Optimization Methods & Software, 2019
Vol. 34, No. 3, 650–664, https://doi.org/10.1080/10556788.2017.1414813

A new conjugate gradient algorithm with cubic Barzilai–Borwein stepsize for unconstrained optimization

M. Momeni[a,b] and M.R. Peyghami[a,b*]

[a]*Faculty of Mathematics, K.N. Toosi Univ. of Tech., P.O. Box 16315-1618, Tehran, Iran;* [b]*Scientific Computations in OPtimization and Systems Engineering (SCOPE), K.N. Toosi Univ. of Tech., Tehran, Iran*

(*Received 15 January 2017; accepted 19 November 2017*)

In this paper, a new conjugate gradient (CG) algorithm in Dai–Liao (DL) family is presented for solving unconstrained optimization problems. The proposed algorithm tries to adjust positive values for the so-called DL parameter by using quadratic and/or cubic models of the objective function. More precisely, the cubic regularization model of the objective function is properly employed when the non-positive curvature is detected. Besides, the CG parameter is introduced so that the generated CG directions are descent. Under some standard assumptions, we establish the convergence property of the new proposed algorithm. Numerical results on some test problems are reported. The results show that the new algorithm performs well and is competitive with CG_DESCENT method.

Keywords: Conjugate gradient methods; Dai–Liao family; Cubic regularization model; CG_DESCENT method

1. Introduction

In this paper, we deal with large-scale unconstrained optimization problems as follows:

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

where $f:\mathbb{R}^n\to\mathbb{R}$ is a smooth function and its gradient $g(x):=\nabla f(x)$ is available at any point $x\in\mathbb{R}^n$. Conjugate Gradient (CG) methods are a class of optimization methods for solving such problems that do not require any second-order information of the objective function. For given $x_0\in\mathbb{R}^n$, CG methods generate a sequence of points according to the following formula:

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

where α_k is a stepsize and the search direction d_k is generated by

$$d_k=\begin{cases}-g_k, & k=0,\\ -g_k+\beta_{k-1}d_{k-1}, & k\geq 1,\end{cases} \tag{3}$$

where β_k is a scaler which is known as conjugate gradient parameter. Different choices of β_k lead to various conjugate gradient methods. We refer the interested reader to [13] and the references

*Corresponding author. Email: peyghami@kntu.ac.ir

therein for more details. Among them, the one proposed by Dai and Liao [8] received more attentions by the researchers as it is obtained by considering both conjugacy condition and quasi-Newton equation along with Wolfe conditions [14] for the stepsize α_k, i.e.

$$\begin{aligned} &f(x_k + \alpha_k d_k) - f(x_k) \leq \rho \alpha_k {g_k}^{\mathrm{T}} d_k, \\ &g(x_k + \alpha_k d_k)^{\mathrm{T}} d_k \geq \sigma {g_k}^{\mathrm{T}} d_k, \end{aligned} \tag{4}$$

where $0 < \rho < \frac{1}{2} \leq \sigma < 1$ are constants. Indeed, they proposed the conjugate gradient parameter as follows:

$$\beta_k^{\mathrm{DL}}(t) = \frac{g_{k+1}^{\mathrm{T}}(y_k - t s_k)}{d_k^{\mathrm{T}} y_k}, \tag{5}$$

where $t \geq 0$ is a scaler and s_k and y_k are defined as follows:

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

Practical performance of this method with $t = 1$ and $t = 0.1$ has been reported in [8] to show its efficiency in practice. Besides, due to [1], the performance of $\beta_k^{\mathrm{DL}}(t)$ in the structure of nonlinear CG algorithms depends on the way of choosing t. Based on the memoryless quasi-Newton scheme of Perry [15] and Shanno [20], Hager and Zhang [12] suggested the following formula for the CG parameter:

$$\beta_k^{\mathrm{HZ}} = \frac{g_{k+1}^{\mathrm{T}} y_k}{d_k^{\mathrm{T}} y_k} - 2 \frac{y_k^{\mathrm{T}} y_k}{d_k^{\mathrm{T}} y_k} \frac{g_{k+1}^{\mathrm{T}} d_k}{d_k^{\mathrm{T}} y_k}. \tag{7}$$

This formula can be regarded as β_k^{DL} with $t = 2 y_k^{\mathrm{T}} y_k / s_k^{\mathrm{T}} y_k$. In other words, one can assume β_k^{HZ} as β_k^{DL} with $t = 2/\hat{\alpha}$, where $\hat{\alpha}$ is the classical Barzilai–Borwein (BB) stepsize [5]. It is worth mentioning that Hager and Zhang's CG algorithm is known as CG_DESCENT algorithm in the literature as it generates conjugate directions satisfying the sufficient descent property, namely

$${g_k}^{\mathrm{T}} d_k \leq \tfrac{-7}{8} \| g_k \|^2, \quad \forall k \geq 0. \tag{8}$$

Recently Xiao *et al.* [21] employed the so-called cyclic BB stepsize in the structure of Dai–Liao parameter and proposed a new CG algorithm competing the CG_DESCENT in practice.

Quadratic approximation of the objective function around the point x_k is traditionally utilized by optimization techniques in order to generate a direction towards optimality. However, an iteration based on this approach may not be a well-defined specially when a non-positive curvature is detected. In this case, there is no warranty for global convergence. A treat for this phenomena is to use some higher order techniques. The earliest attempts based on tensor methods were done by Schnabel and Frank [19] for nonlinear equations and Schnabel and Chow [18] for unconstrained optimization. These methods are based upon a fourth-order approximation of the objective function around the current point by using the Taylor expansion with a low-rank third- and fourth-order terms. Roughly speaking, the so-called cubic regularization model, proposed by Griewank [11], has been employed in the literature as a higher approximation to the objective function. Among them, the one that computes the series terms simply is more preferred. Recently Curtis and Guo [6] proposed a new cubic model that handles nonconvexity in an appropriate way. Here, we use their cubic model in the structure of our new approach.

