-th	Step length	x^T	$f(x)$	g_k^T
20	0.03215979	(2.99999950, 0.49999955)	$2.49979E{-12}$	$(6.23024E{-06}, -1.80341E{-05})$
21	0.01270757	(2.99999789, 0.49999937)	$9.76801E{-13}$	$(3.57342E{-06}, -1.50616E{-05})$
22	0.01109265	(2.99999774, 0.49999941)	$8.39712E{-13}$	$(5.49949E{-07}, -4.94095E{-06})$
23	0.01248351	(2.99999773, 0.49999943)	$8.26063E{-13}$	$(-3.68327E{-07}, -1.43216E{-06})$

Table 5: Detailed Computation for Beale Function on MCD CGM

k-th	Step length	x^T	$f(x)$	g_k^T
1	0.00686801	(3, 0.42823838)	0.10571383	(0, 83.25)
2	0.02267706	(2.98119025, 0.48896238)	0.00095108	(0.82946170, −2.77878951)
3	0.01667700	(2.97995444, 0.49409517)	$8.36296E{-05}$	(0.06563739, −0.28044875)
4	0.01756258	(2.97987055, 0.49489278)	$6.64502E{-05}$	(0.00346781, −0.03997943)
5	0.11424006	(2.98052276, 0.49539973)	$6.36731E{-05}$	(−0.00582642, −0.00332252)
6	0.50477631	(2.99944380, 0.50029594)	$4.39114E{-06}$	(−0.00947982, 0.01206724)
7	0.01994262	(3.00088639, 0.50024348)	$1.38671E{-07}$	(−0.00513879, 0.02001960)
8	0.01974344	(3.00089021, 0.50022166)	$1.26743E{-07}$	$(1.14388E{-05}, 0.00109737)$
9	0.53021806	(3.00075192, 0.50016005)	$1.06441E{-07}$	$(0.00027312, 4.61588E{-05})$
10	0.11733807	(2.99997999, 0.49998718)	$1.48674E{-09}$	(0.00054184, −0.00121628)
11	0.01903899	(2.99996864, 0.49999189)	$1.59938E{-10}$	$(8.3426E{-05}, -0.00036226)$
12	0.02826561	(2.99996878, 0.49999234)	$1.56133E{-10}$	$(-6.28154E{-06}, -1.51586E{-05})$
13	2	(2.99999553, 0.50000042)	$5.73214E{-11}$	$(-1.08789E{-05}, 3.56838E{-06})$
14	0.02274955	(2.99999905, 0.49999975)	$1.51017E{-13}$	$(-1.8952E{-05}, 7.06645E{-05})$

Table 6: Detailed Computation for Beale Function on q-MCD CGM

k-th	Step length	x^T	$f(x)$	$\left(g_k^q\right)^T$	q_k^T
1	0.00844305	(3, 0.34707805)	0.42376924	(0, 77.3325)	(0.1, 0.1)
2	0.02541225	(2.95525052, 0.46000015)	0.01738596	(1.76094129, −4.77904614)	(0.975, 0.975)
3	0.02446862	(2.94626315, 0.48491435)	0.00053657	(0.30642127, −0.86458304)	(0.89166667, 0.89166667)
4	0.02188723	(2.94628702, 0.48639899)	0.00049046	(−0.00259224, −0.06366745)	(0.94427083, 0.94427083)
5	0.12010258	(2.94858196, 0.48805253)	0.00047321	(−0.01900379, −0.00726961)	(0.96222917, 0.96222917)
6	0.36886037	(2.99924104, 0.50135184)	$5.48774E{-05}$	(−0.02990553, 0.04135257)	(0.97327141, 0.97327141)
7	0.01994859	(3.00507400, 0.50139162)	$4.52781E{-06}$	(−0.01784379, 0.07008373)	(0.98013732, 0.98013732)
8	0.01823654	(3.00511674, 0.50127435)	$4.16842E{-06}$	$(5.01141E{-05}, 0.00644679)$	(0.98468535, 0.98468535)
9	0.15219743	(3.00490653, 0.50113231)	$3.98806E{-06}$	(0.00152944, 0.00052635)	(0.98784339, 0.98784339)
10	0.35670156	(3.00028850, 0.49992234)	$5.26701E{-07}$	(0.00249699, −0.00366928)	(0.99012157, 0.99012157)

