BANGMOD INTERNATIONAL JOURNAL OF MATHEMATICAL & COMPUTATIONAL SCIENCE
Volume 9 (2023)
Pages 24–44
ISSN: 2408-154X
https://bangmodjmcs.com/

A HYBRID CONJUGATE GRADIENT METHOD FOR UNCONSTRAINED OPTIMIZATION WITH APPLICATION

Nasiru Salihu[1,2,*], **Huzaifa Aliyu Babando**[2], **Ibrahim Arzuka**[1,3], **Suraj Salihu**[4]
[1] *Center of Excellence in Theoretical and Computational Science (TaCS-CoE), King Mongkut's University of Technology Thonburi (KMUTT), Bangkok, Thailand*
E-mails: nsalihu@mautech.edu.ng/nasirussalihu@gmail.com
[2] *Department of Mathematics, Faculty of Sciences, Modibbo Adama University, Yola, Nigeria*
E-mails: hababando@mau.edu.ng
[3] *Department of Mathematics, Bauchi State University, Gadau, Nigeria*
E-mails: arzuka2000@gmail.com
[4] *Department of Computer Science, Gombe State University, Nigeria*
E-mails: surajsalihu@gmail.com

**Corresponding author.*

Received: 9 August 2023 / Accepted: 25 November 2023

Abstract This article considers a hybrid minimization algorithm from optimal choice of the modulating non-negative parameter of Dai-Liao conjugacy condition. The new hybrid parameter is selected in such away that a convex combination of Hestenes-Stiefel and Dai-Yuan Conjugate Gradient (CG) algorithms is fulfilled. The numerical implementation adopts inexact line search which reveals that the scheme is robust when compared with some known efficient algorithms in literature. Furthermore, the theoretical analysis shows that the proposed hybrid method converges globally. The method is also applicable to solve three degree of freedom motion control robotic model.

MSC: 65K05; 90C30
Keywords: Unconstrained optimization; Hybrid conjugate gradient method; Sufficient descent; Global convergence; Motion control

Published online: 31 December 2023

Please cite this article as: N. Salihu et al., A hybrid conjugate gradient method for unconstrained optimization with application, Bangmod J-MCS., Vol. 9 (2023) 24–44.

TaCS Center of Excellence Publications
https://doi.org/10.58715/bangmodjmcs.2023.9.3

1. Introduction

Fletcher and Reeves in 1964 developed non-linear conjugate gradient method for minimization of an unconstrained problem of the form

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

where $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ is a function that is smooth and its gradient is designated by $g(x) = \nabla f(x)$. Conjugate Gradient (CG) schemes constitute an exceptional choice for solving scientific and engineering problems in the form of (1.1), because of their nice theoretical properties and modest memory requirements [1, 2]. The method can be model into many real life problems arising from; portfolio choice [3, 4], m-tensor system [5, 6], image restoration [7, 8], signal recovery [9, 10] and three degrees of freedom (3DOF) robotic motion [11–13]. Starting with $x_0 \in \mathbb{R}^n$ as an initial guess for the minimizer, the algorithm produces sequence of points $\{x_k\}$ using the following iterative procedure

$$x_{k+1} = x_k + \alpha_k d_k,\ k = 0, 1, 2, \dots, \tag{1.2}$$

where the step-size $\alpha_k > 0$ is obtainable by suitable technique through the search direction d_k. For example, any value of the step-size α_k that satisfies certain conditions is preferred [14]. The weak Wolfe line technique consisting the following inequalities

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \delta \alpha_k g_k^T d_k, \tag{1.3}$$

$$g(x_k + \alpha_k d_k)^T d_k \geq \sigma g_k^T d_k. \tag{1.4}$$

On the other hand, the strong Wolfe condition is given by (1.3) and

$$|g(x_k + \alpha_k d_k)^T d_k| \leq -\sigma g_k^T d_k, \tag{1.5}$$

where $0 < \delta < \sigma < 1$. The search direction d_k is computed from

$$d_0 = -g_0,\ d_{k+1} = -g_{k+1} + \beta_k d_k,\ \forall k \geq 1, \tag{1.6}$$

with β_k, called update parameter that determines the choice of a method.

Various selections for the scalar parameter β_k would produce different CG methods with quite different theoretical and numerical features [15]. Therefore, some earlier proposed CG formulas include; Fletcher and Revees (FR)[16], Dai and Yuan (DY) [17], Fletcher (Conjugate Descent (CD))[18], Hestenes and Stiefel (HS) [19], Polak, Ribière and Polyak (PRP) [20, 21], and Liu and Storey (LS)[22]. Recent survey of CG features shows that DY scheme possesses strong convergence property but numerically uncertain due to jamming [23]. To address this drawback, researchers have developed interest in combining this method with other CG methods that are numerically stable. For example, HS method, despite its promising performance, the method is also affected by convergence problem [24]. Let $\|\cdot\|$ denotes Euclidean norm and define $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$. Thus, the strength of HS and DY with the following β_k:

$$\beta_k^{HS} = \frac{g_{k+1}^T y_k}{d_k^T y_k}, \tag{1.7}$$

$$\beta_k^{DY} = \frac{\|g_{k+1}\|^2}{d_k^T y_k}, \tag{1.8}$$

were examined to avoid their weaknesses [25, 26]. Recently, the idea of combining different CG methods to improve their numerical and theoretical structures attracted more attention [27], especially, the β_k parameters that are derived based on the following classical conjugacy condition

$$d_{k+1}^T y_k = 0. \tag{1.9}$$

The condition (1.9) has been generalized by Perry [28] and Dai and Liao [29]. The Perry and Dai-Liao conjugacy conditions are respectively given as

$$d_{k+1}^T y_k = -g_{k+1}^T s_k \tag{1.10}$$

$$d_{k+1}^T y_k = -t g_{k+1}^T s_k, \ t \geq 0. \tag{1.11}$$

It is important to note that, when $t = 0$, equation(1.11) reduces to (1.9), and if $t = 1$, then Perry's condition is obtained.

As mentioned above, different CG parameters have been proposed in such a way that one of the conditions (1.9)-(1.11) is satisfied. For example, based on the condition (1.11), Dai-Liao [29] proposed another CG method defined by

$$\beta_k^{DL} = \frac{g_{k+1}^T y_k}{d_k^T y_k} - t\frac{g_{k+1}^T s_k}{d_k^T y_k}.$$

where $y_k = g_{k+1} - g_k, s_k = x_{k+1} - x_k$ and $t > 0$ is called Dai-Liao parameter. In order to establish the global convergence for general function, Dai and Liao [29] adjusted the above formula as follows:

$$\beta_k^{DL*} = \max\{\frac{g_{k+1}^T y_k}{d_k^T y_k}, \ 0\} - t\frac{g_{k+1}^T s_k}{d_k^T y_k}, \tag{1.12}$$

for some parameter t.

It can be observed that if $t = 0$, then β_k^{DL} scale down to HS [19] CG formula defined in (1.7). Similar to some classical CG methods, the DL method is globally convergent for uniformly convex objective functions, but, its convergence for general functions depends on the non-negative parameter t [30]. In addition, the DL method may fail to generate sufficient descent direction [31], that is, $d_k^T g_k \leq -c\|g_k\|^2, \ c > 0$. Consequently, by considering the self-scaling memoryless BFGS scheme, Hager and Zhang [32] extended the above idea to construct another new formula as follows:

$$\beta_k^N = \frac{g_{k+1}^T y_k}{d_k^T y_k} - 2\frac{\|y_k\|^2}{(d_k^T y_k)^2} g_{k+1}^T s_k, \tag{1.13}$$

and showed that (1.13) satisfies the descent condition $g_{k+1}^T d_{k+1}^T \leq \frac{7}{8}\|g_{k+1}\|^2$. To show that the method is globally convergence for general functions, Hager and Zhang [32] presented the following restricted version of (1.13):

$$\beta_k^{N+} = \max\{\beta_k^N, \frac{-1}{\|d_k\| \min\{\eta, \|g_k\|\}}\}. \tag{1.14}$$

Results from numerical computations has shown that the method is efficient and promising. Furthermore, the DL-like methods proposed in [33] and [34] happened to be globally convergent and numerically stable, but like the method in [29], they also fail to fulfil the sufficient descent condition. To overcome the defect with DL CG versions; using singular value study, Babaie-Kafaki and Ghanbari [35] and Andrei [36] proposed an adaptive optimal choices for t, which increased the numerical strength of the DL methods. Numerical experiments show that these algorithms are robust and more efficient than Hager and Zhang [32] CG method. Despite the fact that different choices of the parameter t have

been suggested in [24, 31, 35, 37–40], and for nice review on recent advances on Dai-Liao methods by Saman [41], the optimal choice of t in DL-type methods still requires more attention, especially with hybrid CG methods.
Motivated by the above discussions and the idea in [42], we introduce a hybrid CG method for unconstrained optimization problem with the CG parameter taken as the convex combination of the HS and DY CG parameters. The convex combination parameter is derived based on the Dai-Liao conjugacy condition (1.11). Thus, the proposed hybrid method takes HS method [19] and DY method [17] as special cases. Some of the major contributions of this paper can be highlighted as follows:

(1) A new hybrid conjugate gradient parameter is proposed based on the concept of extended conjugacy condition.
(2) The search direction of the new method is sufficient descent using strong Wolfe line search procedure.
(3) The efficiency of proposed scheme is illustrated on variety of large scale benchmark unconstrained optimization problems.
(4) The method is shown to converge globally based on some standard rules.
(5) Finally, the new hybrid algorithm is applied to solve three degrees of freedom robotic model.