In this paper, we propose a new conjugate gradient method in the Dai–Liao family. Our aim is to introduce a specific parameter $t > 0$ so that the generated directions are descent. The main contribution of this paper is to employ the cubic regularization model of the objective function, as proposed by Curtis and Guo [6], in order to introduce an appropriate positive t when the

non-positive curvature is detected. Our approach for choosing t is different from that employed in CG_DESCENT method in which t is not necessarily positive and only quadratic approximation model is used for the non-positive curvature case. Under some standard assumptions, the convergence property of the new method is established. Numerical results on some test problems show that our new approach outperforms CG_DESCENT algorithm in terms of number of iterations, function evaluations and gradient evaluations.

The rest of the paper is organized as follows: In Section 2, a brief exposition of the cubic regularization model is given and the structure of the new proposed algorithm is provided. In Section 3, we construct convergence property of the new algorithm under some standard assumptions. Section 4 deals with computational results on a set of 120 medium- and large-scale unconstrained optimization test problems from Andrei [2] and comparison with CG_DESCENT. We end up the paper by some concluding remarks in Section 5.

2. The new algorithm

To establish the new conjugate gradient algorithm, we need to briefly recall some concepts of the BB gradient method [5]. The BB gradient method for solving unconstrained optimization problems forms the iteration points as follows:

$$x_{k+1} = x_k - \alpha_k g_k, \tag{9}$$

where $g_k = \nabla f(x_k)$, and α_k is chosen appropriately. In this method, the Hessian approximation H_k at x_k is considered as a multiple of identity matrix so that $H_{k+1}s_k = y_k$ holds in the least-squares sense. More precisely, let

$$\min_{\bar{q}\in\mathbb{R}} \frac{1}{2} \parallel (\bar{q}I)s_k - y_k\parallel_2{}^2, \quad \min_{\hat{q}\in\mathbb{R}} \frac{1}{2} \parallel s_k - (\hat{q}^{-1}I)y_k\parallel_2{}^2.$$

Assuming $s_k^{\mathrm{T}}y_k > 0$, one can easily figure out that

$$\bar{q} = \frac{s_k^{\mathrm{T}}y_k}{s_k^{\mathrm{T}}s_k} \quad \text{and} \quad \hat{q} = \frac{y_k^{\mathrm{T}}y_k}{s_k^{\mathrm{T}}y_k},$$

are the solutions of these problems, respectively. Now, by minimizing the following quadratic model at x_k along $-g_k$, with $q_k = \bar{q}$ and $q_k = \hat{q}$,

$$\min_{\alpha>0} f(x_k - \alpha g_k) \approx f_k - \alpha \parallel g_k\parallel_2{}^2 + \frac{1}{2}\alpha^2 q_k \parallel g_k\parallel_2{}^2,$$

one can obtain the following two stepsizes, respectively:

$$\bar{\alpha}_k = \frac{s_k^{\mathrm{T}}s_k}{s_k^{\mathrm{T}}y_k} \quad \text{and} \quad \hat{\alpha}_k = \frac{s_k^{\mathrm{T}}y_k}{y_k^{\mathrm{T}}y_k}. \tag{10}$$

Under some assumptions on the problem, it is shown that the BB method with both stepsizes has global convergence property; see e.g. [5,9,16]. It is worth mentioning that some inefficiencies may arise in the case whenever the non-positive curvature is detected, i.e. $s_k^{\mathrm{T}}y_k \leq 0$.

Several attempts have been done to modify BB methods for the non-convex optimization problems [3,4,17,22]. Among them, Curtis and Guo [6] employed the following cubic model to

compute the stepsizes:

$$m_k(s) = f_k + {g_k}^{\mathrm{T}} s + \tfrac{1}{2} q_k \parallel s \parallel_2{}^2 + \tfrac{1}{6} c_k \parallel s \parallel_2{}^3 \approx f(x_k + s), \tag{11}$$

where q_k and c_k are scalars that are determined properly. First of all, letting $s = -\alpha g_k$, the following problem is solved to determine α_k:

$$\min_{\alpha > 0} \phi(\alpha) = f_k - \alpha \parallel g_k \parallel_2{}^2 + \tfrac{1}{2}\alpha^2 q_k \parallel g_k \parallel_2{}^2 + \tfrac{1}{6}\alpha^3 c_k \parallel g_k \parallel_2{}^3. \tag{12}$$

Now, we focus on the way of computing the values of q_k and c_k. According to Curtis and Guo [6], the matrix $\hat{q}_k I$ is more preferable approximation for the Hessian of f at x_k. Therefore, we use $q_k = \hat{q}_k$ in our setting.To compute c_k, we consider the following cases:

- If $s_k^{\mathrm{T}} y_k > 0$, then the quadratic model of f at x_k is convex, and therefore $c_k = 0$ is a reasonable choice.
- If $s_k^{\mathrm{T}} y_k < 0$, then we look for a strategy for choosing $c_k > 0$ so that (12) has a unique minimizer. A standard strategy is to choose c_k so that the least-squares error between the gradient of the model $m_k(s)$ at $-s_k$ and g_{k-1} becomes zero [6]. That is,

$$\nabla m_k(-s_k) - g_{k-1} = 0, \tag{13}$$

or

$$g_k - \hat{q}_k s_k - \tfrac{1}{2} c_k \parallel s_k \parallel_2 s_k - g_{k-1} = 0.$$

Thus,

$$y_k - \hat{q}_k s_k = \tfrac{1}{2} c_k \parallel s_k \parallel_2 s_k.$$

Multiplying both sides of this equality by s_k^{T} leads to

$${s_k}^{\mathrm{T}} y_k - \hat{q}_k \parallel s_k \parallel_2{}^2 = \tfrac{1}{2} c_k \parallel s_k \parallel_2{}^3, \tag{14}$$

which in turn implies that

$$c_k = \frac{2}{\parallel s_k \parallel_2} (\bar{q}_k - \hat{q}_k). \tag{15}$$

It has been shown in [6] that c_k, defined by (15), is positive.