(continued)

k-th	Step length	x^T	$f(x)$	g_k^T	q_k^T
11	0.01974421	(2.99960950, 0.49988883)	2.91466E−08	(0.00179899, −0.00684182)	(0.99181718, 0.99181718)
12	0.01785774	(2.99960342, 0.49990041)	2.52204E−08	(3.8693E−05, −0.00066311)	(0.99311238, 0.99311238)
13	0.09475379	(2.99961291, 0.49990853)	2.44674E−08	(−0.00011288, −6.1565E−05)	(0.99412360, 0.99412360)
14	0.52377779	(2.99993513, 0.49999841)	5.52074E−09	(−0.00017583, 0.00020454)	(0.99492794, 0.99492794)
15	0.02230830	(3.00002927, 0.50000864)	1.81257E−10	(−0.00018658, 0.00066651)	(0.99557810, 0.99557810)
16	0.01785316	(3.00003003, 0.50000757)	1.44685E−10	(−6.4546E−06, 6.38956E−05)	(0.99611102, 0.99611102)
17	0.06884565	(3.00002954, 0.50000706)	1.41252E−10	(8.19062E−06, 5.91524E−06)	(0.99655325, 0.99655325)
18	1	(2.99999619, 0.49999737)	6.78214E−11	(1.24725E−05, −1.19826E−05)	(0.99692422, 0.99692422)
19	0.02069879	(2.99999458, 0.49999862)	4.73331E−12	(1.80573E−05, −7.76419E−05)	(0.99723844, 0.99723844)
20	0.04490343	(2.99999464, 0.49999870)	4.62524E−12	(−1.29613E−06, −1.79002E−06)	(0.99750690, 0.99750690)
21	2	(3.00000189, 0.50000058)	8.64669E−13	(−2.06545E−06, 1.38282E−06)	(0.99773808, 0.99773808)
22	0.02004485	(3.00000193, 0.50000048)	5.96324E−13	(−6.9181E−07, 5.22512E−06)	(0.99793856, 0.99793856)

Table 7: Comparative Analysis of the CD, MCD and q-MCD CGMs

CGMs		**Starting Point**	**CD**				**MCD**				**q-MCD**			
Name	n		Nit	NG	NF	TR	Nit	NG	NF	TR	Nit	NG	NF	TR
Bard	3	(1,1,1)	158	178	345	0.07	80	95	174	0.213	123	140	270	0.074
Bard	4	(2,2,2,2)	79	101	190	0.048	97	160	243	0.057	77	99	183	0.06
Bard	5	(1,2,3,4,5)	205	240	456	0.091	192	198	393	0.076	55	61	119	0.047
Bard	5	(3,3,3,3,3)	79	99	185	0.053	78	115	191	0.052	F	F	F	F
Bard	5	(4,4,4,4,4)	F	F	F	F	78	106	171	0.049	F	F	F	F
Gulf	3	(5,2.5,0.15)	1	2	5	0.019	1	2	5	0.016	1	2	5	0.027
Gulf	3	(2,1,0.5)	142	208	378	0.062	218	634	771	0.131	1	6	9	0.021
Gulf	3	(4,2,1)	321	980	1621	0.198	280	832	1011	0.175	2	18	63	0.028
Gulf	3	(6,3,1)	672	751	2079	0.204	401	1185	1416	0.22	2	9	55	0.029
Gulf	3	(8,4,1)	95	159	372	0.058	474	1285	1613	0.217	2	7	51	0.031
Sing	4	(3,−1,0,1)	266	341	640	0.06	235	367	618	0.067	109	207	336	0.049
Sing	4	(0.5,0.5,0.5,0.5)	244	340	625	0.055	103	153	260	0.038	118	177	311	0.049
Sing	4	(−1,−1,−1,−1)	F	F	F	F	108	179	305	0.038	157	260	431	0.06
Sing	4	(1,1,1,1)	F	F	F	F	108	179	305	0.039	157	260	431	0.058
Sing	4	(1,2,3,4)	F	F	F	F	142	238	404	0.045	120	230	383	0.046
Kowosb	4	(0.25,0.39,0.415,0.39)	F	F	F	F	387	756	946	0.144	0	2	3	0.019
Kowosb	4	(0,0,0,0)	424	481	907	0.111	136	277	306	0.059	1	3	8	0.024
Kowosb	4	(1,1,1,1)	398	481	907	0.117	503	1105	1351	0.203	F	F	F	F
Kowosb	4	(1.2,1.2,1.2,1.2)	283	575	975	0.115	760	1838	2231	0.29	F	F	F	F
Kowosb	4	(1.3,1.3,1.3,1.3)	342	511	1072	0.111	769	1744	2155	0.281	F	F	F	F
Bd	4	(1,1,1,1)	31	325	500	0.056	65	416	650	0.089	32	587	892	0.05

