Filomat 39:15 (2025), 5071–5084
https://doi.org/10.2298/FIL2515071A

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

A new descent extension of DY conjugate gradient method based on DL approach

Semiu Akinpelu Ayinde[a,*], Idowu Ademola Osinuga[b], Adesina Kamorudeen Adio[a]

[a] *Department of Basic Sciences , Babcock University, Ilishan-Remo, Ogun State, Nigeria*
[b] *Department of Mathematics, Federal University of Agriculture, Abeokuta, Ogun State, Nigeria*

Abstract. Optimization problems are ways by which decisions are being made and these are found in all sectors of human endeavors. Conjugate gradient methods have been widely used as a result of their efficiency and low memory status in solving large-scale optimization problems. This article introduces an inexact line search scheme based algorithm. Analyses done showed that it satisfied decent property and converges globally. It also outperformed other methods considered when subjected to numerical test under Dolan and More performance profile tools..

1. Introduction

The aim of this paper is to solve an optimization problem

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

where $f : R^n \longrightarrow R$ is a smooth function and g_k is the gradient of f at x_k. Conjugate gradient methods are the appropriate tools for solving (1) with the following methods of iteration.

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

where $\alpha_k > 0$ is a step-length determined from a line search with the direction d_k given by

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

The formula $\beta_k \in R$ is known as an update parameter and is used to name and characterize conjugate gradient (CG) methods. The following are the first and second generations of CG methods introduced and are regarded as classical methods.

2020 *Mathematics Subject Classification*. Primary 90C26.
Keywords. Optimization method, inexact line search, gradient method, nonlinear function, update parameter.
Received: 06 May 2022; Revised: 15 September 2024; Accepted: 30 March 2025
Communicated by Predrag Stanimirović
* Corresponding author: Semiu Akinpelu Ayinde
Email addresses: ayindes@babcock.edu.ng (Semiu Akinpelu Ayinde), osinuga09@gmail.com (Idowu Ademola Osinuga), adiok@babcock.edu.ng (Adesina Kamorudeen Adio)
ORCID iDs: https://orcid.org/0000-0003-4917-4709 (Semiu Akinpelu Ayinde), https://orcid.org/0000-0003-1390-7069 (Idowu Ademola Osinuga), https://orcid.org/0000-0002-7330-4852 (Adesina Kamorudeen Adio)

$$\begin{cases} \beta_k^{HS} = \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}}, \\ \beta_k^{FR} = \frac{\|g_k\|^2}{\|g_{k-1}\|^2}, \\ \beta_k^{PRP} = \frac{g_k^T y_{k-1}}{\|g_{k-1}\|^2}, \\ \beta_k^{LS} = \frac{-g_k^T y_{k-1}}{d_{k-1}^T g_{k-1}}, \\ \beta_k^{DY} = \frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}}, \\ \beta_k^{CD} = \frac{-\|g_k+1\|^2}{d_k^T g_k}, \end{cases} \tag{4}$$

where $y_{k-1} = g_k - g_{k-1}$ and $\|.\|$ is the euclidean norm. The above corresponding methods HS, FR, PRP, LS, DY and CD are Hestenes-Stiefel HS [13], Fletcher-Reeves FR [11], Polak-Rebiere Polyak PRP [17, 18], Liu-Storey LS [16], Dai-Yuan DY [7] and Conjugate Descent CD [10] methods respectively.

1.1 Related work and Motivation

Dai and Liao [6] presented a CG method known as DL method that uses update parameter β_k determined by extending the conjugacy condition

$$d_k^T y_{k-1} = -t g_k^T s_{k-1}. \tag{5}$$

Hence, they defined β_k as

$$\beta_k^{DL} = \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} - \frac{t g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}}. \tag{6}$$

where $t > 0$ and $s_{k-1} = x_k - x_{k-1}$. Modification of (6) to

$$\beta_k^{DL+} = \max\{\frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}}, 0\} - \frac{t g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}}. \tag{7}$$

was done to establish its global convergence for general function. Parameter $t > 0$ determines the numerical performance of DL method. A lot of researchers have proposed different modifications of DL method as a result of its efficiency.

Basically, global convergence of any method depends on the choice of step-length α_k and as a result, α_k must satisfy the following Wolfe conditions

$$f(x_k) - f(x_k + \alpha_k d_k) \geq -\delta \alpha_k g_k^T d_k \tag{8}$$

and

$$g(x_k + \alpha_k d_k)^T d_k \geq \sigma g_k^T d_k \tag{9}$$

for $0 < \delta \leq \sigma < 1$

Moreover, a number of researchers have published papers on the global convergence properties of the classical CG methods. They include Al Baali [2], Liu *et al* [15], Powell [19], Hu and Storey [14], Zoutendijk [21]. Special interest is on the CG method proposed by Dai and Yuan [7] which has global convergence for general functions. According to Dai and Yuan [7], β_k^{CD} and β_k^{FR} methods have global convergence dependent

on certain conditions. As a result of the deficiency found in the global convergence properties of notable CG methods, they came up with a new CG method that has strong global convergence properties. They went by way of investigating CG method that satisfies conditions (8)-(9) and generates descent direction. Let d_k be a descent direction with $d_{k-1}^T g_{k-1} < 0$. An update parameter β_k which defines a descent direction d_k was determined with the requirement that

$$-\|g_k\|^2 + \beta_k g_k^T d_{k-1} < 0. \tag{10}$$

$\beta_k > 0$ was assumed. Defined a parameter

$$\tau = \frac{\|g_k\|^2}{\beta_k}. \tag{11}$$

Hence, (11) was said to be equivalent to

$$\tau_k > g_k^T d_{k-1}. \tag{12}$$

By letting $\tau_k = d_{k-1}^T y_{k-1}$, they proposed the following formula.

$$\beta_k = \frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}}. \tag{13}$$

Their method is globally convergent when standard Wolfe conditions (8) and (9) are satisfied.
CG methods have gained wide acceptability and lately, they have found applications in the areas of machine learning, robotics, statistics etc..Our motivation came from the way DL method was constructed together with its performance and also from the fact that DY method converges globally.
The main purpose of this paper is to give another formula for β_k which can compete well with existing methods. The new method is discussed in section 2. Section 3 analysed convergence and descent properties of our method. In section 4, numerical results of the new method in comparison with other methods and discussion of results are presented. Conclusion of the work is given in section 5.