Let us postpone the case $s_k^{\mathrm{T}} y_k = 0$ for a while, and talk about the way of calculating the stepsize in the above-mentioned cases. In case of having positive curvature, we set $c_k = 0$, and therefore $\phi(\alpha)$ is a convex quadratic function with respect to α. Following the same approach in BB method, the stepsize $\alpha_k = 1/\hat{q}_k$ solves (12) properly.

For the case of having negative curvature, we set $c_k > 0$, as given by (15), and α_k is chosen so that it minimizes $\phi_k(\alpha)$, i.e. $\phi_k'(\alpha_k) = 0$. This leads to the following equation:

$$\phi'(\alpha_k) = - \parallel g_k \parallel_2{}^2 + \alpha_k \hat{q}_k \parallel g_k \parallel_2{}^2 + \tfrac{1}{2}{\alpha_k}^2 c_k \parallel g_k \parallel_2{}^3 = 0,$$

or equivalently

$$\tfrac{1}{2} c_k \parallel g_k \parallel_2 {\alpha_k}^2 + \hat{q}_k \alpha_k - 1 = 0.$$

It can be easily verified that this quadratic equation has two real roots, and its positive root, i.e.

$$\alpha_k = \frac{-q_k + \sqrt{{q_k}^2 + 2c_k \parallel g_k \parallel_2}}{c_k \parallel g_k \parallel_2},$$

is the minimizer of $\phi_k(\alpha)$.

Now, we deal with the case $s_k^{\mathrm{T}} y_k = 0$. For this case, we have:

- If $y_k = 0$, then $g_k = g_{k-1}$ and f looks affine at x_k along $-g_k$. In this case, there is no curvature that limits the stepsize and therefore we set α_k to be a maximum predefined allowable stepsize, let say $\alpha_k = \Omega > 0$.
- If $y_k \neq 0$ and $s_k^{\mathrm{T}} y_k = 0$, then the gradient changes. Besides, there is no information about function's curvature. Thus, to be conservative, we set α_k to be a minimum predefined allowable stepsize, let say $\alpha_k = \omega > 0$.

Summarizing the above discussion, we have the following setting for the stepsize:

$$\alpha_k = \begin{cases} \dfrac{1}{\hat{q}_k}, & {s_k}^{\mathrm{T}} y_k > 0, \\ \dfrac{-\hat{q}_k + \sqrt{\hat{q}_k^2 + 2c_k \parallel g_k \parallel_2}}{c_k \parallel g_k \parallel_2}, & {s_k}^{\mathrm{T}} y_k < 0, \\ \Omega, & y_k = 0, \\ \omega, & o.w. \end{cases} \tag{16}$$

It is worth mentioning that to prevent very large and very small stepsizes, we employ the universal safeguard of projecting any computed stepsize onto the interval $[\omega, \Omega]$.

Now, let us consider the conjugate gradient method with β_k^{DL}. Hager and Zhang [12] proposed the CG parameter β_k^{HZ}, given by (7), that can be regarded as β_k^{DL} with $t = 2/\hat{\alpha}$, where $\hat{\alpha}$ is the stepsize computed by (10). Our methodology consists of different choices of t in the Dai–Liao family. Indeed, we follow the same strategy of Hager and Zhang and replace $\hat{\alpha}$ with those given by (16). Moreover, as usual, to have descent directions for general functions, we define β_k^{+} as below:

$${\beta_k}^{+} = \max\{\beta_k, 0\}. \tag{17}$$

The structure of our new proposed CG algorithm with cubic regularization model is outlined in Algorithm 1.

3. Convergence analysis

In this section, our aim is to investigate the convergence property of Algorithm 1 under some standard assumptions for strongly convex and general non-convex functions using Wolfe conditions. To do so, we first show that the search directions generated by Algorithm 1 fulfil the sufficient descent property.

LEMMA 3.1 *If* $d_k^{\mathrm{T}} g_k \neq 0$, *then*

$${g_k}^{\mathrm{T}} d_k \leq -7/8 \parallel g_k \parallel^2. \tag{18}$$

Proof The proof is divided into following cases:

- If $s_k^{\mathrm{T}} y_k > 0$, then our conjugate gradient parameter is the same as CG_DESCENT. Therefore, following the analogous proof line of Theorem 1.1 in [12], one can show that (18) holds.
- For the case $s_k^{\mathrm{T}} y_k \leq 0$, we proceed the proof by induction over k. For $k = 0$, then $d_k = -g_k$, and the inequality (18) holds. Assume that (18) holds for k (induction hypothesis). We prove it for $k + 1$.