(continued)

CGMs		Starting Point	CD				MCD				q-MCD			
Name	n		*Nit*	*NG*	*NF*	*TR*	*Nit*	*NG*	*NF*	*TR*	*Nit*	*NG*	*NF*	*TR*
Bd	4	(4,4,4,4)	76	560	869	0.051	41	526	805	0.079	F	F	F	F
Bd	4	(25,5,−5,−1)	73	623	961	0.049	82	919	1406	0.189	14	340	514	0.025
Bd	4	(50,10,−10,−2)	38	462	707	0.034	69	559	866	0.053	30	683	1037	0.066
Bd	4	(100,100,100,100)	79	767	1182	0.063	92	1199	1830	0.093	43	877	1334	0.067
Watson	2	defined by n	14	40	66	0.041	14	40	66	0.036	F	F	F	F
Watson	3	defined by n	30	58	100	0.048	27	56	96	0.049	F	F	F	F
Watson	4	defined by n	133	271	467	0.151	55	102	178	0.076	42	97	165	0.018
Watson	5	defined by n	136	215	388	0.146	91	147	264	0.107	944	1003	1972	0.619
Rosex	2	defined by n	127	835	1320	0.097	51	266	416	0.061	89	283	466	0.058
Rosex	4	defined by n	127	835	1320	0.103	51	266	416	0.05	89	283	466	0.058
Rosex	6	defined by n	219	923	1497	0.134	51	266	416	0.053	72	273	442	0.062
Rosex	8	defined by n	128	836	1322	0.106	51	266	416	0.056	89	283	466	0.067
Rosex	10	defined by n	102	791	1243	0.111	54	269	421	0.066	73	266	432	0.063
Singx	4	defined by n	266	341	640	0.071	235	367	618	0.084	127	215	365	0.065
Singx	8	defined by n	270	345	648	0.076	128	228	384	0.055	248	362	620	0.092
Singx	12	defined by n	304	379	716	0.081	188	289	507	0.072	232	361	596	0.086
Singx	16	defined by n	301	376	710	0.091	91	183	308	0.057	186	275	474	0.082
Singx	20	defined by n	306	381	720	0.099	212	345	576	0.079	169	279	454	0.088
Pen2	2	defined by n	14	21	35	0.032	14	35	58	0.046	F	F	F	F
Pen2	4	defined by n	272	814	1330	0.149	18	30	52	0.032	27	39	81	0.037
Pen2	5	defined by n	166	581	935	0.129	667	1315	1704	0.229	32	49	99	0.037
Pen2	10	defined by n	428	1479	2437	0.226	385	760	1042	0.166	41	86	165	0.047
Vardim	2	defined by n	4	26	41	0.026	4	26	41	0.081	14	36	61	0.029
Vardim	3	defined by n	5	48	74	0.028	5	48	74	0.027	11	58	91	0.032
Vardim	4	defined by n	7	79	121	0.035	7	79	121	0.031	15	78	123	0.034
Vardim	5	defined by n	7	93	142	0.033	7	93	142	0.034	17	96	152	0.037
Vardim	10	defined by n	9	167	254	0.045	9	167	254	0.042	104	1056	1602	0.167
Trig	2	defined by n	117	152	288	0.054	21	46	64	0.04	F	F	F	F
Trig	3	defined by n	192	284	512	0.068	29	41	50	0.028	F	F	F	F
Trig	4	defined by n	74	76	145	0.034	25	38	42	0.031	F	F	F	F
Trig	5	defined by n	414	656	1171	0.135	52	88	100	0.035	F	F	F	F
Trig	10	defined by n	78	83	158	0.044	55	82	94	0.041	F	F	F	F
Bv	2	defined by n	9	11	21	0.024	9	11	21	0.039	11	13	25	0.032
Bv	3	defined by n	33	35	68	0.029	33	35	68	0.029	25	27	52	0.03
Bv	4	defined by n	56	59	115	0.035	56	59	115	0.044	74	77	151	0.042
Bv	5	defined by n	401	404	804	0.111	34	39	70	0.035	31	34	64	0.037
Bv	10	defined by n	749	751	1498	0.211	131	139	257	0.053	169	171	339	0.071
Ie	2	defined by n	6	8	15	0.025	6	8	15	0.025	21	23	45	0.039
Ie	3	defined by n	6	8	15	0.025	6	8	15	0.027	4	6	11	0.027
Ie	4	defined by n	6	8	15	0.026	6	8	15	0.026	4	6	11	0.029