2. Proposed update parameter β_k

This section discusses the update parameter for the new method. We follow the pattern used in [6]. For an inexact line search, the conjugacy condition (5) given in [6] suffices.
From (3)

$$d_k = -g_k + \beta_k d_{k-1}. \tag{14}$$

Multiply (14) by g_{k-1} to give

$$g_{k-1}^T d_k = -g_{k-1}^T g_k + \beta_k d_{k-1}^T g_{k-1}. \tag{15}$$

Since

$$d_k^T y_{k-1} = d_k^T g_k - d_k^T g_{k-1}. \tag{16}$$

Using (5) and (16), we have

$$g_{k-1}^T d_k = g_k^T d_k + t g_k^T s_{k-1}. \tag{17}$$

By (15) and (17),

$$g_k^T d_k + t g_k^T s_{k-1} = -g_{k-1}^T g_k + \beta_k d_{k-1}^T g_{k-1}. \tag{18}$$

Hence,

$$\beta_k = \frac{d_k^T g_k}{d_{k-1}^T g_{k-1}} + \frac{t g_k^T s_{k-1} + g_{k-1}^T g_k}{d_{k-1}^T g_{k-1}}, \tag{19}$$

for $t > 0$. Hager and Zhang [12] determined the value of t as we have in (6)-(7) for an arbitrary kth and the best approximation in each iteration to be $t_k = \frac{2\|y_{k-1}\|^2}{y_{k-1}^T s_{k-1}}$. On the other hand, Andrei [4] defined t_k as $t_k = \frac{s_{k-1}^T y_{k-1}}{\|s_{k-1}\|^2}$. A good number of notable researchers have worked on the value of t even for its best fixed value. According to Babaie-Kafaki [5], encouraging numerical output were reported by a number of authors following constant setting of t using trial and error scheme. In this work, the value of t takes the same form as was determined in [6]. For the numerical experiment performed here, $t = 0.1$ gives the best result.

The first term in (19) is given by

$$\frac{d_k^T g_k}{g_{k-1}^T d_{k-1}} = \frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}}. \tag{20}$$

Details can be found in (Equations (2.5), (2.6) in [7] and (1.13) in [8]).
Therefore, we propose the following update parameter.

$$\beta_k^{AOA} = \frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}} + \frac{t(g_k^T s_{k-1} + g_{k-1}^T g_k)}{(d_{k-1}^T g_{k-1})}. \tag{21}$$

Parameter t multiplying the whole numerator in the second term of (21) gives best results.
β_k^{AOA} together with procedures (2)-(3) will be called AOA method.

2.1. Algorithm of the new CG method

Step 1: Suppose $x_0 \in R^n$ is the initial point with $d_0 = -g_0$, $k = 0$ and $\epsilon > 0$. If $\|g_k\| \leq \epsilon$, then stop.
Step 2 :Find $\alpha_k > 0$ by using conditions (8)-(9).
Step 3: Determine β_k^{AOA}, then generate $\{d_k\}$, $\{g_k\}$ and $\{x_k\}$.
Step 4: Set $k = k + 1$ and then go to step 2.

We first consider the descent property of the new method.

3. Convergence of the new method

Convergence of the new algorithm is done in this section based on the line search conditions in (8) and (9).

Assumption 3.1
Meanwhile, it is assumed that the objective function (1) satisfies the following conditions.
Define

$$S = \{x | f(x) \leq f(y)\} \tag{22}$$

$y \in R^n$ with S bounded. Given a neighborhood T in S, f is continuously differentiable in T and the gradient is such that there is a constant $B > 0$ where

$$\|g(x) - g(y)\| \leq B\|x - y\| \tag{23}$$

for any $x, y \in T$.
In addition, constants L by Assumptions 3.1 satisfy

$$\|x - y\| \leq L \tag{24}$$

for any $x, y \in T$.

Theorem 3.2 is on descent property of the proposed method.

Theorem 3.2: Suppose d_k and g_k are generated by the conjugate gradient Algorithm 2.1. Then, d_k satisfies

$$g_k^T d_k \leq -c\|g_k\|^2 \tag{25}$$

for each $k \geq 0$ and constant c.

Proof: It is obvious for the case when $k = 0$ and also step 1 of Algorithm 2.1, that

$$d_0^T g_0 = -\|g_0\|^2. \tag{26}$$

So also, pre multiplying (14) by g_k gives

$$d_k^T g_k = -\|g_k\|^2 + \beta_k^{AOA} g_k^T d_{k-1}, \tag{27}$$

when $\beta_k = \beta_k^{AOA}$. Let assume that $\beta_k^{AOA} = 0$, then inner product ensures

$$d_k^T g_k = -\|g_k\|^2 + \beta_k^{AOA} g_k^T d_{k-1} = -\|g_k\|^2 < 0. \tag{28}$$

Hence, we always assume that $\beta_k^{AOA} \neq 0$.
Therefore,

$$d_k^T g_k = -\|g_k\|^2 + \left(\frac{\|g_k\|^2}{y_{k-1}^T d_{k-1}} + \frac{t(g_k^T s_{k-1} + g_{k-1}^T g_k)}{(d_{k-1}^T g_{k-1})}\right) g_k^T d_{k-1}. \tag{29}$$