This article is organized as follows: Preliminaries and description of the hybrid method are given in Section 2. In Section 3, the theoretical analysis of the method are presented. Investigation of the practical implementation of the method are reported in Section 4. Finally, in Section 5, conclusions are made.

2. Hybrid Method and its algorithm

Here, in order to strengthen the behaviour of CG updating parameters proposed by Hestenes and Stiefel [19] with Dai and Yuan [17] CG methods, we proposed their convex combination using relations (1.7) and (1.8) as

$$\begin{aligned}\beta_k &= (1-\theta_k)\beta_k^{HS} + \theta_k\beta_k^{DY} \\ &= \beta_k^{HS} + \theta_k(\beta_k^{DY} - \beta_k^{HS}) \\ &= \frac{g_{k+1}^T y_k}{d_k^T y_k} + \theta_k\left(\frac{g_{k+1}^T g_{k+1}}{d_k^T y_k} - \frac{g_{k+1}^T y_k}{d_k^T y_k}\right) \\ &= \frac{g_{k+1}^T y_k}{d_k^T y_k} + \theta_k\frac{g_{k+1}^T g_k}{d_k^T y_k}.\end{aligned} \tag{2.1}$$

Employing the choice of parameter t in [42], implies that, (1.11) becomes

$$d_{k+1}^T y_k = -\left(\frac{s_k^T y_k + \|y_k\|\|s_k\|}{\|s_k\|^2}\right) g_{k+1}^T s_k, \tag{2.2}$$

where $t = \dfrac{s_k^T y_k}{\|s_k\|^2} + \dfrac{\|y_k\|}{\|s_k\|}$.

Multiplying (1.6) by y_k^T and using (2.1) and (2.2), we have

$$
\begin{aligned}
d_{k+1}^T y_k &= -g_{k+1}^T y_k + \beta_k d_k^T y_k \\
&= g_{k+1}^T y_k + \left(\frac{g_{k+1}^T y_k}{d_k^T y_k} + \theta_k \frac{g_{k+1}^T g_k}{d_k^T y_k} \right) d_k^T y_k \\
&= -g_{k+1}^T y_k + (g_{k+1}^T y_k + \theta_k g_{k+1}^T g_k) \\
&= \theta_k g_{k+1}^T g_k. \qquad (2.3)
\end{aligned}
$$

Equating (2.2) with (2.3) gives

$$\theta_k g_{k+1}^T g_k = -\left(\frac{s_k^T y_k + \|y_k\| \|s_k\|}{\|s_k\|^2} \right) g_{k+1}^T s_k$$

By rearranging, we have

$$\theta_k = -\left(\frac{s_k^T y_k + \|y_k\| \|s_k\|}{\|s_k\|^2} \right) \frac{g_{k+1}^T s_k}{g_{k+1}^T g_k}. \qquad (2.4)$$

Based on these relations, we described the implementation of the new hybrid method as follows:

Algorithm 1: ECCHD

Step 1: Select $x_0 \in \mathbb{R}^n$ and parameter $0 < \delta < \sigma \leq 0.3$. Consider $d_0 = -g_0$ and set $\alpha_0 = 1$.
Step 2: Check for convergence: If $\|g_k\| \leq 10^{-6}$, then stop. Otherwise
Step 3: Compute $\alpha_k > 0$ such that (1.3) and (1.5) are satisfied.
Step 4: Compute θ_k using (2.4).
Step 5: Compute β_k:

$$\beta_k = \begin{cases} \beta_k^{HS}, & if\ \theta_k \leq 0, \\ (1-\theta_k)\beta_k^{HS} + \theta_k \beta_k^{DY}, & if\ 0 < \theta_k < 1, \\ \beta_k^{DY}, & if\ \theta_k \geq 1. \end{cases}$$

Step 6: Compute $d_{k+1} = -g_{k+1} + \beta_k d_k$. If restart criterion of Powell

$$|g_{k+1}^T g_k| > a\|g_{k+1}\|^2, \text{ where } a = 0.2 \qquad (2.5)$$

is satisfied, then set $d_{k+1} = -g_{k+1}$.
Step 7: Set $k = k+1$ and go to Step 2.

Remark 2.1. The convex combination parameter θ_k is more general than that of Andrei [43], that is, $\widehat{\theta}_k = -\dfrac{s_k^T g_{k+1}}{g_k^T g_{k+1}}$. In addition, We select the parameter θ_k in such away that, if $\theta_k \leq 0$, we set $\beta_k = \beta_k^{HS}$ and if $\theta_k \geq 1$, we set $\beta_k = \beta_k^{DY}$. Otherwise, if $0 < \theta_k < 1$, then β_k includes both β_k^{HS} and β_k^{DY}.

3. Convergence Analysis

Now we will show that d_k, the search direction obtained by ECCHD Algorithm is sufficient descent.

Lemma 3.1. *Let $\sigma \in (0, 0.3]$ and suppose the sequences $\{g_k\}$ and $\{d_k\}$ are generated by ECCHD Algorithm. Then the search direction satisfies:*

$$d_{k+1}^T g_{k+1} \leq -c\|g_{k+1}\|^2, \ \forall k \geq 0, \tag{3.1}$$

where $c = \dfrac{(1-3.2\sigma)}{(1-\sigma)}$.

Proof. Suppose that restart criterion of Powell [44] condition (2.5) holds in ECCHD Algorithm, that is, $d_{k+1} = -g_{k+1}$, then (3.1) holds. Let assume that (3.1) does not hold. Then, we have the following inequalities:

$$|g_{k+1}^T g_k| \leq 0.2\|g_{k+1}\|^2. \tag{3.2}$$

We show the proof by mathematical induction. Initially, it follows easily that $g_0^T d_0 = -\|g_0\|^2$, which implies that (3.1) is satisfied. Next, suppose the result in (3.1) holds for k, that is,

$$d_k^T g_k \leq -c\|g_k\|^2. \tag{3.3}$$

We now show for $k+1$. From the strong Wolfe Condition

$$|g_{k+1}^T d_k| \leq -\sigma d_k^T g_k. \tag{3.4}$$

Therefore, using (3.3), we have

$$d_k^T y_k = d_k^T g_{k+1} - d_k^T g_k \geq -(1-\sigma) d_k^T g_k \geq 0. \tag{3.5}$$

Multiplying (1.6) with g_{k+1}^T and using (2.1), we have

$$g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2 + ((1-\theta_k)\beta_k^{HS} + \theta_k \beta_k^{DY}) g_{k+1}^T d_k. \tag{3.6}$$

So, when $\theta_k \leq 0$, we set $\theta_k = 0$, which means $\beta_k = \beta_k^{HS}$, it follows from (1.7) and (3.6) that

$$\begin{aligned} d_{k+1}^T g_{k+1} &= -\|g_{k+1}\|^2 + \beta_k^{HS} d_k^T g_{k+1}, \\ &\leq -\|g_{k+1}\|^2 + \frac{|g_{k+1}^T y_k|}{d_k^T y_k} |d_k^T g_{k+1}|. \end{aligned} \tag{3.7}$$

Since $y_k = g_{k+1} - g_k$, we now use (3.2), to get

$$\begin{aligned} |g_{k+1}^T y_k| &\leq \|g_{k+1}\|^2 + |g_{k+1}^T g_k|, \\ &\leq \|g_{k+1}\|^2 + 0.2\|g_{k+1}\|^2, \\ &= 1.2\|g_{k+1}\|^2. \end{aligned}$$

Now, from the above inequalities and (3.4) , (3.5) and (3.7), we get

$$\begin{aligned}
g_{k+1}^T d_{k+1} &\leq -\|g_{k+1}\|^2 + \frac{1.2\|g_{k+1}\|^2}{d_k^T y_k}|d_k^T g_{k+1}| \\
&\leq -\|g_{k+1}\|^2 - \frac{1.2\sigma\|g_{k+1}\|^2}{d_k^T y_k} d_k^T g_k \\
&\leq -\|g_{k+1}\|^2 + \frac{1.2\sigma\|g_{k+1}\|^2}{(1-\sigma)} \\
&\leq -\frac{(1-2.2\sigma)}{1-\sigma}\|g_{k+1}\|^2 \\
g_{k+1}^T d_{k+1} &\leq -c_1\|g_{k+1}\|^2. \qquad (3.8)
\end{aligned}$$

Since $\sigma \in (0, 0.3]$. Also, when $\theta_k \geq 1$, we set $\theta_k = 1$, which implies that $\beta_k = \beta_k^{DY}$, and from (1.8) , (3.4),(3.5),(3.6), we obtain

$$\begin{aligned}
g_{k+1}^T d_{k+1} &\leq -\|g_{k+1}\|^2 + \frac{\|g_{k+1}\|^2}{d_k^T y_k}|d_k^T g_{k+1}| \\
&\leq -\frac{(1-2\sigma)}{1-\sigma}\|g_{k+1}\|^2 \\
&\leq -\frac{(1-2\sigma)}{1-\sigma}\|g_{k+1}\|^2 \\
g_{k+1}^T d_{k+1} &\leq -c_2\|g_{k+1}\|^2. \qquad (3.9)
\end{aligned}$$