(continued)

CGMs		Starting Point	CD				MCD				q-MCD			
Name	n		*Nit*	*NG*	*NF*	*TR*	*Nit*	*NG*	*NF*	*TR*	*Nit*	*NG*	*NF*	*TR*
Ie	5	defined by n	6	8	15	0.027	6	8	15	0.028	4	6	11	0.028
Ie	10	defined by n	6	8	15	0.028	6	8	15	0.027	2	4	7	0.023
Trid	2	defined by n	18	39	67	0.029	18	39	67	0.025	15	39	65	0.03
Trid	3	defined by n	15	28	52	0.028	15	28	52	0.027	22	41	73	0.033
Trid	4	defined by n	23	37	67	0.031	17	30	54	0.028	21	33	62	0.032
Trid	5	defined by n	22	44	78	0.03	24	45	80	0.029	22	36	67	0.033
Band	2	defined by n	7	31	50	0.026	7	31	50	0.039	7	34	53	0.036
Band	3	defined by n	6	26	42	0.026	6	26	42	0.026	12	45	71	0.032
Band	4	defined by n	12	44	72	0.028	12	44	72	0.029	18	87	138	0.042
Band	5	defined by n	16	61	99	0.032	16	61	99	0.03	20	100	158	0.045
Lin	n=2 m=4	defined by n, m	1	2	3	0.017	1	2	3	0.015	1	2	3	0.021
Lin	n=4 m=10	defined by n, m	1	2	3	0.015	1	2	3	0.014	1	2	3	0.016
Lin1	n=2 m=4	defined by n, m	1	2	3	0.015	1	2	3	0.014	1	2	3	0.019
Lin1	n=4 m=10	defined by n, m	1	2	3	0.014	1	2	3	0.014	1	2	3	0.016

Following Dolan and Moré [10] established benchmarking approach, we employ performance profiles for comparative analysis of the proposed methods. Let P denote the entire test set n_p, its solution set as S. Let $t_{p,s}$ be the computing time required to solve problem p by solver s. For example $t_{p,s}$ can be the CPU time of $s \in S$ solves the problem $p \in P$. Let us define the coefficient of performance profile

$$r_{p,s} = \frac{t_{p,s}}{\min\left\{t_{p,s} : s \in S\right\}}. \tag{63}$$

The performance of each method s is defined as

$$\rho_s(\tau) = \frac{1}{n_p}\text{size}\left\{p \in P : r_{p,s} \leq \tau\right\}, \tag{64}$$

where size$\left\{p \in P : r_{p,s} \leq \tau\right\}$ is the number of elements of set $\left\{p \in P : r_{p,s} \leq \tau\right\}$. The performance profile $\rho_s(\tau)$ is the probability of corresponding coefficient $r_{p,s}$ less than or equal to the factor $\tau \in \mathbb{R}$, and it is easy to know that the function $\rho_s(\tau)$ is the distribution function of the performance profile coefficient. In general, a solver with higher values of $\rho_s(\tau)$, which lies closer to the upper left corner of the figure, is more preferable.

Figs. 1, 2, 3 and 4 show performance profiles based on the numerical results summarized in Table 7. From four figures, we conclude that the MCD performs better than the CD and q-MCD. In Table 7, with the give initial points, MCD solves all the problems but CD and q-MCD are overwhelmed when dealing with some problems such as test problem Bard et al. The complete numerical results in Table 7 validate that both proposed methods significantly improve upon the classical CD approach given by Fletcher in Ref. [14].