Consider

$$y_{k-1}^T d_{k-1} = (g_k - g_{k-1})^T d_{k-1} \tag{30}$$

$$= g_k^T d_{k-1} - g_{k-1}^T d_{k-1} \tag{31}$$

$$\leq |g_k^T d_{k-1}| - g_{k-1}^T d_{k-1}. \tag{32}$$

For (32) and $y_{k-1}^T d_{k-1} > 0$ to hold always, then

$$g_{k-1}^T d_{k-1} < 0. \tag{33}$$

See [7] also.
Hence,

$$d_k^T g_k = -\|g_k\|^2 + \left(\frac{\|g_k\|^2}{y_{k-1}^T d_{k-1}} - \frac{t(g_k^T s_{k-1} + g_k - 1^T g_k)}{-(d_{k-1}^T g_{k-1})}\right) g_k^T d_{k-1} \tag{34}$$

with $t > 0$ as stated above. Therefore,

$$d_k^T g_k \leq -\|g_k\|^2 + \beta_k^{DY} g_k^T d_{k-1}. \tag{35}$$

By [7], $c = \frac{1}{1+\sigma}$. Hence,

$$d_k^T g_k \leq -\left(\frac{1}{1+\sigma}\right)\|g_k\|^2. \tag{36}$$

The result follows.
We next consider results on global convergence for the new method.

Lemma 3.3: Given equation (2) with descent direction d_k, let Assumption 3.1 be satisfied where α_k also satisfies the Wolfe conditions in (8)-(9), then

$$\sum_{k=0}^{\infty} \frac{\|g_k\|^4}{\|d_k\|^2} < \infty. \tag{37}$$

The proof of Lemma 3.3 is in [21]

Theorem 3.4: Let Assumption 3.1 hold and also that x_k is determine by the Algorithm 2.1, then

$$\liminf_{k\to\infty} \|g_k\| = 0. \tag{38}$$

Proof: We proof the result by contradiction. This implies that

$$\|g_k\| \geq c_1 \ \ \forall\, k. \tag{39}$$

See [20].

$$\beta_k^{AOA} = \frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}} - \frac{tg_k^T s_{k-1}}{(-g_{k-1}^T d_{k-1})} - \frac{tg_k^T g_{k-1}}{(-g_{k-1}^T d_{k-1})}. \tag{40}$$

By (40),

$$\beta_k^{AOA} \leq \frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}} + \frac{tg_k^T s_{k-1}}{(-g_{k-1}^T d_{k-1})} + \frac{tg_k^T g_{k-1}}{(-g_{k-1}^T d_{k-1})}. \tag{41}$$

From (20),

$$\beta_k^{AOA} \leq \frac{d_k^T g_k}{g_{k-1}^T d_{k-1}} + \frac{tg_k^T s_{k-1}}{(-g_{k-1}^T d_{k-1})} + \frac{tg_k^T g_{k-1}}{(-g_{k-1}^T d_{k-1})}. \tag{42}$$

By (25),

$$\beta_k^{AOA} \leq \frac{-c\|g_k\|^2}{g_{k-1}^T d_{k-1}} + \frac{tg_k^T s_{k-1}}{(-g_{k-1}^T d_{k-1})} + \frac{tg_k^T g_{k-1}}{(-g_{k-1}^T d_{k-1})}. \tag{43}$$

$$\beta_k^{AOA} \leq |\frac{c\|g_k\|^2}{-g_{k-1}^T d_{k-1}} + \frac{tg_k^T s_{k-1}}{(-g_{k-1}^T d_{k-1})} + \frac{tg_k^T g_{k-1}}{(-g_{k-1}^T d_{k-1})}|. \tag{44}$$

Hence,

$$\beta_k^{AOA} \leq |\frac{c\|g_k\|^2 + tg_k^T s_{k-1} + tg_k^T g_{k-1}}{(-g_{k-1}^T d_{k-1})}|. \tag{45}$$

$$\beta_k^{AOA} \leq \frac{c\|g_k\|^2 + t\|g_k\|\|s_{k-1}\| + t\|g_k\|\|g_{k-1}\|}{\|g_{k-1}\|\|d_{k-1}\|}. \tag{46}$$

By (24) and (39), we have

$$\beta_k^{AOA} \leq \frac{cc_1\|g_k\| + t\|g_k\|L + t\|g_k\|c_1}{c_1\|d_{k-1}\|} \tag{47}$$

$$= \frac{(cc_1 + t(L + c_1))\|g_k\|}{c_1\|d_{k-1}\|}. \tag{48}$$

$$\|d_k\| \leq \|g_k\| + \frac{(cc_1 + t(L + c_1))\|g_k\|}{c_1\|d_{k-1}\|}\|d_{k-1}\|. \tag{49}$$

$$\|d_k\| \leq (1 + \frac{(cc_1 + t(L + c_1)}{c_1})\|g_k\|. \tag{50}$$

$$\|d_k\|^2 \leq \frac{c_1^2 + 2c_1(cc_1 + t(L + c_1)) + (cc_1 + t(L + c_1))^2}{c_1^2}\|g_k\|^2. \tag{51}$$

$$\frac{\|d_k\|^2}{\|g_k\|^4} \leq \frac{(\frac{c_1^2+2c_1(cc_1+t(L+c_1))+(cc_1+t(L+c_1))^2}{c_1^2})c_1^2}{c_1^4}. \tag{52}$$

$$= \frac{c_1^2 + 2c_1(cc_1 + t(L + c_1)) + (cc_1 + t(L + c_1))^2}{c_1^4}. \tag{53}$$

Hence,

$$\frac{\|g_k\|^4}{\|d_k\|^2} \geq \frac{c_1^4}{c_1^2 + 2c_1(cc_1 + t(L + c_1)) + (cc_1 + t(L + c_1))^2}. \tag{54}$$

Equation (54) implies

$$\sum_{k=0}^{\infty} \frac{\|g_k\|^4}{\|d_k\|^2} \geq \frac{c_1^4}{c_1^2 + 2c_1(cc_1 + t(L + c_1)) + (cc_1 + t(L + c_1))^2} = \infty. \tag{55}$$

This is a contradiction as a result of $\|g_k\| \geq c_1$. Hence, (38) is satisfied and the result follows.