Finally, if $\theta_k \in (0, 1)$, then the parameter θ_k is computed by (2.4). Indeed, it follows from (1.7), (1.8), (3.4), (3.5), (3.6) and $|g_{k+1}^T y_k| \leq 1.2\|g_{k+1}\|^2$, that

$$\begin{aligned}
d_{k+1}^T g_{k+1} &\leq -\|g_{k+1}\|^2 + |\beta_k^{HS}||d_k^T g_{k+1}| + |\beta_k^{DY}||d_k^T g_{k+1}| \\
&\leq -\|g_{k+1}\|^2 + \sigma|\beta_k^{HS}||d_k^T g_k| + \sigma|\beta_k^{DY}||d_k^T g_k| \\
&\leq -\|g_{k+1}\|^2 + \sigma\frac{|g_{k+1}^T y_k|}{|d_k^T y_k|}|d_k^T g_k| + \sigma\frac{\|g_{k+1}\|^2}{|d_k^T y_k|}|d_k^T g_k| \\
&\leq -\|g_{k+1}\|^2 + \frac{1.2\sigma\|g_{k+1}\|^2}{|d_k^T y_k|}|d_k^T g_k| + \frac{\sigma\|g_{k+1}\|^2}{|d_k^T y_k|}|d_k^T g_k| \\
&\leq -\|g_{k+1}\|^2 + \frac{1.2\sigma\|g_{k+1}\|^2}{(1-\sigma)} + \frac{\sigma\|g_{k+1}\|^2}{(1-\sigma)} \\
&\leq -\|g_{k+1}\|^2 + \frac{2.2\sigma}{(1-\sigma)}\|g_{k+1}\|^2 \\
&\leq -\left(1 - \frac{2.2\sigma}{1-\sigma}\right)\|g_{k+1}\|^2 \\
&\leq -\left(\frac{1-3.2\sigma}{1-\sigma}\right)\|g_{k+1}\|^2 \\
d_{k+1}^T g_{k+1} &\leq -c\|g_{k+1}\|^2, \qquad (3.10)
\end{aligned}$$

where the fifth inequality follows from (3.5) and since $\sigma \in (0, 0.3]$, this inequality shows that (3.1) holds for $k+1$. ∎

The following assumptions are necessary for the convergence analysis.

Assumption 3.1: *The level set $S = \{x \in \mathbb{R} : f(x) \leq f(x_0)\}$ is bounded and there exists a constant $B > 0$ such that*

$$\|x\| \leq B, \forall x \in S.$$

Assumption 3.2: *In a neighborhood N of S, the objective function f is continuously differentiable, its gradient is Lipschitz continuous and there exists a constant $L > 0$ such that*

$$\|g(x) - g(y)\| \leq L\|x - y\|, \tag{3.11}$$

for all $x, y \in N$. Under these assumptions, there exists a constant $\Gamma > 0$ such that.

$$\|g(x)\| \leq \Gamma, \tag{3.12}$$

for all $x \in S$.

Lemma 3.2. [45] *Suppose that Assumptions* 3.1 *and* 3.2 *hold. Consider the CG method* (1.2)*, where the search direction d_k is sufficient descent and α_k satisfies strong Wolfe condition, then*

$$\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < +\infty. \tag{3.13}$$

Lemma 3.3. *Suppose that the sequences $\{x_k\}$ and $\{d_k\}$ are generated by ECCHD Algorithm, and Assumptions* 3.1 *and* 3.2 *hold. If there exists a constant $\epsilon > 0$, such that*

$$\|g_k\| \geq \epsilon, \;\; \forall k \geq 0, \tag{3.14}$$

then by the second strong Wolfe condition and (3.1)*, we have*

$$d_k^T y_k = d_k^T g_{k+1} - d_k^T g_k \geq -(1-\sigma) d_k^T g_k \geq c(1-\sigma)\|g_k\|^2. \tag{3.15}$$

Theorem 3.4. *Suppose that the sequences $\{x_k\}$ and $\{d_k\}$ are generated by ECCHD Algorithm, where the search direction d_k is such descent and α_k satisfies strong Wolfe condition, then*

$$\liminf_{k \to \infty} \|g_k\| = 0. \tag{3.16}$$

Proof. Suppose on the contrary, that (3.16) does not hold, which means the gradient is bounded away from zero and there exists a constant $\epsilon > 0$, such that $\|g_k\| \geq \epsilon$.

Claim The search direction defined by (1.6) is bounded, i.e., there exists a constant $P > 0$, such that

$$\|d_{k+1}\| \leq P, \;\; \forall k \geq 0. \tag{3.17}$$

We prove this claim by induction. Let D be the diameter of the level set. Then from the Lipschitz continuity of the gradient, it follows that $\|y_k\| = \|g_{k+1} - g_k\| \leq L\|s_k\| \leq LD$.

https://doi.org/10.58715/bangmodjmcs.2023.9.3

Therefore, using (1.7), (1.8), (2.1) and (3.12), we have

$$\begin{aligned}|\beta_k| &= |(1-\theta_k)\beta_k^{HS} + \theta_k\beta_k^{DY}|\\ &\leq |\beta_k^{HS}| + |\beta_k^{DY}|\\ &\leq \frac{|g_{k+1}^T y_k|}{|d_k^T y_k|} + \frac{\|g_{k+1}\|^2}{|d_k^T y_k|}\\ &\leq \frac{\|g_{k+1}\|\|y_k\|}{c(1-\sigma)\|g_k\|^2} + \frac{\|g_{k+1}\|^2}{c(1-\sigma)\|g_k\|^2}\\ &\leq \frac{\|g_{k+1}\|LD + \|g_{k+1}\|^2}{c(1-\sigma)\|g_k\|^2}\\ &\leq \frac{\Gamma LD + \Gamma^2}{c(1-\sigma)\epsilon^2} = E. \end{aligned} \tag{3.18}$$

For $k = 0$, we have, $d_1 = -g_1 + \beta_1 d_0$, which implies that $d_1 = -g_1 - \beta_1 g_0$, since $d_0 = -g_0$. This yield

$$\begin{aligned}\|d_1\| &\leq \|g_1\| + |\beta_1|\|g_0\|\\ &\leq \Gamma + E\Gamma = \Gamma^*,\end{aligned}$$

that is, the claim (3.17) holds for $k = 0$ Next we assume that the claim (3.17) is true for k, that is, $\|d_k\| \leq P$. To show it is true for $k+1$, consider the search direction (1.6)

$$d_{k+1} = -g_{k+1} + \beta_k d_k.$$

Now, using (3.12) and (3.18), we obtain

$$\begin{aligned}\|d_{k+1}\| &\leq \|g_{k+1}\| + |\beta_k|\|d_k\|\\ &\leq \Gamma + EP,\end{aligned}$$

and therefore the claim holds. Now since (3.17) holds for all k, then we have

$$\frac{1}{\|d_k\|} \geq \frac{1}{P},\ \ P > 0. \tag{3.19}$$

From the above inequality, it shows that

$$\sum_{k=0}^{\infty} \frac{1}{\|d_k\|^2} = +\infty. \tag{3.20}$$

Considering (3.1), (3.13) and (3.14) we conclude that

$$c^2\epsilon^4 \sum_{k=0}^{\infty} \frac{1}{\|d_k\|^2} \leq \sum_{k=0}^{\infty} \frac{c^2\|g_k\|^4}{\|d_k\|^2} \leq \sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < +\infty. \tag{3.21}$$

It is obvious that, (3.20) and (3.21) cannot hold concurrently. Thus, (3.16) must hold. ∎

‘

4. Numerical Results

In this section, we present the implementation of ECCHD Algorithm on the set of 230 benchmark problems obtained from [46, 47]. The method is compared with the hybrid CG methods in [27, 43, 48, 49]. To implement CG parameters in all the methods, we use the following parameter; $\delta = 0.00001$ and $\sigma = 0.0001$, and the code is written in Matlab (R2018a) version and run on a personal computer with processor 2.20 GHz and memory 3.0 GB. The iteration is terminated when $\|g_k\|_{\leq} 10^{-6}$. Numerical results were compared based on performance profile of Dolan and Moré [50]. To visualize the performance of the methods, the test function results in the Figures 1-2 are achieved using Table 1 and by running each solver on the benchmark problems and recording the number of iterations and elapsed time to minimize the problems. The higher the solver goes, the more efficient is the method, that is, when the value of $P_s(\tau)$ is high. The $P_s(\tau)$ as given in [46], is the fraction from the set of problems, with the high appearance of τ ratio. Given the problem P and the optimization solver S respectively, the performance comparison of a problem by a particular algorithm is measured. So if we allow $P_s(\tau) = P(\tau)$ and $S = \tau$, then the numerical results were compared graphically. Figures 1-2 show the performance of the hybrid coefficients are compared based on number of iteration and central processing time per unit with 100 and 1000000 as the smallest and highest dimensions of the test problems respectively. The y-xis of the figures shows the fraction of how fast the coefficient converge while the x-axis determines the fraction of how many problems a solver is able to solve successively.