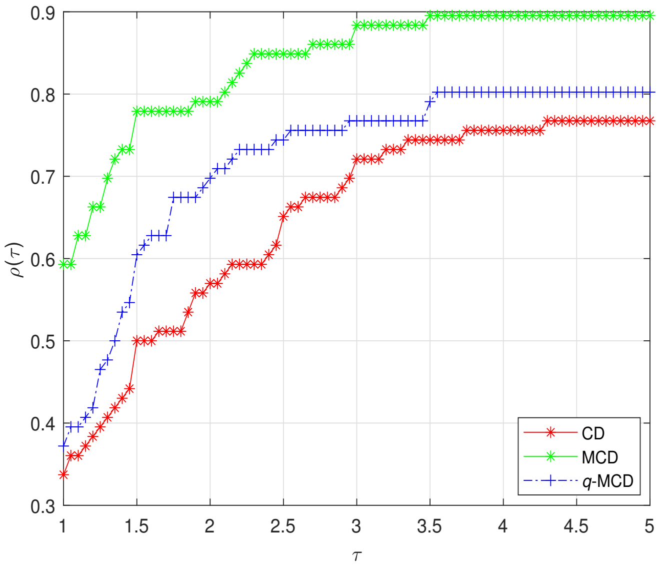

Figure 1: Number of Iteration

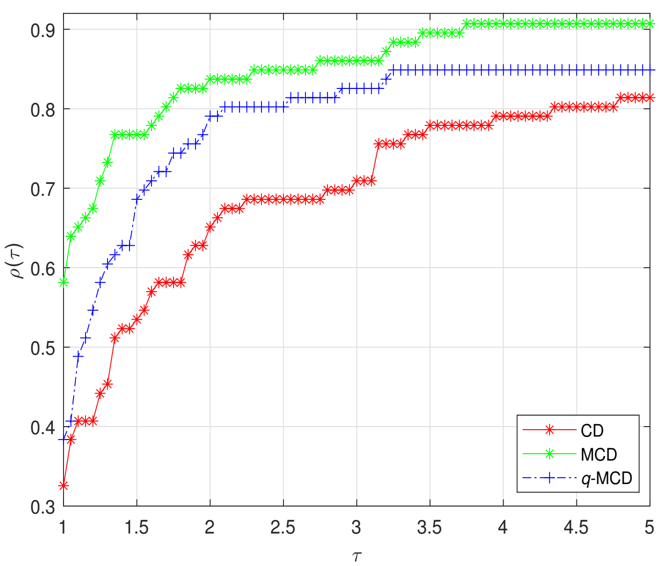

Figure 2: Number of Gradient Evaluations

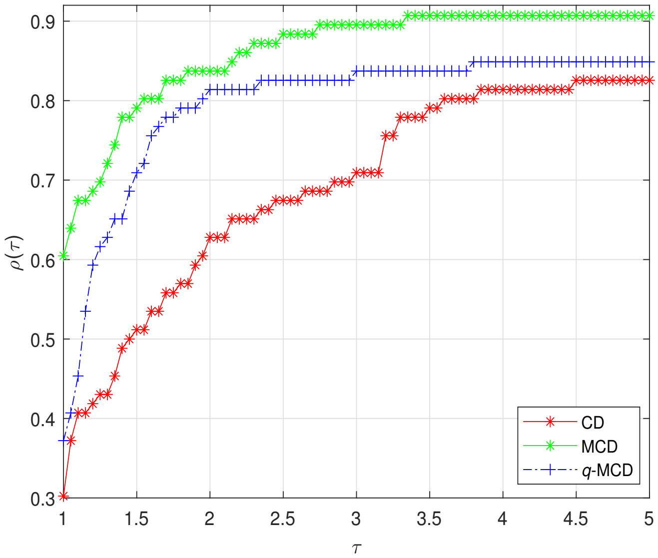

Figure 3: Number of Function Evaluations

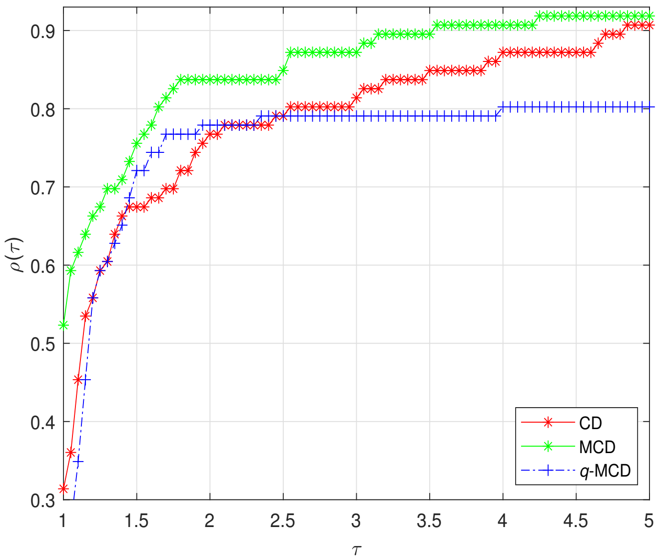

Figure 4: Time of Running

5. Conclusions

In this paper, we develop a modified Conjugate-Descent (MCD) method and its corresponding q-variant (q-MCD) by incorporating q-gradient concepts. Both proposed methods satisfy the sufficient descent condition regardless of whether exact or inexact line searches are employed, while preserving the desirable properties of the original CD method. Under standard Wolfe line search conditions and appropriate assumptions, we have proven the global convergence properties of these methods. Extensive numerical experiments demonstrate the superior computational efficiency of both MCD and q-MCD. We believe that these proposed methods will significantly inspire and facilitate future research in this field.

Acknowledgements

The authors would like to thank the reviewers and the editor Prof. Predrag Stanimirović for their valuable comments and suggestions.