4. Numerical Test and Analysis

A number of objective functions taken from [3] are considered here for the numerical evaluation to do the comparison of the new method with existing methods.

4.1 Numerical Test

Numerical results of the test functions considered for AOA, hAO, DY and DL CG methods respectively are reported here. The following abreviations shall be used in the tables: Linear-LiR; Extended Powell-ExP; Extended Maratos-ExM; Extended Cliff- ExC; Extended Quadratic Penalty QP1-ExQP; MODF SINE-MoS; MDF EXPLIN 1-MdF1; RMODF COSINE- RMoC; RMODF SINE-RMoS; Diagonal 4-Di4; Extended Himmelblau-ExH; ARGLINB-ARB; Generalized PSCI-GePS; Generalized Rosenbrock-GeR; Extended Wood-GeW; Extended EPI-ExE; Extended Tridiagonal-2-ExT2; FLETCHCR- FLR; MODF COSINE-MoC; Power-PoW; Extended Booth-ExB; Chebyquad-CbD.

Table 1. Numerical result of CPU and function evaluation *F*.

s/*n*	*Prob.*	*Dim.*	*AOA*	*DY*	*hAO*	*DL*
			F/*CPU*	*F*/*CPU*	*F*/*CPU*	*F*/*CPU*
1	*LiR*	2	108/0.16	15/0.025	15/0.028	288/0.397
2		100	40/0.011	40/0.013	40/0.253	40/0.016
3		1000	60/0.017	60/0.018	60/0.019	60/0.021
4		10000	80/0.067	80/0.279	80/0.069	80/0.071
5	*ExP*	2	1/0.002	1/0	1/0.002	1/0.002
6	*ExM*	2	587/0.84	1299/0.911	1117/0.941	455/0.563
7		100	2523/2.297	122688/24.386	743/0.922	1904/1.774
8		1000	137331/28.555	114766/26.291	1610/1.439	21318/25.146
9		10000	2033/1.75	153918/83.78	9825/13.895	20510/38.248
10	*ExC*	2	331/0.547	1671/2.064	1654/2.54	*F*/*F*
11		100	1048/1.416	406/0.553	1914/2.971	*F*/*F*
12		1000	*F*/*F*	1065/1.535	3637/6.123	984/1.688
13		10000	*F*/*F*	2751/4.858	*F*/*F*	656/1.74
14	*ExQP*	2	30/0.021	30/0.083	30/0.022	29/0.026
15	*MoS*	2	29/0.12	25/0.1	21/0.084	505/2.297
16		100	31/0.12	25/0.12	21/0.084	257/0.391
17		1000	33/0.147	25/0.127	21/0.088	212/0.341
18		10000	33/0.173	29/0.178	21/0.137	971/2.017
19	*MdF*1	2	37/0.151	25/0.098	63/0.247	133/1.318
20		100	45/0.176	29/0.12	75/0.295	277/1.275
21		1000	49/0.195	33/0.13	79/0.318	299/1.406
22		10000	53/0.275	33/0.181	85/0.444	319/2.017
23	*RMoC*	100	189/0.401	313/0.396	218/0.431	152/0.298
24		1000	240/0.522	330/0.624	322/0.466	184/0.33
25	*RMoS*	2	37/0.155	137/0.507	55/0.224	247/1.097
26		100	41/0.165	37/0.157	63/0.259	158/0.319
27		1000	41/0.181	41/0.188	67/0.288	303/0.499
28		10000	41/0.217	41/0.255	73/0.434	265/0.948
29	*Di*4	2	104/0.028	114/0.281	233/0.724	409/0.237
30		100	254/0.077	677/0.615	412/0.349	969/0.47
31		1000	261/0.07	1309/1.211	343/0.102	769/1.704
32		10000	584/0.674	1467/1.321	705/0.813	409/0.511
33	*ExH*	2	419/0.467	1029/1.091	85/0.359	632/0.71
34		100	449/0.584	1112/1.191	409/0.394	721/0.804
35		1000	464/0.576	1186/1.459	425/0.875	917/0.917
36		10000	449/1.114	1210/2.962	425/1.253	775/2.137
37	*ARB*	2	108/0.141	15/0.015	15/0.027	288/1.328

Table 2 contd.