Algorithm 1 A CG Method with Cubic Regularization

Given $x_0 \in \mathbb{R}^n$, $\epsilon > 0, \omega > 0$ and $\Omega > 0$, set $k = 0$ and $d_0 = -g_0$.
While $\|g_k\| > \epsilon$
Find α_k satisfying Wolfe conditions (4).
Set $x_{k+1} = x_k + \alpha_k d_k, s_k = x_{k+1} - x_k, y_k = g_{k+1} - g_k$.
If $y_k = 0$, then set $t = \frac{2}{\Omega}$.
Else if $s_k^{\mathrm{T}} y_k = 0$, then set $t = \frac{2}{\omega}$.
Else, set $\hat{q} = \frac{y_k^{\mathrm{T}} y_k}{s_k^{\mathrm{T}} y_k}$ and $\bar{q} = \frac{s_k^{\mathrm{T}} y_k}{s_k^{\mathrm{T}} s_k}$.
If $s_k^{\mathrm{T}} y_k > 0$, then set $t = 2\hat{q}$.
Else, set $c_k = \frac{2(\bar{q} - \hat{q})}{\|s_k\|_2}$ and

$$t = \frac{2c_k \parallel g_k\|_2}{-\hat{q}_k + \sqrt{\hat{q}_k{}^2 + 2c_k \parallel g_k\|_2}}.$$

Replace t by its projection onto the interval $[\omega, \Omega]$.
Compute $\beta_k = \frac{g_{k+1}^{\mathrm{T}}(y_k - ts_k)}{d_k^{\mathrm{T}} y_k}$ and set $\beta_k{}^+ = \max\{\beta_k, 0\}$.
Set $d_{k+1} = -g_{k+1} + \beta_k{}^+ d_k$ and $k = k + 1$.
End (While)

From $s_k^{\mathrm{T}} y_k \leq 0$, we conclude that $\alpha_k d_k^{\mathrm{T}}(g_{k+1} - g_k) \leq 0$, and therefore $d_k^{\mathrm{T}} g_{k+1} \leq d_k^{\mathrm{T}} g_k$. Thus, using (3), one can write

$$\begin{aligned} g_{k+1}{}^{\mathrm{T}} d_{k+1} &= - \parallel g_{k+1}\|^2 + \beta_k{}^+ g_{k+1}{}^{\mathrm{T}} d_k \\ &\leq - \parallel g_{k+1}\|^2 + \beta_k{}^+ g_k{}^{\mathrm{T}} d_k. \end{aligned}$$

Now, from induction hypothesis and the fact that $\beta_k{}^+ \geq 0$, we have

$$g_{k+1}{}^{\mathrm{T}} d_{k+1} \leq - \parallel g_{k+1}\|^2 \leq \frac{-7}{8} \parallel g_{k+1}\|^2.$$

Considering the above two cases, the proof is completed. ■

Now, we prove the global convergence property of Algorithm 1. For this purpose, we consider the following assumptions on the problem:

Assumption 1 The level set $\mathcal{L} = \{x | f(x) \leq f(x_0)\}$ is bounded.

Assumption 2 In some neighbourhood $\mathcal{N}$ of $\mathcal{L}$, f is continuously differentiable function, and its gradient is Lipschitz continuous, i.e. there exists a constant $L > 0$ so that

$$\parallel \nabla f(x) - \nabla f(y) \parallel \leq L \parallel x - y \parallel \quad \forall x, y \in \mathcal{N}. \tag{19}$$

For the convergence property of the new proposed algorithm, we need the following lemma which is well known as Zoutendijk condition. One can find its proof in [23].

LEMMA 3.2 *Suppose that $\{x_k\}$ is generated by a line search algorithm, with descent direction d_k, satisfying the Wolfe conditions (4). Moreover, assume that Assumptions 1 and 2 hold. Then, one has*

$$\sum_{k=0}^{\infty} \frac{(g_k^{\mathrm{T}} d_k)^2}{\parallel d_k\|^2} < \infty. \tag{20}$$

Using Lemma 3.2, one can easily conclude a lower bound for the stepsizes generated by Wolfe conditions.

LEMMA 3.3 (Theorem 3.2 in [14]) *Suppose that d_k is a descent direction, and Assumptions 1 and 2 hold. If α_k satisfies Wolfe conditions (4), then*

$$\alpha_k \geq \frac{1-\sigma}{L}\frac{|g_k^{\mathrm{T}} d_k|}{\| d_k\|^2}. \tag{21}$$

The following theorem states the global convergence property of Algorithm 1 for strongly convex functions under Assumptions 1 and 2.

THEOREM 3.4 *Suppose that f is a strongly convex function over $\mathcal{L}$, i.e. there is a constant $\mu > 0$ so that*

$$\mu \| x-y\|^2 \leq (\nabla f(x) - \nabla f(y))^{\mathrm{T}}(x-y) \quad \forall x, y \in \mathcal{L}.$$

Let $\{x_k\}$ be the sequence of points generated by Algorithm 1. Then, Algorithm 1 either stops at an iterate k with $g_k = 0$, or $\lim_{k\to\infty} g_k = 0$.

Proof Assume that $g_k \neq 0$, for all k. Due to strong convexity of f, one has

$$y_k{}^{\mathrm{T}} d_k \geq \mu\alpha_k \| d_k\|^2. \tag{22}$$

Besides, Lemma 3.1 implies that $d_k \neq 0$, and therefore $y_k^{\mathrm{T}} d_k > 0$ by (22). Adding up both sides of the first inequality in (4) over k and using Assumption 1, we get

$$\sum_{k=0}^{\infty} \alpha_k g_k{}^{\mathrm{T}} d_k > -\infty. \tag{23}$$

This inequality together with (18) and (21) imply that

$$\sum_{k=0}^{\infty} \frac{\| g_k\|^4}{\| d_k\|^2} < \infty. \tag{24}$$

On the other hand, from Assumption 2, we have

$$\begin{aligned} \| y_k \| &= \| g_{k+1} - g_k \| = \| \nabla f(x_k + \alpha_k d_k) - \nabla f(x_k) \| \\ &\leq L\alpha_k \| d_k \| . \end{aligned} \tag{25}$$