References

[1] A. B. Abubakar, P. Kumam, M. Malik, A. H. Ibrahim, A hybrid conjugate gradient based approach for solving unconstrained optimization and motion control problems, *Math. Comput. Simulation*, **201** (2022), 640–657.
[2] A. B. Abubakar, M. Malik, P. Kumam, H. Mohammad, M. Sun, A. H. Ibrahim, A. I. Kiri, A Liu–Storey-type conjugate gradient method for unconstrained minimization problem with application in motion control, *Journal of King Saud University-Science*, **34** (2022), Article ID 101923, 11 pages.
[3] 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.
[4] A. Alhawarat, Z. Salleh, M. Mamat, M. Rivaie, An efficient modified Polak–Ribière–Polyak conjugate gradient method with global convergence properties, *Optim. Methods Softw.*, **32** (6) (2017), 1299–1312.
[5] A. M. Awwal, P. Kumam, A. B. Abubakar, A modified conjugate gradient method for monotone nonlinear equations with convex constraints, *Appl. Numer. Math.*, **145** (2019), 507–520.
[6] S. K. Chakraborty, G. Panda, Newton like line search method using q-calculus, *In: Giri, D., Mohapatra, R., Begehr, H., Obaidat, M. (eds) Mathematics and Computing, ICMC 2017,* Communications in Computer and Information Science, Vol. 655, Springer, Singapore, https://doi.org/10.1007/978-981-10-4642-1.
[7] Y. H. Dai, Y. X. Yuan, A nonlinear conjugate gradient method with a strong global convergence property, *SIAM J. Optim.*, **10** (1999), 177–182.
[8] J. Deepho, A. B. Abubakar, M. Malik, I. K. Argyros, Solving unconstrained optimization problems via hybrid CD-DY conjugate gradient methods with applications, *J. Comput. Appl. Math.*, **405** (2022), Article ID 113823, 16 pages.
[9] Y. Ding, Global convergence of the original Liu–Storey conjugate gradient method, *ScienceAsia*, **42** (2016), 146–149.
[10] E. D. Dolan, J. J. Moré, Benchmarking optimization software with performance profiles, *Math. Program.*, **91** (2002), 201–213.
[11] T. Ernst, A method for q-calculus, *J. Nonlinear Math. Phys.*, **10** (4) (2003), 487–525.
[12] M. L. Fang, M. Wang, M. Sun, R. Chen, A modified hybrid conjugate gradient method for unconstrained optimization, *J. Math.*, **2021** (2021), Article ID 5597863, 9 pages.
[13] P. Faramarzi, K. Amini, A modified spectral conjugate gradient method with global convergence, *J. Optim. Theory Appl.*, **182** (2) (2019), 667–690.
[14] R. Fletcher, Practical methods of optimization Vol. 1: Unconstrained Optimization, John Wiley and Sons, New York, 1987.