s/n	*Prob.*	*Dim.*	*AOA*	*DY*	*hAO*	*DL*
			F/CPU	*F/CPU*	*F/CPU*	*F/CPU*
38		100	40/0.012	40/0.014	40/0.013	40/0.014
39		1000	60/0.016	60/0.016	60/0.019	60/0.019
40		10000	80/0.065	80/0.066	80/0.065	80/0.067
41	*GePS*	2	87/0.222	91/0.229	58/0.151	81/0.229
42		100	554/1.977	1598/3.928	2964/11.044	961/3.77
43		1000	776/2.906	6742/16.186	7179/28.89	1224/5.073
44		10000	1361/9.801	19290/101.108	*F/F*	2058/16.595
45	*GeR*	100	3260/2.584	118451/57.087	5775/4.796	12757/11.888
46		1000	3541/3.369	83363/49.871	5901/4.601	43742/47.583
47		10000	3496/9.895	84861/186.101	5901/6.525	43733/123.008
48	*GeW*	2	1/0.003	1/0.003	1/0.003	1/0.003
49		100	5037/4.618	433207/107.898	3332/3.426	*F/F*
50		1000	4782/5.275	433210/216.981	3525/4.588	6165/7.977
51		10000	4156/14.737	*F/F*	3314/11.823	6425/24.453
52	*ExE*	2	53/0.034	27/0.037	27/0.035	70/0.067
53		100	53/0.036	27/0.038	27/0.048	70/0.064
54		1000	53/0.043	27/0.045	27/0.043	70/0.076
55		10000	53/0.192	27/0.127	27/0.167	70/0.28
56	*ExT*2	2	61/0.228	87/0.316	96807/39.008	*F/F*
57		100	121516/43.668	99668/43.982	52657/37.611	66197/45.647
58		1000	43982/27.529	111829/63.962	51353/48.208	88707/65.02
59	*FLR*	2	147/0.134	367/0.294	163/0.139	528/0.303
60		100	4892/5.105	7725/6.154	5219/4.757	5102/5.218
61		1000	38417/39.257	44002/42.053	38794/39.997	48247/46.886
62		10000	40069/109.747	44038/118.163	44309/115.117	39340/123.636
63	*MoC*	2	309/0.348	857/0.636	1999/1.819	665/0.976
64		100	270/0.33	1599/1.267	3255/2.584	396/0.749
65		1000	399/0.676	297/0.514	2119/3.29	396/0.892
66		10000	673/4.39	932/5.672	1028/7.045	702/5.098
67	*PoW*	2	36/0.128	10646/31.393	30/0.096	282/1.247
68		100	334/0.167	43/0.021	43/0.023	123/0.201
69		1000	742/0.259	114313/41.419	13263/32.704	157/0.397
70		10000	42/0.042	42/0.044	42/0.043	42/0.042
71	*ExB*	2	96/0.181	131/0.241	146/0.278	111/0.204
72		100	96/0.177	146/0.349	156/0.285	123/0.235
73		1000	96/0.17	155/0.303	170/0.332	111/0.208
74		10000	115/0.403	164/0.564	184/0.608	128/0.459
75	*CbD*	2	5/0.013	5/0.018	5/0.014	5/0.015
76		100	5/0.017	5/0.019	175/0.771	35/0.353
77		1000	5/0.018	5/0.02	198560/49.389	590/0.562
78		10000	5/0.022	5/0.026	144323/128.468	498/0.654

Table 2. Numerical result of number of iterations and gradient norm.

s/n	*Prob.*	*Dim.*	*AOA*	*DY*	*hAO*	*DL*
			IT/*GN*	*IT*/*GN*	*IT*/*GN*	*IT*/*GN*
1	*LiR*	2	$17/6.61e-07$	$2/2.98e-15$	$2/2.98e-15$	$141/9.88e-07$
2		100	$1/1.18e+11$	$1/1.18e+11$	$1/1.18e+11$	$1/1.18e+11$
3		1000	$1/3.30e+17$	$1/3.30e+17$	$1/3.30e+17$	$1/3.30e+17$
4		10000	$1/9.06e+23$	$1/9.06e+23$	$1/9.06e+23$	$1/9.06e+23$
5	*ExP*	2	$0/0.00e+00$	$0/0.00e+00$	$00.00e+00$	$0/0.00e+00$
6	*ExM*	2	$59/5.56e-07$	$113/9.60e-07$	$117/1.15e+05$	$44/6.81e-07$
7		100	$233/9.57e-07$	$4000/1.07e-05$	$76/5.01e-07$	$178/8.65e-07$
8		1000	$4000/1.30e-05$	$4000/3.02e-05$	$167/7.82e-07$	$4000/4.26e-06$
9		10000	$94/1.30e-07$	$4000/1.59e-04$	$1043/7.32e-07$	$4000/1.33e-04$
10	*ExC*	2	$62/6.64e-07$	$233/4.97e-07$	$315/6.86e-07$	F/F
11		100	$173/9.86e-07$	$69/9.21e-07$	$364/6.13e-07$	F/F
12		1000	F/F	$182/6.81e-07$	$690/4.50e-07$	$159/6.57e-07$
13		10000	$7/F$	$8/F$	$8/F$	$5/9.57E-07$
15	*MoS*	2	$14/4.68e-07$	$12/3.55e-07$	$10/1.09e-08$	$252/9.76e-07$
16		100	$15/6.82e-07$	$12/2.26e-07$	$10/1.21e-08$	$37/8.49e-07$
17		1000	$16/3.34e-07$	$12/6.80e-07$	$10/3.67e-08$	$33/9.66e-07$
18		10000	$16/9.76e-07$	$14/1.09e-07$	$10/1.16e-07$	$115/8.90e-07$
19	*MdF*1	2	$18/7.96e-07$	$12/4.29e-07$	$31/7.67e-07$	$116/9.28e-07$
20		100	$22/6.74e-07$	$14/4.80e-07$	$37/6.70e-07$	$138/9.09e-07$
21		1000	$24/6.25e-07$	$16/1.71e-07$	$39/9.46e-07$	$149/9.07e-07$
22		10000	$26/5.76e-07$	$16/5.42e-07$	$42/8.86e-07$	$159/1.00E-06$
23	*RMoC*	100	$48/4.71e-07$	$53/9.30e-07$	$54/5.28e-07$	$29/5.75e-07$
24		1000	$60/4.86e-07$	$76/8.22e-07$	$73/7.61e-07$	$29/5.35e-07$
25	*RMoS*	2	$18/5.50e-07$	$68/9.55e-07$	$27/7.24e-07$	$123/9.21e-07$
26		100	$20/5.28e-07$	$18/8.77e-07$	$31/7.87e-07$	$31/5.85e-07$
27		1000	$20/6.06e-07$	$20/3.22e-07$	$33/8.94e-07$	$47/9.23e-07$
28		10000	$20/5.76e-07$	$20/6.46e-07$	$36/6.10e-07$	$70/8.67e-07$
29	*Di*4	2	$33/9.44e-07$	$34/9.27e-07$	$92/9.97e-07$	$17/6.26e-08$
30		100	$27/7.72e-07$	$75/7.20e-07$	$45/5.54e-07$	$38/5.56e-07$
31		1000	$26/7.99e-07$	$146/8.55e-07$	$36/3.90e-07$	$35/9.16e-07$
32		10000	$63/1.76e-07$	$151/9.46e-07$	$78/3.82e-07$	$111/8.10e-16$
33	*ExH*	2	$56/8.77e-15$	$123/1.24e-14$	$42/4.17e-15$	$64/8.57e-07$
34		100	$60/2.86e-15$	$133/1.64e-14$	$45/1.34e-15$	$72/5.65e-07$
35		1000	$62/5.07e-17$	$142/2.94e-15$	$47/9.92e-17$	$74/9.58e-07$
36		10000	$60/6.13e-15$	$145/7.50e-15$	$47/9.92e-16$	$78/8.87e-07$
37	*ARB*	2	$17/6.61e-07$	$2/2.98e-15$	$2/2.98e-15$	$1419.88e-07$