The analysis of Figure 1, for the value τ chosen within $0 < \tau < 0.5$ interval, shows the portion of ECCHD Algorithm is the best on the set of problems P is 54%. While HHD, FRPRPCC, HHSFR and CCOMB algorithms are 30%, 25%, 20% and 10% respectively. Clearly, ECCHD method is efficient and closer to the optimal solution with the highest probability. However, if we increase the τ to an interval $\tau \geq 0.5$, the ECCHD and HHD methods solved problems with 98% accuracy respectively in the elapsed time, while FRPRPCC, HHSFR and CCOMB algorithms is 94%. This shows that, the ECCHD and HHD methods are computationally efficient than other schemes. Meanwhile, if τ of interest is between $0.5 < \tau < 1.0$, the proposed method has 98% of the problems solved, against 84% of HHD. Therefore, we further make comparisons among the five schemes with the number of iterations in Figure 2, which shows ECCHD and HHD methods are the best on the given problems with 97% accuracy respectively. On the other hand, the HHD, FRPRPCC, HHSFR and CCOMB methods solve the problems with the following percentages 80%, 79%, 70% and 69% respectively, when the value of τ is within $0 < \tau < 0.05$ interval. Obviously, the ECCHD scheme has demonstrated to the best method. However, if we increase the τ to an interval $\tau \geq 0.05$, the ECCHD and HHD methods solved the benchmark problems with 97% number of iterations, while other methods attain 95%, this implies that the numerical results of ECCHD and HHD algorithms are computationally efficient than other schemes. Clearly, both figures indicated that the ECCHD is promising and efficient than other CG coefficients.

https://doi.org/10.58715/bangmodjmcs.2023.9.3
Bangmod J-MCS 2023

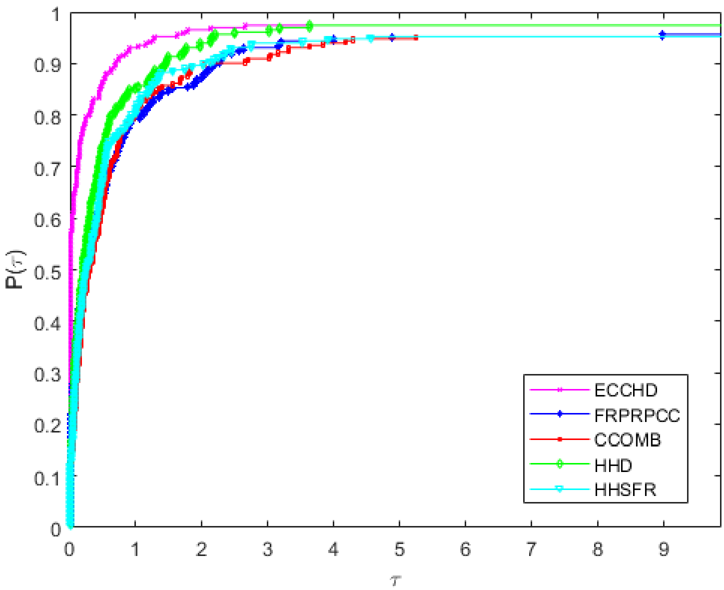

FIGURE 1. Time performance profiles of the methods.

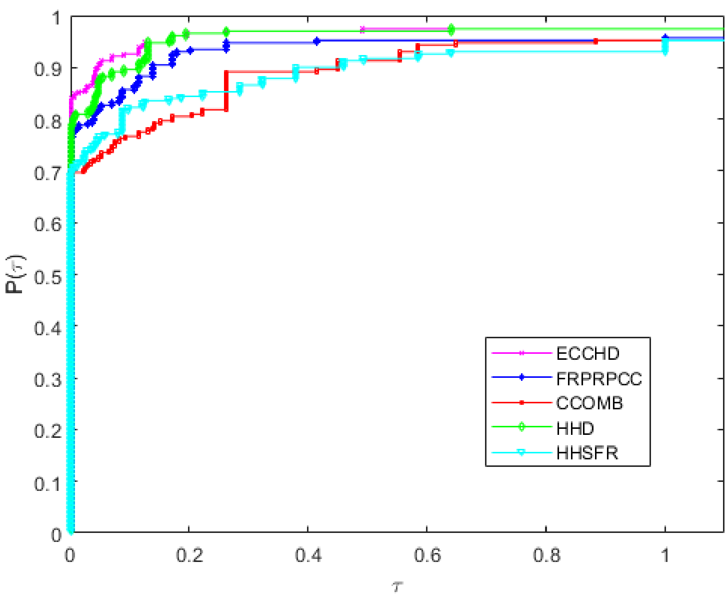

FIGURE 2. Number of iterations performance profiles of the methods.

Table 1. Numerical Results of FRPRPCC, CCOMB, HHD, ECCHD and HHSFR Methods

S/N	Test Functions	DIMM	IN. PT.	FRPRPCC		CCOMB		HHD		ECCHD		HHSFR	
				NI	CPUT	NI	CPUT	NI	CPUT	NI	CPUT	NI	CPUT
1	EXT. WHITE & HOLST	100	(-1.2,1,,-1.2,1)	9	11.1358	9	0.0786	9	0.1249	9	0.0372	9	0.0222
2		200		9	0.0243	9	0.0349	9	0.027	9	0.0261	9	0.0276
3		500		9	0.1354	9	0.0512	9	0.0668	9	0.048	9	0.0528
4		1000		9	0.0993	9	0.0784	9	0.1916	9	0.074	9	0.0866
5		2000		9	0.1923	9	0.2215	9	0.2007	9	0.1291	9	0.2149
6		5000		9	0.3605	9	0.4119	9	0.4987	9	0.3704	9	1.8073
7		10000		9	1.4406	9	0.7433	9	0.7542	9	0.6634	9	0.7506
8		20000		9	1.4008	9	1.8182	9	1.41	9	1.3482	9	1.441
9		50000		9	4.3154	9	3.3201	9	3.3965	9	3.2868	9	3.3691
10		100000		9	6.5216	9	6.663	9	6.6951	9	6.4014	9	6.7161
11		200000		9	16.2495	9	13.461	9	12.988	9	12.8021	9	13.27
12	POWER	100	(1,,1)	141	0.1564	141	0.1598	141	0.1575	141	0.106	141	0.1223
13		200		297	0.3928	296	0.265	297	0.269	296	0.3333	296	0.3735
14		500		771	0.8712	768	0.9454	769	0.9301	769	0.6165	770	0.9295
15		1000		1564	1.9313	1560	2.2459	1563	2.3713	1562	2.0863	1562	5.0019
16		2000		3180	15.6438	3151	16.3639	3169	19.299	3166	17.2814	3163	21.038
17		5000		8627	121.587	7985	130.331	8290	105.32	8280	70.0589	8241	74.183
18	QUADRATIC QF1	100	(1,,1)	56	2.6411	56	2.2889	56	1.1531	56	1.1494	56	0.8973
19		200		81	1.9852	81	2.6594	81	2.684	81	0.2409	81	0.2161
20		500		131	0.3755	131	0.3979	131	0.3345	131	0.2756	131	0.2913
21		1000		187	0.5925	187	0.4562	187	0.4444	187	0.3678	187	0.5003
22		2000		267	0.8061	267	0.8809	267	0.8166	267	0.7987	267	0.7807
23		5000		426	3.1434	426	8.3423	426	8.7483	426	5.2584	426	6.3023
24		10000		606	19.950	606	20.767	606	15.7457	606	13.4312	606	19.744
25		20000		862	41.886	862	53.5483	862	59.7891	862	57.2932	862	59.281
26		50000		1373	305.98	1373	219.48	1373	186.73	1373	182.774	1373	193.22
27	EXT. ROSENBROCK	100	(-1.2,1,,-1.2,1)	16	0.0336	16	0.1527	16	0.0199	16	0.0188	17	0.0629
28		200		16	0.0295	16	0.0348	16	0.0286	16	0.0249	17	0.0295
29		500		16	0.0629	16	0.0479	16	0.041	16	0.0399	17	0.044
30		1000		16	0.064	16	0.0584	16	0.0668	16	0.0587	17	0.3893
31		2000		16	0.4093	16	0.0964	16	0.1351	16	0.0901	17	0.1808
32		5000		16	0.9443	16	0.2803	16	0.2903	16	0.2523	17	0.3711
33		10000		16	0.6076	16	0.4581	16	0.5381	16	0.4586	17	0.6135
34		20000		16	0.8636	16	1.0742	16	0.8229	16	0.7883	17	1.342
35		50000		16	1.937	16	2.0762	16	1.9201	16	1.9755	17	2.2186
36	EXT. QUADRATIC P.	100	(1,,1)	20	0.2902	22	0.0766	21	0.0741	20	0.0735	20	0.0787
37		200		25	0.324	25	0.2507	25	0.5685	25	0.5111	25	0.7371
38		500		38	2.8968	38	10.340	37	12.845	37	3.4006	38	12.558
39		1000		41	6.3256	42	20.400	41	18.531	41	5.8633	42	7.988
40		2000		60	12.061	62	11.813	60	8.8443	61	6.8858	61	4.739
41		5000		F	F	85	185.84	84	170.15	82	41.709	82	18.826
42		10000		F	F	F	F	103	16.034	104	17.746	103	17.588
43		20000		F	F	F	F	132	36.967	F	F	133	37.691
44		200000		8	0.0214	9	0.0226	9	0.0247	9	0.0219	10	0.021
45		500000		13	0.0502	13	0.0556	12	0.0591	12	0.0537	14	0.0539
46		1.00E+06		12	0.0959	14	0.1369	13	0.0842	13	0.0823	18	0.1342
47	EXT. FREUD. & ROTH	100000	(-2,,-2)	16	4.0322	16	4.116	16	3.8218	16	3.8763	17	25.715
48	EXT. FREUD. & ROTH	100	(2,,2)	2	0.0046	2	0.0046	2	0.0046	2	0.0045	2	0.0047
49		200		2	0.008	2	0.0079	2	0.0091	2	0.0088	2	0.0093
50		500		2	0.0141	2	0.0151	2	0.0188	2	0.0147	2	0.0148
51		1000		2	0.0225	2	0.0234	2	0.0215	2	0.0211	2	0.0219
52		2000		2	0.0344	2	0.039	2	0.0339	2	0.0337	2	0.0386
53		5000		2	0.1557	2	0.1685	2	0.1535	2	0.1427	2	0.1396
54		10000		2	0.2457	2	0.2191	2	0.2249	2	0.2244	2	0.249
55		20000		2	0.2777	2	0.3338	2	0.375	2	0.3219	2	0.4121
56		50000		2	0.6799	2	0.9597	2	0.5487	2	0.5274	2	0.7138
57		100000		2	1.6453	2	2.0904	2	1.7307	2	1.7616	2	0.9995
58		200000		2	1.7656	2	1.7916	2	2.5342	2	2.2577	2	4.0865
59		500000		2	8.5502	2	99.107	2	20.060	2	32.074	2	5.1211
60		1.00E+06		2	17.583	2	11.338	2	27.472	2	17.541	2	61.437
61	EXTENDED PENELTY	100	(1,2,3,)	8	0.165	9	0.0187	9	0.0862	9	0.0205	10	0.0204
62		200		13	0.0479	13	0.0592	12	0.0614	12	0.0459	14	0.0494
63		500		12	0.0826	14	0.0839	13	0.0853	13	0.081	18	0.1459
64		1000		23	0.2081	21	0.2594	21	0.2669	21	0.2089	20	0.1921
65		2000		13	0.2305	12	0.2671	13	0.3756	13	0.3693	15	0.441
66		5000		49	4.4818	49	3.0709	F	F	F	F	49	7.2762
67		10000		F	F	63	21.600	63	16.9713	63	16.582	F	F