[15] R. Fletcher, C. M. Reeves, Function minimization by conjugate gradients, *Comput. J.*, **7** (1964), 149–154.
[16] É. J. C. Gouvêa, R. G. Regis, A. C. Soterroni, M. C. Scarabello, F. M. Ramos, Global optimization using q-gradients. *European J. Oper. Res.*, **251** (2016), 727–738.
[17] W. W. Hager, H. Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search, *SIAM J. Optim.*, **16** (2005), 170–192.
[18] X. Han, J. Zhang, J. Chen, A new hybrid conjugate gradient algorithm for unconstrained optimization, *Bull. Iranian Math. Soc.*, **43** (2017), 2067–2084.
[19] M. R. Hestenes, E. Stiefel, Methods of conjugate gradients for solving linear systems, *J. Res. Natl. Bur. Stand.*, **49** (1952), 409–436.
[20] A. H. Ibrahim, P. Kumam, A. Kamandi, A. B. Abubakar, An efficient hybrid conjugate gradient method for unconstrained optimization, *Optim. Methods Softw.*, **37** (4) (2022), 1370–1383.
[21] A. H. Ibrahim, P. Kumam, A. B. Abubakar, U. B. Yusuf, S. E. Yimer, K. O. Aremu, An efficient gradient-free projection algorithm for constrained nonlinear equations and image restoration, *AIMS Math.*, **6** (1) (2021), 235–260.
[22] F. H. Jackson, On q-functions and a certain difference operator, *Trans. Roy. Soc. Edin.*, **46** (1909), 253–281.
[23] X. Z. Jiang, L. Han, J. B. Jian, A globally convergent mixed conjugate gradient method with Wolfe line search, *Math. Numer. Sin.*, **34** (1) (2012), 103–112.
[24] X. Z. Jiang, J. B. Jian, A sufficient descent Dai–Yuan type nonlinear conjugate gradient method for unconstrained optimization problems, *Nonlinear Dyn.*, **72** (2013), 101–112.
[25] V. Kac, P. Cheung, Quantum calculus, Springer-Verlag, New York, 2002.
[26] K. K. Lai, S. K. Mishra, G. Panda, M. A. T. Ansary, B. Ram, On q-steepest descent method for unconstrained multiobjective optimization problems, *AIMS Math.*, **5** (6) (2020), 5521–5540.
[27] K. K. Lai, S. K. Mishra, G. Panda, S. K. Chakraborty, M. E. Samei, B. Ram, A limited memory q-BFGS algorithm for unconstrained optimization problems, *J. Appl. Math. Comput.*, **66** (2021), 183–202.
[28] K. K. Lai, S. K. Mishra, B. Ram, On q-quasi-Newton's method for unconstrained multiobjective optimization problems, *Mathematics*, **8** (4) (2020), Article ID 616, 14 pages.
[29] K. K. Lai, S. K. Mishra, B. Ram, A q-conjugate gradient algorithm for unconstrained optimization problems, *Pac. J. Optim.*, **17** (2021), 57–76.