Table 2 contd.

s/n	*Prob.*	*Dim.*	*AOA*	*DY*	*hAO*	*DL*
			IT/*GN*	*IT*/*GN*	*IT*/*GN*	*IT*/*GN*
38		100	$1/1.18E+11$	$1/1.18E+11$	$1/1.18E+11$	$1/1.18E+11$
39		1000	$1/3.30E+17$	$1/3.30E+17$	$1/3.30E+17$	$1/3.30E+17$
40		10000	$1/9.06E+23$	$1/9.06E+23$	$1/9.06E+23$	$1/9.06E+23$
41	*GePS*	2	$25/9.71E-07$	$26/6.70E-07$	$17/4.64E-07$	$22/5.62E-07$
42		100	$235/9.05E-07$	$478/9.60E-07$	$1335/8.88E-07$	$396/7.52E-07$
43		1000	$333/7.14e-07$	$1724/7.70e-07$	$3257/9.91e-07$	$503/7.25e-07$
44		10000	$564/8.71e-07$	$4000/7.50e-05$	F/F	$846/7.10e-07$
45	*GeR*	100	$283/8.95e-07$	$4000/1.48e+01$	$532/6.96e-07$	$1159/9.83E-07$
46		1000	$308/8.51e-07$	$4000/1.27e+02$	$540/7.38e-07$	$4000/1.21e+01$
47		10000	$297/8.34e-07$	$4000/1.27e+02$	$457/8.97e-07$	$4000/1.79e+01$
48	*GeW*	2	$0/0.00e+00$	$0/0.00e+00$	$0/0.00e+00$	$0/0.00e+00$
49		100	$454/9.03e-07$	$4000/5.21e+04$	$317/8.86e-07$	F/F
50		1000	$410/9.85e-07$	$4000/1.65e+05$	$335/8.77e-07$	$552/9.44e-07$
51		10000	$361/7.57e-07$	F/F	$315/3.80e-07$	$578/9.35e-07$
52	*ExE*	2	$3/1.46e+11$	$3/4.55e+05$	$3/4.55e+05$	$5/F$
53		100	$3/1.03e+12$	$3/3.22e+06$	$3/3.22e+06$	$5/F$
54		1000	$3/3.26e+12$	$3/1.02e+07$	$3/1.02e+07$	$5/F$
55		10000	$3/1.03e+13$	$3/3.22e+07$	$3/3.22e+07$	$5/F$
56	*ExT*2	2	$26/7.88e-07$	$38/8.97e-07$	$4000/3.15e+00$	F/F
57		100	$3763/1.01E+10$	$4000/1.58e+03$	$4000/3.09e+00$	$4000/4.18e+01$
58		1000	$1852/1.28e+10$	$4000/2.37e+04$	$4000/2.33e+01$	$4000/1.69e+02$
59	*FLR*	2	$14/1.05e-07$	$31/5.79e-07$	$14/3.18e-08$	$26/2.29e-07$
60		100	$482/9.77e-07$	$702/9.88e-07$	$533/9.00e-07$	$503/9.91e-07$
61		1000	$3837/9.92e-07$	$4000/1.51e-05$	$4000/5.75e-04$	$3862/9.12e-07$
62		10000	$4000/2.46e-03$	$4000/1.09e-02$	$4000/3.14e-02$	$4000/1.06e-03$
63	*MoC*	2	$41/1.68e+11$	$70/F$	$209/5.83e+09$	$99/4.53e+10$
64		100	$36/2.53e+09$	$134/F$	$272/1.64e+10$	$75/2.80e+10$
65		1000	$44/1.63e+10$	$37/1.40e+10$	$202/3.70e+09$	$67/7.40e+09$
66		10000	$84/F$	$119/4.11e+11$	$126/2.24e+10$	$94/1.21e+10$
67	*PoW*	2	$15/9.98e-07$	$4000/7.54e+00$	$11/8.12e-07$	$139/9.60e-07$
68		100	$17/3.12e+07$	$2/0.00e+00$	$2/0.00e+00$	$19/4.42e+07$
69		1000	$23/6.98e-07$	$4000/1.68e+07$	$4000/6.38e-02$	$42/7.78e-07$
70		10000	$1/1.42e+12$	$1/1.42e+12$	$1/1.42e+12$	$1/1.42e+12$
71	*ExB*	2	$20/4.23e-08$	$27/8.62e-07$	$30/4.84e-07$	$19/6.97e-07$
72		100	$20/2.99e-07$	$30/9.74e-07$	$32/8.17e-07$	$21/2.87e-07$
73		1000	$20/9.47e-07$	$32/6.55e-07$	$35/4.72e-07$	$19/F$
74		10000	$24/3.87e-07$	$34/9.83e-07$	$38/7.58e-07$	$23/3.63e-07$
75	*CbD*	2	$1/0.00e+00$	$1/0.00e+00$	$1/0.00e+00$	$1/0.00e+00$
76		100	$2/2.68E-16$	$2/2.68e-16$	$85/9.41e-07$	$35/F$
77		1000	$2/2.36e-15$	$2/2.36e-15$	$4000/6.05e-02$	$49/F$
78		10000	$2/1.98e-15$	$2/1.98e-15$	$4000/1.92e-02$	$36/F$