S/N	Test Functions	DIMM	IN. PT.	FRPRPCC		CCOMB		HHD		ECCHD		HHSFR	
				NI	CPUT	NI	CPUT	NI	CPUT	NI	CPUT	NI	CPUT
68	EXTENDED POWEL 1	100	(0,,0)	3	0.0293	3	0.0068	3	0.0055	3	0.0054	3	0.0058
69		200		3	0.01	3	0.0094	3	0.0092	3	0.0089	3	0.0095
70		500		3	0.0158	3	0.0183	3	0.0148	3	0.0062	3	0.0152
71		1000		3	0.0928	3	0.0522	3	0.0333	3	0.0226	3	0.0244
72		2000		3	0.0989	3	0.0425	3	0.0574	3	0.0789	3	0.1917
73		5000		3	0.4774	3	1.0642	3	0.768	3	0.6637	3	2.4394
74		10000		3	0.3591	3	0.3431	3	0.2404	3	0.2315	3	0.5248
75		20000		3	0.4544	3	0.4888	3	0.6899	3	0.5718	3	1.0432
76		50000		3	3.0521	3	0.8877	3	0.9623	3	0.9261	3	1.0112
77		100000		3	2.0481	F	F	3	1.9546	3	1.929	3	2.0928
78	DIXON & PRICE	100	(-1,,-1)	218	0.2595	227	0.2255	206	0.2639	207	0.1909	213	0.2479
79		200		350	0.4738	386	0.5014	400	0.4614	401	0.4803	398	0.4425
80	SUM OF SQUARES	100	(5,,5)	60	0.0509	60	0.0466	60	0.0458	60	0.0458	60	0.0477
81		200		87	0.1086	87	0.1155	87	0.1098	87	0.1094	87	0.1135
82		500		140	0.3038	140	0.2557	140	0.3186	140	0.2721	140	0.32
83		1000		200	0.4705	200	0.4405	200	0.448	200	0.4448	200	0.512
84		2000		284	0.8714	284	0.759	284	0.8685	284	0.8443	284	0.9415
85		5000		453	9.6165	453	7.575	453	2.3703	453	5.540	453	9.5885
86		10000		645	21.346	645	22.94	645	23.875	645	21.979	645	21.650
87		20000		916	50.923	916	56.85	916	49.003	916	48.749	916	54.732
88		50000		1457	173.56	1457	250.1	1457	98.808	1457	72.764	1457	183.77
89		100000		2070	573.19	2070	5E+03	F	F	F	F	F	F
90	EXTENDED BEALE	100	(1.8,,1.8)	7	0.0222	7	0.0213	7	0.0216	7	0.0211	7	0.0223
91		200		7	0.0385	7	0.0387	7	0.0387	7	0.0377	7	0.039
92		500		7	0.0741	7	0.0756	7	0.0736	7	0.0753	7	0.0791
93		1000		7	0.1253	7	0.1452	7	0.1263	7	0.1255	7	0.1704
94		2000		7	0.2344	7	0.2437	7	0.2147	7	0.2136	7	0.27
95		5000		7	0.3835	7	0.5829	7	0.5549	7	0.5212	7	0.554
96		10000		7	0.8223	7	0.8216	7	0.8302	7	0.7445	7	0.925
97		20000		7	1.3135	7	1.3129	7	1.3155	7	1.2607	7	1.2767
98		50000		7	2.6412	7	2.5963	7	3.5127	7	4.4183	7	1.2767
99		100000		7	6.2606	7	4.2528	7	9.8756	7	5.8942	7	10.810
100		200000		7	24.487	7	54.752	7	82.367	7	59.214	7	57.810
101		500000		7	150.30	7	62.831	7	205.46	7	117.52	7	339.63
102		1.00E+06		F	F	F	F	F	F	F	F	F	F
103	RAYDAN 1	100	(1,,1)	69	2.7277	65	1.6133	64	1.4692	64	1.4136	65	1.5453
104	EXT. TRIDIAGONAL 1	100	(2,,2)	5	0.8581	5	0.3836	5	0.4004	5	0.2252	5	0.3261
105		200		5	0.8136	5	0.5028	5	0.5798	5	0.4355	5	5
106		500		5	1.2614	5	1.464	5	1.5136	5	1.4393	5	1.3706
107		1000		5	1.9037	5	1.9558	5	3.6689	5	1.8612	5	1.9089
108		2000		5	2.774	5	2.9164	5	3.6039	5	3.0007	5	3.5958
109		5000		5	9.1082	5	6.9239	5	8.4434	5	6.2449	5	6.6989
110		10000		5	12.495	5	13.488	5	13.259	5	12.374	5	12.636
111		20000		5	28.427	5	26.162	5	24.781	5	24.489	5	25.570
112		50000		5	58.945	5	60.457	5	60.175	5	59.881	5	59.496
113		100000		6	9.0109	6	12.308	5	10.120	5	8.9121	5	10.673
114		200000		6	21.602	6	25.107	5	25.013	6	20.775	5	25.556
115		500000		6	57.716	6	59.530	5	55.896	5	54.737	5	55.331
116		1.00E+06		5	114.44	6	115.60	6	117.17	5	116.12	5	120.38
117	EXT. HIMMELBLAU	100	(1,,1)	4	0.0374	4	0.0103	4	0.0087	4	0.0085	4	0.0085
118		200		4	0.0138	4	0.0144	4	0.0138	4	0.0136	4	0.0141
119		500		4	0.0238	4	0.0225	4	0.0216	4	0.0106	4	0.0225
120		1000		4	0.0324	4	0.0322	4	0.0312	4	0.0309	4	0.034
121		2000		4	0.0541	4	0.0556	4	0.0537	4	0.0512	4	0.0554
122		5000		4	0.19	4	0.2137	4	0.2265	4	0.1971	4	0.1569
123		10000		4	0.3308	4	0.3487	4	0.3189	4	0.2421	4	0.313
124		20000		4	0.4021	4	0.4551	4	0.4623	4	0.4328	4	0.3955
125		50000		4	0.7933	4	0.7276	4	0.7254	4	0.7223	4	0.7793
126		100000		4	2.4102	4	1.3898	4	2.4247	4	1.3318	4	2.7941
127		200000		4	4.8897	4	5.1342	4	3.3299	4	2.6184	4	4.937
128		500000		4	12.595	4	10.873	4	11.279	4	11.333	4	12.120
129		1.00E+06		4	24.992	4	21.040	4	18.88	4	14.343	4	19.981