Now, using (22) and (25), we obtain:

$$\begin{aligned} |\beta_k| &= \left| \frac{g_{k+1}^{\mathrm{T}}(y_k - ts_k)}{d_k^{\mathrm{T}} s_k} \right| \\ &\leq \frac{\| g_{k+1} \| \| y_k - ts_k \|}{|d_k^{\mathrm{T}} y_k|} \\ &\leq \frac{\| g_{k+1} \| (\| y_k \| + t\alpha_k \| d_k \|)}{\alpha_k \mu \| d_k\|^2} \\ &\leq \frac{\| g_{k+1} \| (L\alpha_k \| d_k \| + t\alpha_k \| d_k \|)}{\alpha_k \mu \| d_k\|^2} \\ &= \frac{L+t}{\mu}\frac{\| g_{k+1} \|}{\| d_k \|}. \end{aligned}$$

Letting $c = (L + t)/\mu$ and using (3), one has

$$\begin{aligned} \| d_{k+1} \| &\leq \| g_{k+1} \| + |\beta_k| \| d_k \| \\ &\leq \| g_{k+1} \| + c \frac{\| g_{k+1} \|}{\| d_k \|} \| d_k \| \\ &= (1 + c) \| g_{k+1} \| . \end{aligned}$$

This inequality together with (24) yield

$$\sum_{k=0}^{\infty} \| g_k \|^2 < \infty, \tag{26}$$

which completes the proof of theorem. ■

Now, using β_k^+, we can establish the global convergence property of Algorithm 1 for general non-convex functions under Assumptions 1 and 2 in the sense that $\liminf_{k\to\infty} g_k = 0$.

THEOREM 3.5 *Suppose that Assumptions* 1 *and* 2 *hold and f is a non-convex function. Let* $\{x_k\}$ *be the sequence of points generated by Algorithm 1. Then, Algorithm 1 either stops at an iterate k with* $g_k = 0$, *or* $\liminf_{k\to\infty} g_k = 0$.

Proof On the contrary, suppose that $\liminf_{k\to\infty} g_k \neq 0$. Then, there exists a constant $\varepsilon > 0$ so that $\| g_k \| \geq \varepsilon$, for all $k \geq 0$.

Following the same proof line of Theorem 3.2 in [12], one can easily show that $|\beta_k| \leq C \| s_k \|$, which in turn states that β_k is bounded. Furthermore, using Theorem 3.2 in [12], one can see that $\| s_k \|$ and $\| d_k \|$ are bounded.

On the other hand, (24) implies that

$$\epsilon^4 \sum_{k=0}^{\infty} \frac{1}{\| d_k \|^2} \leq \sum_{k=0}^{\infty} \frac{\| g_k \|^4}{\| d_k \|^4} < \infty. \tag{27}$$

This shows that $\| d_k \| \to \infty$, which is a contradiction with the fact that $\| d_k \|$ is bounded. This completes the proof of theorem. ■

4. Numerical results

In this section, our aim is to investigate the practical behaviour of Algorithm 1 on some mostly large-scale test problems. We have also compared the performance of Algorithm 1 with CG_DESCENT Version 5.3. The algorithms are implemented in MATLAB 8.3.0.532 (R2014a) on a PC (Intel(R) Core(TM) i7-4510u CPU @2.00GHz 2.60 GHz 8.00 GB RAM) with Windows operating system. Our experiments are performed on a set of 120 unconstrained optimization problems taken from [2]. Table 1 provides the list of considered problems along with their dimensions. We set $\rho = 0.1$ and $\sigma = 0.9$ in the Wolfe conditions (4). Both algorithms are being stopped

Table 1. Specification of test functions, taken from [2].

Prob.	Dim.
The Fletcher–Powell helical valley function	50
The Biggs EXP6 function	50
The Gaussian function	50
The Powell badly scaled function	50
The Box 3-dimensional function	50
The variably dimensioned function	50
The Penalty Function 1	50
The Penalty Function 2	50
The Brown Badly Scaled Function	50
The BDQRTIC function	50
The Gulf R& D Function.	50
The Leon cubic valley function	50
The Gregory and Karney Tridiagonal Matrix Function	50
The De Jong Function F1	50
The De Jong Function F2	50
The De Jong Function F3, (discontinuous)	50
The De Jong Function F4 (with Gaussian noise)	50
The De Jong Function F5	50
The Schaffer Function F6	50
The Goldstein Price Polynomial	50
The Goldstein Price Polynomial	50
The Branin RCOS Function	50
The Six-Hump Camel-Back Polynomial	50
The Shubert Function	50
The Easom Function	50
The Bohachevsky Function 1	50
The Bohachevsky Function 2	50
The Bohachevsky Function 3	50
The Colville Polynomial	50
The Himmelblau function	50
ROSEN	50
Freudenstein and Roth	50
Bard	50
Meyer	50
Kowalik and Osborne function	50
Brown and Dennis	50
Osborne 1	50
Osborne 2	50
Broyden tridiagonal function	50
Broyden Banded	50
Linear — Full Rank	50
Linear — Rank 1	50
Linear - Rank 1 with Zero Cols. & Rows	50
CUBE function	50
EXPLIN1 function	50
Diagonal 6	50
DIXON3DQ function (CUTE)	50
SINE function	50
Powell Singular	50
The Hilbert Matrix Function F = xAx	100
FLETCBV3	100
LIARWHD function	100
POWER function	100
INDEF function (CUTE)	100
Diagonal 9 function	100
CURLY20 function	100
HIMMELBG function (CUTE)	200
Extended penalty function	200
The Trigonometric Function	500

(*Continued*).