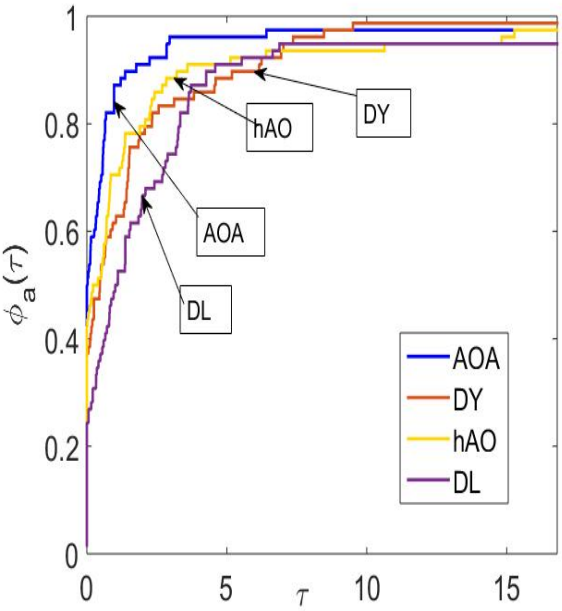

Figure 1: Function Evaluations (F)

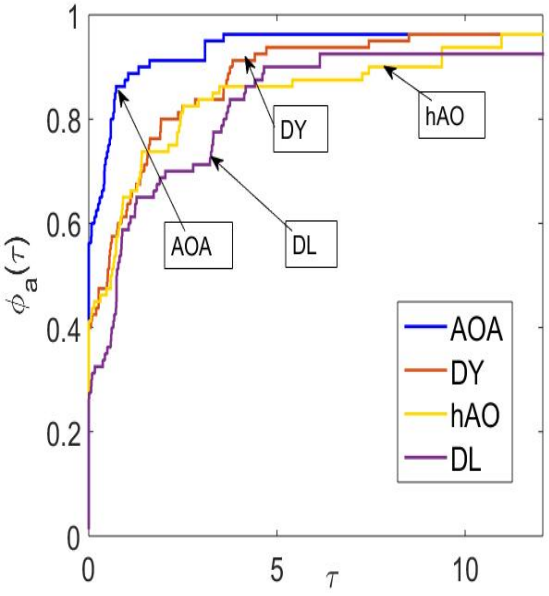

Figure 3: Iterations (IT)

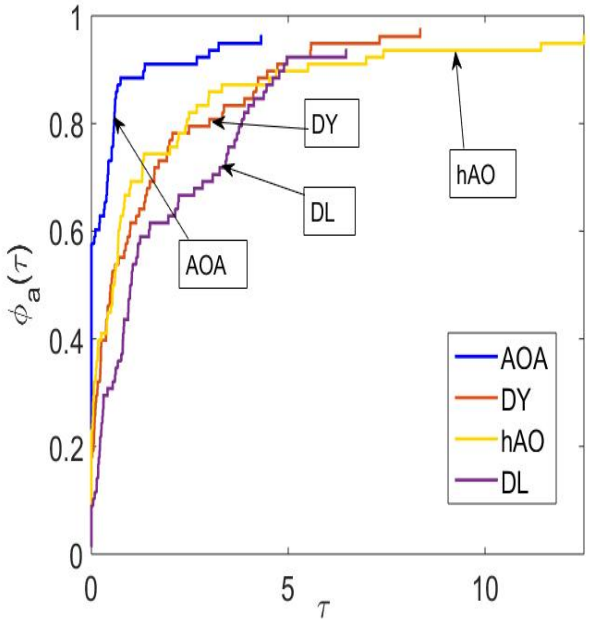

Figure 2: CPU time

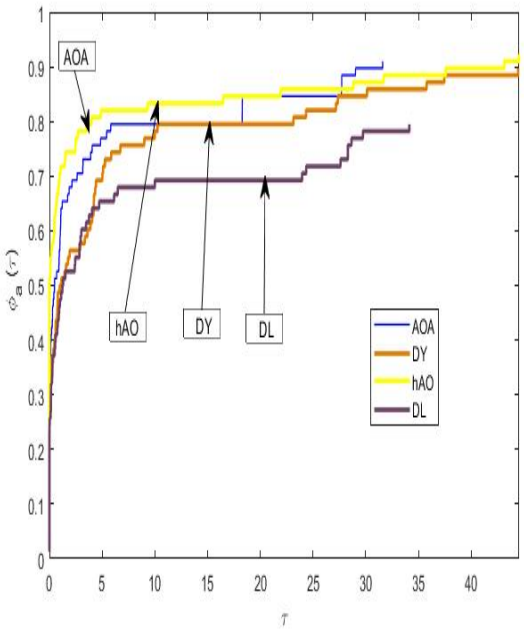

Figure 4: Gradient norm (GN)