[30] M. Li, H. Y. Feng, A sufficient descent LS conjugate gradient method for unconstrained optimization problems, *Appl. Math. Comput.*, **218** (2011), 1577–1586.
[31] T. W. Liu, Two class of conjugate gradient methods with a variable parameter for unconstrained optimization, *J. Numer. Methods Comput. Appl.*, **2** (2001), 106–112.
[32] Y. Liu, C. Storey, Efficient generalized conjugate gradient algorithms, part 1: theory, *J. Optim. Theory Appl.*, **69** (1991), 129–137.
[33] S. K. Mishra, S. K. Chakraborty, M. E. Samei, B. Ram, A *q*-Polak–Ribière–Polyak conjugate gradient algorithm for unconstrained optimization problems, *J. Inequal. Appl.*, **2021** (2021), Article ID 25, 29 pages.
[34] S. K. Mishra, G. Panda, M. A. T. Ansary, B. Ram, On *q*-Newton's method for unconstrained multiobjective optimization problems, *J. Appl. Math. Comput.*, **63** (2020), 391–410.
[35] S. K. Mishra, G. Panda, S. K. Chakraborty, M. E. Samei, B. Ram, On *q*-BFGS algorithm for unconstrained optimization problems, *Adv. Difference Equ.*, **2020** (2020), Article ID 638, 24 pages.
[36] S. K. Mishra, M. E. Samei, S. K. Chakraborty, B. Ram, On *q*-variant of Dai–Yuan conjugate gradient algorithm for unconstrained optimization problems, *Nonlinear Dyn.*, **104** (2021), 2471–2496.
[37] J. J. Moré, B. S. Garbow, K. E. Hillstrom, Testing unconstrained optimization software, *ACM Trans. Math. Software*, **7** (1981), 17–41.
[38] B. Polak, G. Ribière, Note sur la convergence de méthodes de directions conjuguées, *Rev. Fr. Inform. Rech. Oper.*, **3** (1) (1969), 35–43.
[39] B. T. Polyak, The conjugate gradient method in extremal problems, *USSR Comput. Math. Math. Phys.*, **9** (1969), 94–112.
[40] M. J. D. Powell, Nonconvex minimization calculations and the conjugate gradient method, In: Numerical Analysis, pp. 122–141, Springer, Berlin, 1984.
[41] P. M. Rajković, S. D. Marinković, M. S. Stanković, On *q*-Newton–Kantorovich method for solving systems of equations, *Appl. Math. Comput.*, **168** (2) (2005), 1432–1448.
[42] P. M. Rajković, M. S. Stanković, S. D. Marinković, Mean value theorems in *q*-calculus, *Matematički Vesnik*, **54** (2002), 171–178.
[43] J. Sabi'u, A. Shah, An efficient three-term conjugate gradient-type algorithm for monotone nonlinear equations, *RAIRO Oper. Res.*, **55** (2021), S1113–S1127.
[44] G. Sana, P. O. Mohammed, D. Y. Shin, M. A. Noor, M. S. Oudat, On iterative methods for solving nonlinear equations in quantum calculus, *Fractal Fract.*, **5** (2021), Article ID 60, 17 pages.
[45] A. C. Soterroni, R. L. Galski, F. M. Ramos, The *q*-gradient vector for unconstrained continuous optimization problems, B. Hu et al. (eds.), *Operations Research Proceedings 2010*, Springer-Verlag Berlin Heidelberg, 2011, 365–370.
[46] D. A. Tarzanagh, P. Nazari, M. R. Peyghami, A nonmonotone PRP conjugate gradient method for solving square and underdetermined systems of equations, *Comput. Math. Appl.*, **73** (2017), 339–354.
[47] M. Y. Waziri, K. Ahmed, Two descent Dai–Yuan conjugate gradient methods for systems of monotone nonlinear equations, *J. Sci. Comput.*, **90** (2022), Article ID 36, 53 pages.
[48] M. Y. Waziri, K. Ahmed, A. S. Halilu, A modified Dai–Kou-type method with applications to signal reconstruction and blurred image restoration, *Comput. Appl. Math.*, **41** (2022), Article ID 232, 33 pages.
[49] M. Y. Waziri, K. Ahmed, A. S. Halilu, A. M. Awwal, Modified Dai–Yuan iterative scheme for nonlinear systems and its application, *Numer. Alg. Control Optim.*, **13** (1) (2023), 53–80.
[50] Y. H. Xiao, Z. X. Wei, Z. G. Wang, A limited memory BFGS-type method for large-scale unconstrained optimization, *Comput. Math. Appl.*, **56** (2008), 1001–1009.
[51] Y. Yu, X. C. Luo, Z. Y. Wu, Q. X. Zhang, Y. W. Qi, X. F. Pang, Estimation of the boundary condition of a 3D heat transfer equation using a modified hybrid conjugate gradient algorithm, *Appl. Math. Model.*, **102** (2022), 768–785.
[52] G. L. Yuan, T. T. Li, W. J. Hu, A conjugate gradient algorithm for large-scale nonlinear equations and image restoration problems, *Appl. Numer. Math.*, **147** (2020), 129–141.
[53] G. L. Yuan, J. Y. Lu, Z. Wang, The PRP conjugate gradient algorithm with a modified WWP line search and its application in the image restoration problems, *Appl. Numer. Math.*, **152** (2020), 1–11.
[54] Z. B. Zhu, D. D. Zhang, S. Wang, Two modified DY conjugate gradient methods for unconstrained optimization problems, *Appl. Math. Comput.*, **373** (2020), Article ID 125004, 10 pages.