Table 1. Continued.

Prob.	Dim.
The Beale Function.	500
Generalized Rosenbrock function	500
Raydan 1 function	500
Diagonal 1 function	500
Generalized White and Holst function	500
Full Hessian FH2 function	500
Extended Hiebert function	500
Quadratic QF1 function	500
FLETCHER function	500
TRIDIA function	500
QUARTC function	500
Hager function	500
The Extended Rosenbrock parabolic valley Function	1000
The Wood Function	1000
The Chebyquad Function	1000
Extended Freudenstein & Roth function	1000
Diagonal 3 function	1000
Generalized Tridiagonal-2	1000
Perturbed quadratic diagonal function	1000
Quadratic QF2 function	1000
ARWHEAD function (CUTE)	1000
QUARTC function	1000
BIGGSB1 function (CUTE)	1000
Diagonal 2 function	1000
Extended Three Exponential Terms	1000
Generalized PSC1 Function	1000
Extended Quadratic Penalty QP2 Function	1000
EG2 function	1000
Perturbed Tridiagonal Quadratic function	1000
Extended White & Holst function	2000
Perturbed Quadratic function	2000
NONDIA function	2000
Broyden tridiagonal function	2000
NONDQUAR function	2000
Extended Tridiagonal-2 Function	5000
Almost Perturbed Quadratic function	5000
Extended DENSCHNB function	5000
Extended DENSCHNF function	5000
Extended Quadratic Penalty QP1 Function	5000
Extended Beale function	10,000
Extended Powell function	10,000
Raydan 2 function	10,000
Generalized Tridiagonal 1 function	10,000
Extended Tridiagonal 1 function	10,000
Diagonal 4 function	10,000
Diagonal 5 function	10,000
Extended Himmelblau function	10,000
Full Hessian FH1 function	10,000
Extended Block Diagonal BD1 Function	10,000
Extended Maratos Function	10,000
Extended Cliff	10,000
Extended Wood function	10,000
Extended EP1 Function	10,000
DQDRTIC function (CUTE)	10,000
Extended Cliff	10,000
Diagonal 8 function	10,000
Full Hessian FH3 function	10,000
SINCOS function	10,000
HIMMELH function (CUTE)	10,000
Diagonal 7 function	10,000
Generalized Quartic function	10,000

either

$$\| g_k \|_\infty < 10^{-6},$$

or the number of iterations or function evaluations exceed 10,000 and 50,000, respectively. The following parameters are employed during Algorithm 1's run:

$$\omega = 10^{-4}, \quad \Omega = 10^{4}.$$

It is worth mentioning that the formula in [7] is used instead of (7) in Version 5.3 of CG_DESCENT. Moreover, the reason for considering CG_DESCENT algorithm in our comparisons is that this algorithm outperforms L-BFGS and majority of CG algorithms in the literature [12].

Note that, as a large set of test problems are considered, in order to shorten the length of the manuscript, the numerical results have been posted in the second author's home page at:

http://wp.kntu.ac.ir/peyghami/pdf/CUCG.xlsx

The comparison between considered algorithms is based on number of iterations, number of function evaluations and number of gradient evaluations. In order to have a proper comparison, we have utilized the performance profile of Dolan and Moré [10]. Figures 1–3 illustrate the performance profile of the considered algorithms in $\log_2$ scale.

Figures 1 and 2 show the performance of considered algorithms in terms of number of iterations and function evaluations, respectively. At a glance to these figures, one can figure out that Algorithm 1 performs well in comparison with CG_DESCENT. As it is clear from these figures, Algorithm 1 solves roughly 67% of test problems in less number of iterations and more than 69% of them in less function evaluations. Besides, Algorithm 1 solves more than 90% of the test problems successfully, while CG_DESCENT solves about 85% of the problems. Figure 3 makes it obvious that Algorithm 1 solves about 75% of test problems in less number of gradient evaluations.

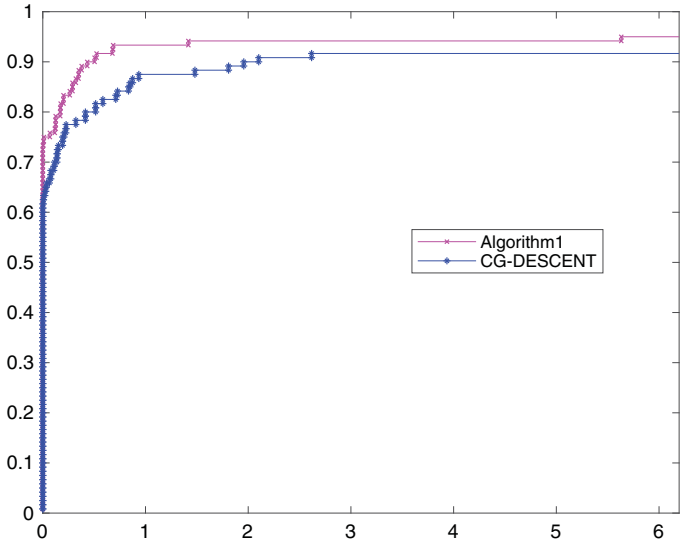

Figure 1. Performance profiles of considered algorithms in terms of number of iterations.