4.2 Numerical Analysis

This subsection presents a comparison of numerical efficiency of the AOA method against hAO [1], DY [7] and DL [6] CG methods. A number of test functions are drawn from [3]. Meanwhile, the algorithms are coded and applied on MATLAB *R*2015*a* on HP 650 windows 10 OS and RAM 3GB. A varying dimensions $2, 100, 1000, 10000$ were used while the performance comparison were based on the theory of Dolan and Moré [9]: Suppose set A of r_a methods is to be compared and U is the set of n_u test problems. The implementation of the tools is done using the ratio

$$r_{u,a} = \frac{R_{u,a}}{min\{R_{u,a} : a \in A \text{ and } u \in U\}} \tag{56}$$

to compare different methods, where $R_{u,a}$ is either IT, F, GN or CPU time for each method A and problem U. However,

$$\phi_a(\tau) = \frac{1}{n_u}|U \in U : logr_{u,a}| \leq \tau \tag{57}$$

gives the overall cumulative frequency function for $r_{u,a}$, where τ is nonnegative. The chances that $r_{u,a}$ falls within a factor $\tau \geq 1$ in connection to the method a is $\phi_a(\tau)$. However, for the value $\tau = 1$, $\phi_a(\tau)$ is the probability that one method will perform better than other methods. The chosen method $a \in A$ fails to solve a problem if $r_i = r_{u,a}$ for some parameter r_i, .
Meanwhile, the iteration is stopped, if

$$\|g_k\| \leq 10^{-6}.$$

Parameter Setting: The algorithms run with $\delta = 0.0001$, $\sigma = 0.9$ and we select $t = 0.1$ for AOA and DL methods. Maximum number of iterations is set at 4000.

In Table 1, we present the numerical results in the form F/CPU, where F and CPU denote function evaluations and CPU time in seconds respectively. While in Table 2, we have numerical results in the form IT/GN for which IT and GN stand for number of iterations and gradient norms respectively. In addition, F in the table indicates that the method fails in solving the problem.
For convincing performance display, we employ a Dolan and More [9] performance profile of the four comparable schemes. Plotted in Figures 1, 2, 3 and 4 are the corresponding profiles. Performance profile of function evaluation is shown in Figure 1, Figure 2 shows the performance profile of CPU time, Figure 3 presents the profile of IT and in Figure 4, we have the performance profile for the gradient norms. Figure 1 illustrates that AOA method performs better than the other methods for the values of τ between 0 and 10 and compete well for the other values of τ in the area of function evaluations. The proposed method converges faster than the other methods in Figure 3. In term of CPU time shown in Figure 2, AOA method outperformed other methods while in Figure 4, it competes well with the other methods for the initial values of τ up till 30 and latter did better than the rest.

5. Conclusion

A new CG method was presented in this paper due to the global acceptability of CG method in solving large-scale optimization problem of the form (1). This construction was done by using conjugacy condition introduced by Dai and Liao [6] coupled with some modifications on the direction d_k. Global convergence and descent properties of the new method were shown. Numerical test with this method showed that the method competes well with an hybrid method as well as notable CG algorithms in the literature. Various modifications of this algorithm can still be considered for further work.

References

[1] Adeleke, O. J. and Osinuga, I. A., *A five-term hybrid conjugate gradient method with global convergence and descent properties for unconstrained optimization problem*, Asian journal of scientific research, **11(2)**, (2018), 185-194.

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

[3] Andrei, N., *Test functions for unconstrained optimization, Research Institute for informatics*, Center for Advanced Modeling and Optimization, 8-10, Averescu Avenue, Sector 1, Bucharest, Romania, (2004), 1-15.

[4] Andrei, N., *Open problems in nonlinear conjugate gradient algorithms for unconstrained optimization*, Bulletin of the Malaysian Mathematical Sciences Society, Second Series, **34(2)**, (2011), 319–330.

[5] Babaie-Kafaki, S., *A survey on the Dai-Liao family of nonlinear conjugate gradient methods*, RAIRO-Operations Research, **57**, (2023), 43-58.

[6] Dai, Y. H. and Liao, L. Z., *New conjugacy conditions and related nonlinear conjugate gradient methods*, Applied Mathematics and Optimization, (2001), **43**, 87-101.

[7] Dai, Y. H. and Yuan, Y., *A nonlinear conjugate gradient method with a strong global convergence property*, SIAM Journal on Optimization, **10**, (1999), 177-182.

[8] Dai, Y.H. and Yuan, Y., *An efficient hybrid conjugate gradient method for unconstrained optimization*, Annals of Operations Research, **103**, (2001), 33-47.

[9] Dolan, E. D. and More, J. J., *Benchmarking optimization software with performance profiles*, Mathematical Programming, **91**, (2002), 201-213.

[10] Fletcher, R., *Practical methods of optimization*, John Wiley & Sons, 2013.

[11] Fletcher, R. and Reeves, C. M. *Function minimization by conjugate gradients*, Computer Journal, **7**, 1964, 149-154.

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

[13] Hestenes, M. R. and Stiefel, E., *Methods of conjugate gradients for solving linear systems*, Journal of Research of the National Bureau of Standards, **49**, (1952), 409-435.

[14] Hu Y. and Storey, C., *Global convergence result for conjugate gradient methods*, Journal of Optimization Theory and Applications, **71**, (1991), 399-405.

[15] Liu, G. Han J. and Yin, H., *Global convergence of the Fletcher-Reeves algorithm with inexact line search*, Appl. Math. J. Chinese Univ. Ser. B, **10**, (1995), 75-82.

[16] Liu, Y. and Storey, C., *Efficient generalized conjugate gradient algorithms, part 1, Theory*, Journal of Optimization Theory and Applications, **69**, (1991), 129-137.

[17] Polak, E. and Ribiere, G., *Note sur la convergence de directions conjugees*, Revue Francaise d'Informatique et de Recherche Operationnelle, **16**, (1969), 35-43.

[18] Polyak, B. T., *The conjugate gradient method in extremal problems*, USSR Computational Mathematics and Mathematical Physics, **9**, (1969), 94-112.

[19] Powell, M. J. D., *Restart procedures for the conjugate gradient method*, Mathematical Programming, **2**, (1977), 241-254.

[20] Stanimirovic, P. S., Ivanov, B and Mosic, D., *A survey of gradient methods for solving nonlinear optimization*, Electronic Research Archive, **28(4)**, (2020), 1573– 1624.

[21] Zoutendijk, G., *Nonlinear programming, computational methods*, In: J. Abadie (eds.): Integer and Nonlinear Programming, North-Holland, Amsterdam, (1970), 37–86.