S/N	Test Functions	DIMM	IN. PT.	FRPRPCC		CCOMB		HHD		ECCHD		HHSFR	
				NI	CPUT	NI	CPUT	NI	CPUT	NI	CPUT	NI	CPUT
130	FLETCHER FUNCTION	100	(0,,0)	355	0.4989	358	0.4435	352	0.3941	352	0.2947	352	0.3829
131		200		699	0.9529	701	0.9919	660	0.9876	659	0.9069	658	1.0044
132		500		1245	2.1097	1248	2.2616	1325	2.3926	1325	2.3075	1331	2.663
133		1000		1743	9.9177	1645	8.4317	2563	9.5703	2310	9.4725	2562	10.565
134		2000		3898	35.816	6346	51.358	3443	30.686	3541	31.874	3499	34.039
135		5000		8177	268.52	8124	284.94	7993	251.94	7964	248.01	7970	287.19
136	SHALLO FUNCTION	100	(-2,,-2)	5	0.0106	6	0.0107	5	0.0099	5	0.0094	5	0.01
137		200		5	0.0159	6	0.0222	5	0.0167	5	0.0166	5	0.0176
138		500		5	0.0262	6	0.0278	5	0.0299	5	0.0264	5	0.0272
139		1000		5	0.0461	6	0.0414	5	0.0404	5	0.0366	5	0.0377
140		2000		5	0.0962	6	0.0747	5	0.0773	5	0.0616	5	0.0807
141		5000		5	0.248	6	0.3307	5	0.2422	5	0.1963	5	0.2837
142		10000		5	0.292	6	0.3883	5	0.2748	5	0.2382	5	0.4323
143		20000		5	0.6562	6	0.6936	5	0.6411	5	0.7072	5	1.008
144		50000		5	2.5096	6	2.8122	5	2.6169	5	2.6089	5	3.4761
145		100000		5	7.7083	6	10.004	5	10.064	5	8.9219	5	15.646
146		200000		5	32.578	6	34.305	5	5.6922	5	5.5089	5	5.6798
147		500000		5	13.576	6	13.440	5	11.464	5	10.782	5	14.251
148		1.00E+06		5	29.063	6	28.882	5	25.746	5	25.527	5	28.749
149	EXTENDED POWEL	100	(-1,,-1)	33	0.0222	41	0.0402	31	0.0748	30	0.0732	39	0.0802
150		200		33	0.1309	41	0.1885	31	0.1428	30	0.1332	39	0.1904
151		500		33	0.2586	41	0.3131	31	0.2971	30	0.2465	39	0.2976
152		1000		33	0.3954	41	0.5249	31	0.3969	30	0.3614	39	0.3803
153		2000		33	0.5737	47	0.8672	31	0.5486	30	0.524	39	0.5577
154		5000		34	1.3223	47	1.7564	35	1.3843	32	1.3526	39	1.2876
155		10000		34	2.0873	47	2.7708	35	2.2914	32	1.9508	39	2.3024
156		20000		36	4.1631	47	5.961	35	3.9255	32	3.9024	39	4.3003
157		50000		36	19.711	47	22.247	35	14.095	32	14.306	44	14.844
158		100000		36	31.177	48	47.318	35	34.709	32	34.861	44	39.519
159		200000		36	47.429	48	95.609	35	68.743	32	76.641	44	100.12
160		500000		36	222.60	48	233.17	35	209.92	32	226.72	45	310.02
161		1.00E+06		36	591.66	F	F	F	F	F	F	F	F
162	G. TRIDIAGONAL 1	100	(2,,2)	21	0.027	21	0.0264	20	0.026	21	0.024	20	0.0299
163		200		21	0.1458	F	F	F	F	21	0.0291	F	F
164	G. TRIDIAGONAL 2	100	(10,,10)	56	0.0607	56	0.0719	54	0.0586	54	0.0544	55	0.0781
165		200		54	0.1165	55	0.1163	54	0.1159	55	0.113	F	F
166		500		F	F	49	0.5032	50	0.4987	50	0.4454	F	F
167		1000		55	0.9442	F	F	51	1.3797	51	1.3347	F	F
168	DIAGONAL 4	100	(1,,1)	1	0.0036	1	0.0036	1	0.0097	1	0.0058	1	0.0073
169		200		1	0.0141	1	0.0077	1	0.0069	1	0.0067	1	0.0074
170		500		1	0.0105	1	0.011	1	0.0111	1	0.0109	1	0.0114
171		1000		1	0.0175	1	0.0159	1	0.0148	1	0.0156	1	0.0172
172		2000		1	0.0271	1	0.0257	1	0.0271	1	0.0256	1	0.0216
173		5000		1	0.0828	1	0.0828	1	0.0636	1	0.0567	1	0.0762
174		10000		1	0.1946	1	0.0808	1	0.0681	1	0.0405	1	0.0841
175		20000		1	0.0946	1	0.1466	1	0.1018	1	0.0925	1	0.1758
176		50000		1	0.2899	1	0.281	1	0.3045	1	0.2361	1	0.3367
177		100000		1	0.3641	1	0.4007	1	0.3987	1	0.2325	1	0.3723
178		200000		1	0.929	1	1.0057	1	0.9599	1	0.9076	1	1.0822
179		500000		1	12.761	1	46.662	1	2.6979	1	2.6056	1	2.9852
180		1.00E+06		1	5.8091	1	6.1905	1	7.5586	1	4.7769	1	3.3464
181	NONSCOMP FUNCTION	100	(3,,3)	33	0.0595	33	0.0801	33	0.0574	33	0.0562	33	0.0764
182	NONSCOMP FUNCTION	200	(-5,,-5)	33	0.1791	33	0.4742	34	0.2061	34	0.2068	33	0.1468
183		500		35	0.339	35	0.4512	36	0.4168	36	0.4625	35	0.4544
184		1000		37	0.6032	37	0.781	37	0.6056	37	0.5484	37	0.7639
185		2000		38	0.9586	40	1.328	39	0.7333	39	0.6687	38	0.6873
186		5000		37	1.5229	39	1.373	37	1.0129	37	0.8368	37	1.2256
187		10000		34	102.89	35	31.057	33	8.3701	33	3.9223	32	3.4828
188		20000		35	6.6459	35	9.0979	35	15.524	35	6.1521	35	5.5417
189		50000		36	17.596	40	28.232	37	23.197	37	2.8626	36	42.864
190		100000		33	24.274	38	81.169	37	5.6929	37	6.4839	37	6.6469
191		200000		35	42.713	35	55.033	35	27.210	35	25.871	37	26.598
192		500000		35	77.060	36	360.34	34	128.56	34	66.271	37	55.823
193		1.00E+06		34	180.35	34	323.52	37	341.26	37	340.56	37	78.408

S/N	Test Functions	DIMM	IN. PT.	FRPRPCC		CCOMB		HHD		ECCHD		HHSFR	
				NI	CPUT	NI	CPUT	NI	CPUT	NI	CPUT	NI	CPUT
194	QUADRATIC QFN 2	100	(.5,,.5)	100	0.1088	98	0.1117	98	0.0889	98	0.0861	99	0.0981
195		200		151	0.2198	146	0.2806	151	0.2202	151	0.2046	152	0.2567
196		500		250	0.4401	242	0.4418	250	0.4663	250	0.4456	246	0.4104
197		1000		358	0.805	349	0.8064	359	0.8233	359	0.8218	360	0.7704
198		2000		512	1.3849	498	1.4736	512	1.1935	512	1.4744	513	1.5794
199	EXTENDED DENSCHNB	100	(1,,1)	3	0.1687	3	0.1451	3	0.096	3	0.0954	3	0.1071
200		200		3	0.1712	3	0.3304	3	0.1696	3	0.1661	3	0.1373
201		500		3	0.2304	3	0.016	3	0.0148	3	0.0146	3	0.0151
202		1000		3	0.0214	3	0.0225	3	0.0214	3	0.0214	3	0.0219
203		2000		3	0.0442	3	0.0389	3	0.0387	3	0.0359	3	0.0369
204		5000		3	0.1913	3	0.1018	3	0.1608	3	0.1566	3	0.1055
205		10000		3	0.2588	3	0.2779	3	0.256	3	0.2017	3	0.2694
206		20000		3	0.3859	3	0.7056	3	1.4985	3	0.7014	3	0.7708
207		50000		3	1.9576	3	1.9885	3	1.8556	3	1.4079	3	1.5602
208		100000		4	3.3985	4	3.6695	3	2.2708	3	2.594	3	2.6158
209		200000		4	17.961	4	7.1017	4	5.9478	4	4.9275	4	7.4591
210		500000		4	6.1093	4	10.417	4	16.734	4	14.587	4	143.30
211		1.00E+06		4	11.560	4	15.908	4	12.856	4	11.642	4	16.519
212	EXT. QUADRATIC P1	100	(1,...,1)	3	0.0381	3	0.0122	3	0.01	3	0.009	3	0.0125
213		200		F	F	F	F	5	0.0173	5	0.0169	5	0.0172
214		2000		5	0.0697	F	F	5	0.0692	5	0.0687	5	0.0718
215		5000		F	F	5	0.2618	5	0.2882	5	0.2733	F	F
216		10000		5	0.4959	F	F	5	0.4252	5	0.3421	F	F
217	HAGER	100	(1,,1)	22	0.0308	23	0.0347	24	0.0291	24	0.0269	24	0.0355
218	GENERALIZED QUARTIC	100	(-2,,-2)	1	0.0772	1	0.0451	1	0.0419	1	0.0352	1	0.0356
219		200		1	0.0553	1	0.0592	1	0.053	1	0.0523	2	0.0723
220		500		1	0.0726	1	0.0666	1	0.0581	1	0.0574	1	0.0716
221		1000		2	0.1284	1	0.0971	1	0.1701	1	0.1195	2	0.1202
222		2000		1	0.2376	1	0.1968	1	0.2384	1	0.1804	2	0.2543
223		5000		1	0.5933	1	0.4688	1	0.4442	1	0.3654	2	0.5195
224		10000		1	1.0798	1	1.2139	1	1.6326	1	1.582	2	2.7674
225		20000		2	4.2819	2	5.238	2	1.6326	2	5.0241	2	7.6603
226		50000		2	19.759	2	25.267	2	36.501	2	21.286	2	36.371
227		100000		F	F	1	24.635	1	24.686	1	24.432	1	26.214
228		200000		1	52.473	1	402.19	1	12.533	1	10.689	1	14.771
229		500000		2	364.93	2	144.55	2	177.63	2	58.728	2	40.648
230		1.00E+06		2	109.78	2	87.413	2	101.32	2	118.92	2	153.96