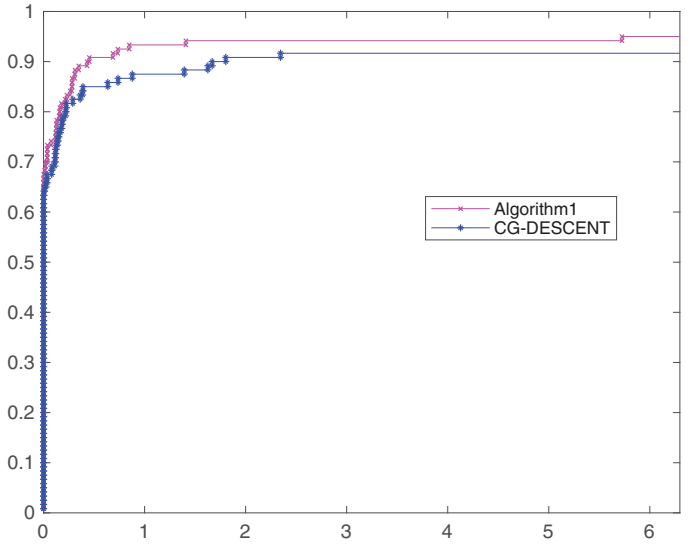

Figure 2. Performance profiles of considered algorithms in terms of number of function evaluations.

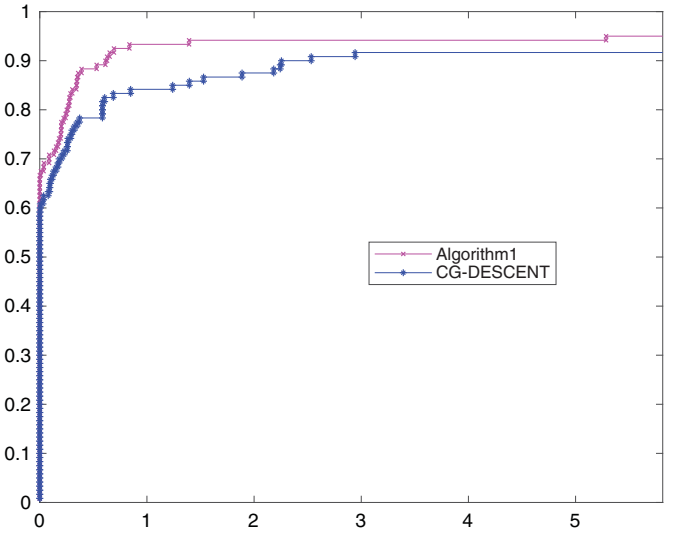

Figure 3. Performance profiles of considered algorithms in terms of number of gradient evaluations.

To demonstrate the decreasing behaviour of $\|g(x)\| = \|\nabla f(x)\|$ on some particular problems, the convergence history of considered algorithms is drawn in Figures 4 and 5 for the following functions [2]:

- *FLETCBV3 function (CUTE)*:

$$f(x) = \frac{1}{2}p({x_1}^2 + {x_n}^2) + \sum_{i=1}^{n-1}\frac{p}{2}(x_i - x_{i+1})^2 - \sum_{i=1}^{n}\frac{p(h^2+2)}{h^2}x_i + \frac{cp}{h^2}\cos(x_i),$$

where $n = 100$, $p = 10^{-8}$, $h = 1/(n+1)$, $c = 1$ and $x_0 = [h, 2h, \ldots nh]$.

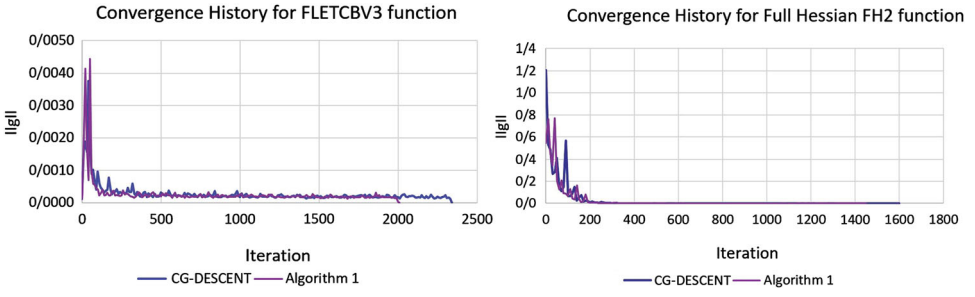

Figure 4. Convergence history of FLETCBV3 and FH2 functions.

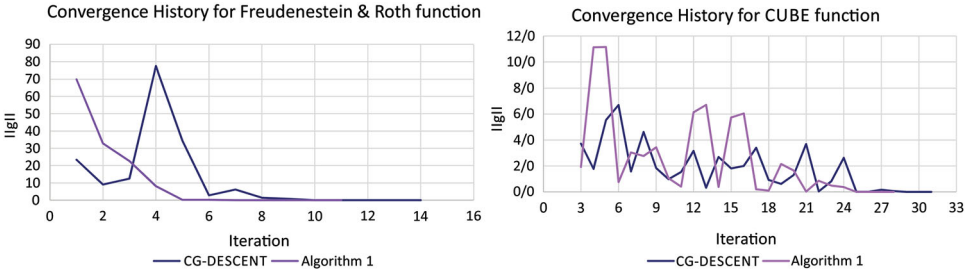

Figure 5. Convergence history of CUBE and Freudenstein& Roth functions.

- *Full Hessian FH2 function*:

$$f(x) = (x_1 - 5)^2 + \sum_{i=2}^{500} (x_1 + x_2 + \cdots + x_i - 1)^2,$$

where $x_0 = [0.01, 0.1, \ldots, 0.1] \in \mathbb{R}^{500}$.

- *The Extended Freudenstein and Roth function*:

$$f(x) = \sum_{i=1}^{500} (-13 + x_{2i-1} + ((5 - x_{2i})x_{2i} - 2)x_{2i})^2 + (-29 + x_{2i-1} + ((x_{2i} + 1)x_{2i} - 14)x_{2i})^2,$$

where: $x_0 = [0.5, -2, 0.5, \ldots, 0.5, -2] \in \mathbb{R}^{1000}$.