5. Application of ECCHD method on 3DOF robotic motion control model

In this subsection, we illustrate additional implementation of Algorithm 1 in solving three degrees of freedom (3DOF) real-time robotic model as suggested in [51]. Briefly, we describe the three-joints of the discrete-time kinematics model at the position level of a planar robot manipulator by

$$f(\theta_k) = \eta_k. \tag{5.1}$$

The relation given by (5.1), implies that the function $f(\cdot)$ is the kinematics mapping, which relate the orientation of any part of the robot is given by the following model:

$$f(\theta) = \begin{bmatrix} b_1\cos(\theta_1) + b_2\cos(\theta_1+\theta_2) + b_3\cos(\theta_1+\theta_2+\theta_3) \\ b_1\sin(\theta_1) + b_2\sin(\theta_1+\theta_2) + b_3\sin(\theta_1+\theta_2+\theta_3) \end{bmatrix}, \tag{5.2}$$

where, the length of $i^{th}rod$, is denoted by b_i (for $i = 1, 2, 3$) and $\theta \in \mathbb{R}^3$ of $f(\theta)$, is the vector that indicate the end effector position. Let $\eta_k \in \mathbb{R}^2$ be the vector that indicates the required path at time, t_k. In modeling a motion control robot, at time interval say, $t_k \in [0, t_f]$ a series of nonlinear least square problems are generated, which are formulated in form of problem (1.1) as:

$$\min_{\theta \in \mathbb{R}^3} \frac{1}{2}\|f(\theta) - \eta_k\|^2, \tag{5.3}$$

where η_k represents the end effector-controlled path at t_k of a required curve (Lissajous), which is expressed by [52] as:

$$\eta_k = \begin{bmatrix} 1.5 + 0.4\sin(\frac{\pi t_k}{5}) \\ \\ \frac{\sqrt{3}}{2} + 0.4\sin(\frac{\pi t_k}{5} + \frac{\pi}{3}) \end{bmatrix}. \tag{5.4}$$

The code and implementation of the Algorithm 1 on (5.1)-(5.4) was performed using MATLAB R2022a 11^{th} Gen. Intel(R) Core i7-1195G7 and run on a PC with RAM 16 GB that is has CPU of 2.90GHZ. The joint was initialized at time instant, $t = 0$, and position vector to be $\theta_0 = [\theta_1, \theta_2, \theta_3] = [0, \frac{\pi}{3}, \frac{\pi}{2}]$, with the task period $[0, t_f inal]$ that is divided into 200 parts equally, where length of the rod is, $b_i = 1$ (for $i = 1, 2, 3$) and $t_f inal = 10$ seconds. The report of motion control experiment of Algorithm 1 are plotted in Figures 3–6. Clearly, results of the figures show that the ECCHD method synthesized the robot trajectories and pass through the desired path as shown in Figures 5–6 with residuals error of 10^{-8} as observed from Figures 3–4.

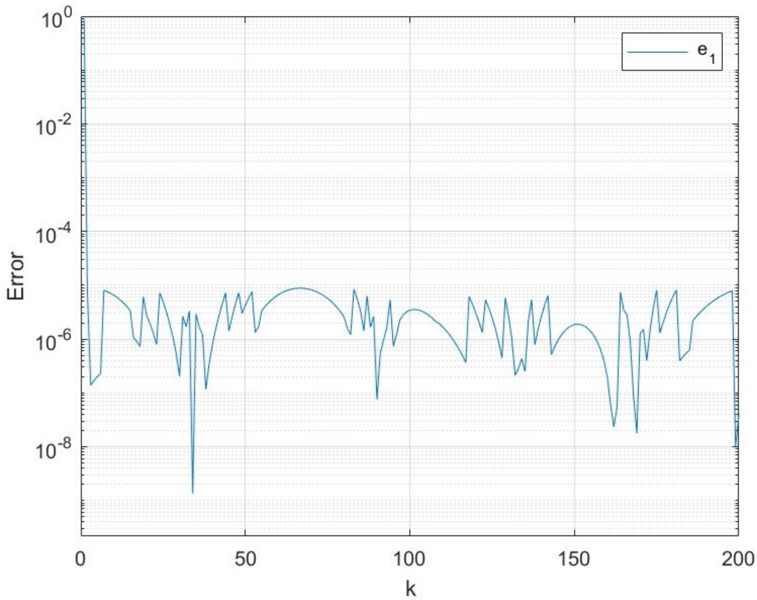

FIGURE 3. Error tracking by ECCHD on x–axis.

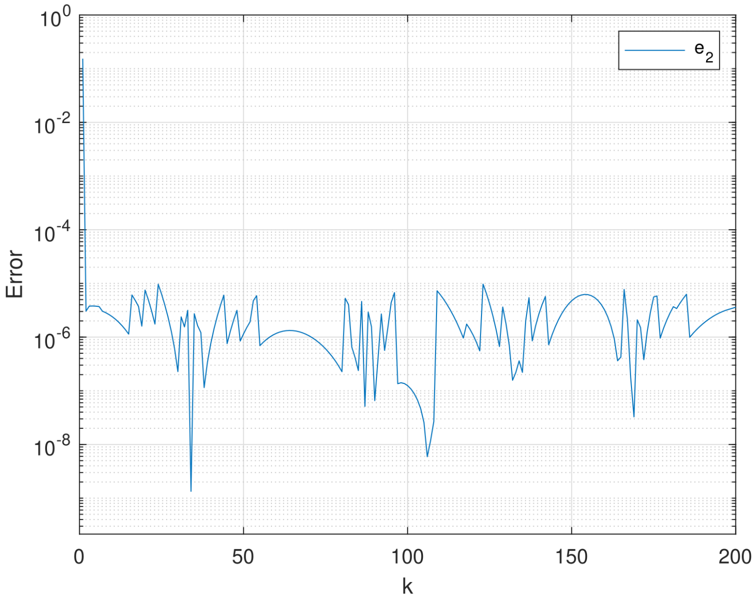

FIGURE 4. Error tracking by ECCHD on y–axis.

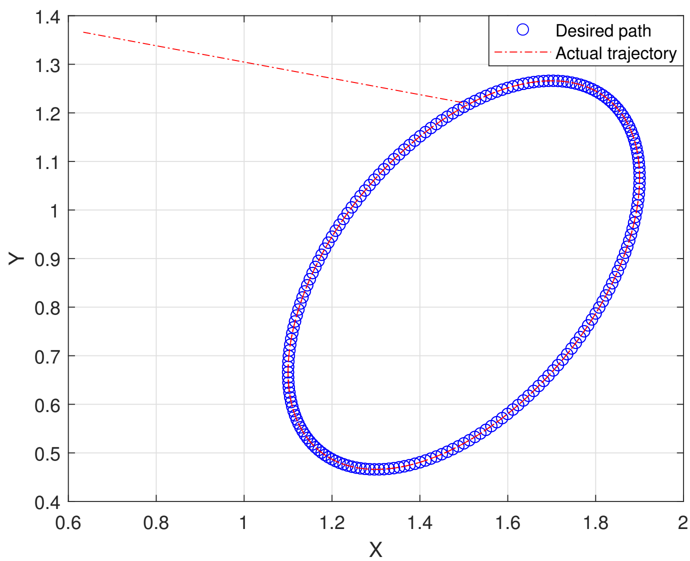

FIGURE 5. End effector of the ECCHD trajectory of desired path.

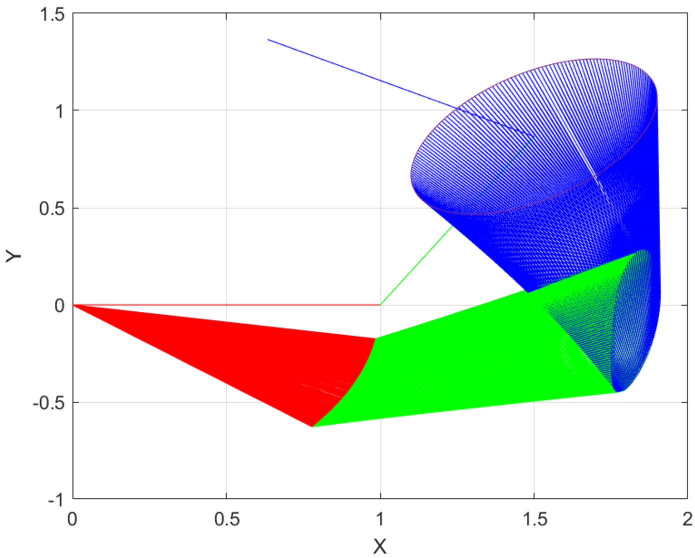

FIGURE 6. Robot trajectories synthesized by ECCHD.

6. CONCLUSION

In this paper, we have presented a hybrid CG method from the extended conjugacy condition. The method uses the choice of the modulating parameter t that incorporate the classical HS and DY updates in such away that, we generalize DL-type parameter, so that if $t = 1$, then its scale down to a method that uses the secant equation. Theoretical and numerical computations adopt inexact line search. The results of the comparison with some known CG coefficients show the algorithm is robust, efficient and converge globally using strong Wolfe condition. The Computational experiments indicated that the DL algorithms are robust and more efficient than some well-known CG methods. Despite the fact that several optimal choices for DL parameter were proposed, the best choice of the parameter t still remains subject of consideration. The proposed method is also applied to solve three degrees of freedom (3DOF) real-time motion control model.

REFERENCES

[1] Y. Dai, Y. Yuan, An efficient hybrid conjugate gradient method for unconstrained optimization. Annals of Operations Research, 103(1)(2001) 33–47.
[2] H.A. Sani, W.M. Yusuf, A transformed double step length method for solving large-scale systems of nonlinear equations. Journal of Numerical Mathematics and Stochastics, 9(1)(2017) 20–23.
[3] A.M. Awwal, I.M. Sulaiman, M. Maulana, M. Mustafa, P. Kumam, K. Sitthithakerngkiet, A spectral RMIL+ conjugate gradient method for unconstrained optimization with applications in portfolio selection and motion control. IEEE Access, 9(2021) 75398–75414.