- *CUBE function (CUTE)*:

$$f(x) = (x_1 - 1)^2 + 100(x_2 - {x_1}^3)^2, \quad x_0 = [-1.2, 1].$$

Finally, the number of utilizing the new proposed stepsize (16) for the negative curvature during the solution process of the above mentioned functions is reported in Table 2.

Table 2. Number of using new stepsize for negative curvature.

Prob.	Iter.	New stepsize usage
FLETCBV3 function (CUTE)	2005	73
Full Hessian FH2 function	1573	79
The Extended Freudenstein & Roth function	12	4
CUBE function (CUTE)	29	3

5. Conclusion

In this paper, a new conjugate gradient algorithm in the Dai–Liao family for large-scale unconstrained optimization problems is proposed. Despite CG_DESCENT algorithm that only uses the quadratic model of the objective in its structure, our approach employs both quadratic and cubic regularization models to introduce positive values for the so-called Dai–Liao parameter. The cubic model is used when a non-positive curvature is detected. Moreover, the CG parameter is computed so that the generated CG directions satisfy descent condition. Under some standard assumptions, the global convergence property of the new proposed algorithm is investigated. Numerical results on some test problems show the efficiency and effectiveness of the new algorithm in practice. The results reveal that the new algorithm can be competitive with CG_DESCENT method.

Acknowledgements

The authors would like to thank the Research Council of K.N. Toosi University of Technology and SCOPE Research Center for supporting this work.

Disclosure statement

No potential conflict of interest was reported by the authors.

References

[1] N. Andrei, *Open problems in nonlinear conjugate gradient algorithms for unconstrained optimization*, Bull. Malays. Math. Sci. Soc. 34(2) (2011), pp. 319–330.
[2] N. Andrei, *An unconstrained optimization test functions collection*, Adv. Model. Optim. 10(1) (2008), pp. 147–161.
[3] R. De Asmundis, D. di Serafino, F. Reccio, and G. Toraldo, *On spectral properties of steepest descent methods*, IMA J. Numer. Anal. 33 (2013), pp. 1416–1435.
[4] S. Babaie-Kafaki and M. Fatemi, *A modified two point stepsize gradient algorithm for unconstrained minimization*, Optim. Methods Softw. 28(5) (2013), pp. 1040–1050.
[5] J. Barzilai and J.M. Borwein, *Two-point step size gradient methods*, IMA J. Numer. Anal. 8(1) (1988), pp. 141–148.
[6] F.E. Curtis and W. Guo, *Handling nonpositive curvature in a limited memory steepest descent method*, IMA J. Numer. Anal. 36(2) (2016), pp. 717–742.
[7] 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(1) (2013), pp. 296–320.
[8] Y.H. Dai and L.Z. Liao, *New conjugacy conditions and related nonlinear conjugate gradient methods*, Appl. Math. Optim. 43 (2001), pp. 87–101.
[9] Y.H. Dai and L.Z. Liao, *R-linear convergence of the Barzilai and Borwein gradient method*, IMA J. Numer. Anal. 22 (2002), pp. 1–10.
[10] E.D. Dolan and J.J. Moré, *Benchmarking optimization software with performance profiles*, Math. Program. 91 (2002), pp. 201–213.
[11] A. Griewank, *The modification of Newton's method for unconstrained optimization by bounding cubic terms*, Technical Report NA/12 (1981), Department of Applied Mathematics and Theoretical Physics, University of Cambridge, United Kingdom, 1981.
[12] W.W. Hager and H. Zhang, *A new conjugate gradient method with guaranteed descent and an efficient line search*, SIAM J. Optim. 16 (2005), pp. 170–192.
[13] W.W. Hager and H. Zhang, *A survay of nonlinear conjugate gradient methods*, Pacific J. Optim. 2 (2006), pp. 35–58.
[14] J. Nocedal and S.J. Wright, *Numerical Optimization*, 2nd ed., Springer, New York, 2006.
[15] J.M. Perry, *A class of conjugate gradient algorithms with a two-step variable-metric memory*, Discussion Paper269, Center of Mathematical Studies in Economics and Management Sciences, Northwestern University, Evanston, IL, USA, 1977.
[16] M. Raydan, *On the Barzilai and Borwein choise of steplength for the gradient method*, IMA J. Numer. Anal. 13 (1993), pp. 321–326.
[17] M. Raydon, *The Barzilai and Borwein gradient method for large scale unconstrained minimization problem*, SIAM J. Optim. 7(1) (1997), pp. 26–33.

[18] R.B. Schnabel and T.T. Chow, *Tensor methods for unconstrained optimization using second derivatives*, SIAM J. Optim. 1 (1991), pp. 293–315.
[19] R.B. Schnabel and P.D. Frank, *Tensor methods for nonlinear equations*, SIAM J. Numer. Anal. 21 (1984), pp. 815–843.
[20] D.F. Shanno, *On the convergence of a new conjugate gradient algorithm*, SIAM J. Numer. Anal. 15 (1978), pp. 1247–1257.
[21] Y. Xiao, H. Song, and Z. Wang, *A modified conjugate gradient algorithm with cyclic Barzilai-Borwein steplength for unconstrained optimization*, J. Comp. Appl. Math. 236 (2012), pp. 3101–3110.
[22] Y.X. Yuan, *A new stepsize for the steepest descent method*, J. Comput. Math. 24(2) (2006), pp. 149–156.
[23] G. Zoutendijk, *Nonlinear programming computational methods*, in *Integer and Nonlinear Programming*, J. Abadei, North-Holland, Amsterdam, 1970, pp. 37–86.