[4] H.A. Sani, M. Arunava, W.M. Yusuf, A. Kabiru, A.M. Aliyu, Motion control of the two joint planar robotic manipulators through accelerated Dai–Liao method for solving system of nonlinear equations. Engineering Computations, 39(5)(2022) 1802–1840.

[5] J. Liu, S. Du, Modified three-term conjugate gradient method and its applications. Mathematical Problems in Engineering, 2019(2019).

[6] W. Jirakitpuwapat, P. Kumam, Y.J. Cho, K. Sitthithakerngkiet, A general algorithm for the split common fixed point problem with its applications to signal processing. Mathematics, 7(3)(2019) 226.

[7] G. Yuan, J. Lu, Z. Wang, The PRP conjugate gradient algorithm with a modified WWP line search and its application in the image restoration problems. Applied Numerical Mathematics, 152(2020) 1–11.

[8] J. Abubakar, P. Kumam, A.I. Garba, A.H. Ibrahim, W. Jirakitpuwapat, Hybrid Iterative Scheme for Variational Inequality Problem Involving Pseudo-monotone Operator with Application in Signal Recovery. Bulletin of the Iranian Mathematical Society, 48(6)(2022) 2995–3017.

[9] A.S. Halilu, A. Majumder, M.Y. Waziri, K. Ahmed, Signal recovery with convex constrained nonlinear monotone equations through conjugate gradient hybrid approach. Mathematics and Computers in Simulation, 187(2021) 520–539.

[10] A.H. Ibrahim, P. Kumam, A.B. Abubakar, W. Jirakitpuwapat, J. Abubakar, A hybrid conjugate gradient algorithm for constrained monotone equations with application in compressive sensing. Heliyon, 6(3)(2020) 1–15.

[11] M.M. Yahaya, P. Kumam, A.M. Awwal, P. Chaipunya, S. Aji, S. Salisu, A new generalized quasi-newton algorithm based on structured diagonal hessian approximation for solving nonlinear least-squares problems with application to 3dof planar robot arm manipulator. IEEE Access, 10(2022) 10816–10826.

[12] N. Salihu, P. Kumam, A.M. Awwal, I.M. Sulaiman, T. Seangwattana, The global convergence of spectral RMIL conjugate gradient method for unconstrained optimization with applications to robotic model and image recovery. Plos one, 18(3)(2023) e0281250.

[13] N. Salihu, P. Kumam, A.M. Awwal, I. Arzuka, T. Seangwattana, A Structured Fletcher-Revees Spectral Conjugate Gradient Method for Unconstrained Optimization with Application in Robotic Model. Operations Research Forum, 4(4)(2023) 81.

[14] Y.H. Dai, C.X. Kou, A nonlinear conjugate gradient algorithm with an optimal property and an improved wolfe line search. SIAM Journal on Optimization, 23(1)(2013) 296–320.

[15] W.W. Hager, H. Zhang, A survey of nonlinear conjugate gradient methods. Pacific journal of Optimization, 2(1)(2006) 35–58.

[16] R. Fletcher, C.M. Reeves, Function minimization by conjugate gradients. The computer journal, 7(2)(1964) 149–154.

[17] Y.H. Dai, Y. Yuan, A nonlinear conjugate gradient method with a strong global convergence property. SIAM Journal on optimization, 10(1)(1999) 177–182.

[18] R. Fletcher, Practical methods of optimization. A Wiley Interscience Publication, 1987.

[19] M.R. Hestenes, E. Stiefel, Methods of conjugate gradients for solving. Journal of research of the National Bureau of Standards, 49(6)(1952) 409.

[20] B.T. Polyak. A general method for solving extremal problems. Dokl. Akad. Nauk SSSR, 174(1)(1967) 33–36.
[21] E. Polak, G. Ribiere. Note sur la convergence de methodes de directions conjuguees. USSR Computational Mathematics and Mathematical Physics, 9(4)(1969) 94–112.
[22] Y. Liu, C. Storey, Efficient generalized conjugate gradient algorithms, part 1: theory. Journal of optimization theory and applications, 69(1)(1991) 129–137.
[23] J. Liu, S.J. Li. New hybrid conjugate gradient method for unconstrained optimization. Applied Mathematics and Computation, 245(2014) 36–43.
[24] S. Babaie-Kafaki, R. Ghanbari, Two hybrid nonlinear conjugate gradient methods based on a modified secant equation. Optimization, 63(7)(2014) 1027–1042.
[25] N. Andrei, Another hybrid conjugate gradient algorithm for unconstrained optimization. Numerical Algorithms, 47(2)(2008) 143–156.
[26] M. Abdullahi, A.S. Halilu, A.M. Awwal, N. Pakkaranang, On efficient matrix-free method via quasi-Newton approach for solving system of nonlinear equations. Advances in the Theory of Nonlinear Analysis and its Application, 5(4)(2021) 568–579.
[27] N. Andrei, Hybrid conjugate gradient algorithm for unconstrained optimization. Journal of Optimization Theory and Applications, 141(2)(2009) 249–264.
[28] A. Perry, A modified conjugate gradient algorithm. Operations Research, 26(6)(1978) 1073–1078.
[29] Y.H. Dai, L.Z. Liao, New conjugacy conditions and related nonlinear conjugate gradient methods. Applied Mathematics and Optimization, 43(1)(2001) 87–101.
[30] W. Sun, Y.X. Yuan, Optimization theory and methods: nonlinear programming, volume 1. Springer Science & Business Media, 2006.
[31] S. Babaie-Kafaki. On optimality of the parameters of self-scaling memoryless quasi-newton updating formulae. Journal of Optimization Theory and Applications, 167(1)(2015) 91–101.
[32] W.W. Hager, H. Zhang. Algorithm 851: Cg descent, a conjugate gradient method with guaranteed descent. ACM Trans. Math. Softw., 32(1)(2006) 113–137.
[33] M.R. Arazm, S. Babaie-Kafaki, R. Ghanbari, An extended Dai-Liao conjugate gradient method with global convergence for nonconvex functions. Glasnik matematički, 52(2)(2017) 361–375.
[34] Y. Narushima, H. Yabe, Conjugate gradient methods based on secant conditions that generate descent search directions for unconstrained optimization. Journal of Computational and Applied Mathematics, 236(17)(2012) 4303–4317.
[35] S. Babaie-Kafaki, R. Ghanbari, Two optimal Dai–Liao conjugate gradient methods. Optimization, 64(11)(2015) 2277–2287.
[36] N. Andrei, An adaptive scaled BFGS method for unconstrained optimization. Numerical Algorithms, 77(2)(2018) 413–432.
[37] N. Andrei, A Dai-Liao conjugate gradient algorithm with clustering of eigenvalues. Numerical Algorithms, 77(4)(2018) 1273–1282.
[38] S. Babaie-Kafaki, R. Ghanbari, A descent family of Dai–Liao conjugate gradient methods. Optimization Methods and Software, 29(3)(2014) 583–591.
[39] N. Salihu, M.R. Odekunle, A.M. Saleh, S. Salihu, A Dai-Liao hybrid Hestenes-Stiefel and Fletcher-Revees methods for unconstrained optimization. International Journal of Industrial Optimization, 2(1)(2021) 33–50.
[40] N. Salihu, M.R. Odekunle, M.Y. Waziri, A.S. Halilu, S. Salihu, A Dai-Liao hybrid conjugate gradient method for unconstrained optimization. International Journal of

Industrial Optimization, 2(2)(2021) 69–84.
[41] S. Babaie-Kafaki. A survey on the dailiao family of nonlinear conjugate gradient methods. RAIRO-Oper. Res., 57(1)(2023) 43–58.
[42] S. Babaie-Kafaki, R. Ghanbari, The Dai–Liao nonlinear conjugate gradient method with optimal parameter choices. European Journal of Operational Research, 234(3)(2014) 625–630.
[43] N. Andrei, A hybrid conjugate gradient algorithm for unconstrained optimization as a convex combination of Hestenes-Stiefel and Dai-Yuan. Studies in Informatics and Control, 17(1)(2008) 57.
[44] M.J.D. Powell, Nonconvex minimization calculations and the conjugate gradient method. In Numerical analysis, pages 122–141. Springer, 1984.
[45] G. Zoutendijk, Nonlinear programming, computational methods. In: J. Abadie Ed., Integer and Nonlinear Programming, North-Holland, Amsterdam, pages 37–86, 1970.
[46] N. Andrei, Nonlinear Conjugate Gradient Methods for Unconstrained Optimization. Springer Cham, 2020.
[47] J. Momin, Y. Xin-She, A literature survey of benchmark functions for global optimization problems. Int. Journal of Mathematical Modelling and Numerical Optimisation, 4(2)(2013) 150–194.
[48] S.S. Djordjevic, New hybrid conjugate gradient method as a convex combination of HS and FR conjugate gradient methods. Journal of Applied Mathematics and Computation, 2(9)(2018) 366–378.
[49] S.S. Djordjevic, New hybrid conjugate gradient method as a convex combination of FR and PRP methods. Filomat, 30(11)(2016) 3083–3100.
[50] E.D. Dolan, J.J. Moré, Benchmarking optimization software with performance profiles. Mathematical programming, 91(2)(2002) 201–213.
[51] A. Renfrew, Introduction to robotics: Mechanics and control. International Journal of Electrical Engineering and Education, 41(4)(2004) 388.
[52] Y. Zhang, L. He, C. Hu, J. Guo, J. Li, Y. Shi. General four-step discrete-time zeroing and derivative dynamics applied to time-varying nonlinear optimization. Journal of Computational and Applied Mathematics, 347(2019) 314–329.