*Journal of Interdisciplinary Mathematics*
ISSN: 0972-0502 (Print), ISSN: 2169-012X (Online)
Vol. 28 (2025), No. 1, pp. 19–30
DOI : 10.47974/JIM-1771

# New conjugate gradient methods based on the modified secant condition

Hawraz N. Jabbar [@]
*Department of Mathematics*
*College of Sciences*
*University of Kirkuk*
*Kirkuk 36001*
*Iraq*

Ali A. Al-Arbo [†]
*College of Arts*
*University of Mosul*
*Mosul 41002*
*Iraq*

Yoksal A. Laylani [§]
*Department of Mathematics*
*College of Sciences*
*University of Kirkuk*
*Kirkuk 36001*
*Iraq*

Hisham M. Khudhur [‡]
Basim Abbas Hassan *
*Department of Mathematics*
*College of Computers Sciences and Mathematics*
*University of Mosul*
*Mosul 41002*
*Iraq*

[@] *ORCID-ID:* 0000-0001-6867-8027
[†] *ORCID-ID:* 0000-0002-7023-7236
[§] *ORCID-ID:* 0000-0003-4898-5083
[‡] *ORCID-ID:* 0000-0001-7572-9283
* *ORCID-ID:* 0000-0003-3510-9818

[@] *E-mail:* hawrazmath@uokirkuk.edu.iq
[†] *E-mail:* ali.abbass.y@uomosul.edu.iq
[§] *E-mail:* yuksel@uokirkuk.edu.iq
[‡] *E-mail:* hisham892020@uomosul.edu.iq
* *E-mail:* basimah@uomosul.edu.iq; (Corresponding Author)
basimabas39@gmail.com

© Taru Publications

**Abstract**

Conjugate gradient (CG) algorithms are an optimization method with fast convergence. To date, several CG methods have been developed to improve computational performance and have been used to solve unconstrained optimization problems. In this paper, a new CG algorithm based on the modified secant condition is proposed to solve unconstrained optimization problems. The proposed algorithm has the following properties:it achieves sufficient descent property and global convergence property, We used a number of test functions for large-scale unconstrained optimization problems and the numbers showed that the proposed algorithm works better than other algorithms like Hestenes and Steifel (HS) and Fletcher and Revees (FR).

***Subject Classification:*** *90C30, 65K05, 49M37.*

***Keywords:*** *Algorithm, Conjugacy condition, Secant condition, Global, Optimization.*

## 1. Introduction

In the field of unconstrained optimization, there arises the challenge of tackling large-scale problems that are formulated of the following form

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

problems in the form of Eq(1) have garnered significant attention within the optimization community due to their extensive range of applications in various fields [1]. These problems typically arise when considering a smooth function $f: R^n \to R$, which implies that the gradient $g_k = \nabla f(x)$ is obtainable. Numerical techniques that use the first partial derivatives of $f$ may tackle unconstrained optimization problems [23-25]. A key algorithm in this discipline is the conjugate gradient algorithm. This method is popular because it solves these problems with low memory and has good theoretical features. To solve, start with an initial value and successively acquire values until reaching the end sequence $\{x_k\}$. This method is recognized for solving optimization issues well, making it a good option for real-world applications. The conjugate gradient technique rapidly finds the solution space, resulting in excellent results. Usually, the conjugate gradient algorithm is guaranteed to reach the solution because it has guaranteed convergence to the solution. Because this algorithm is iterative, each successive step brings us closer to the solution until we reach it. In addition to this, the CG approach makes use of the gradient of the smooth function $f$ in order to navigate through the solution space in a planned manner, so avoiding any unnecessary detours or side excursions. The CG method quickly gets to the right answer by changing the solution over and over again based on the gradient information [26]. Overall, the Conjugate Gradient method is a useful tool for optimization because it has both useful applications and strong theoretical roots. It is widely used and has a good image, which make it even more of a go-to method for solving problems like the ones in Eq.

$$x_{k+1} = x_k + \alpha_k d_k, k = 0,1,2,3, \ldots \quad (2)$$

The scalar $\alpha_k > 0$, which is also known as the step-size or steplength, is found by doing a lot of careful calculations with the goal of almost meeting the condition.

$$f(x_k + \alpha_k d_k) \approx \min_{\alpha \geq 0} \{f(x_k + \alpha d_k)\} \quad (3)$$

The use of the well-known Wolfe condation, which is discussed in reference [2], is a technique that may be used in practice to get the step-size. This particular technique, which is executed at the $k - th$ iteration, is calculated in such a manner that $\alpha_k$ adheres to specific pre-established principles, as outlined in reference [3]. The frequently employed technique, commonly referred to as the standard line search technique, encompasses a set of inequalities that are utilized for this purpose

$$f(x_k + \alpha_k d_k) - f(x_k) \leq \delta \alpha_k g_k^T d_k \quad (4)$$

$$g_{k+1}^T d_k \geq \sigma g_k^T d_k \quad (5)$$

Using the formula in place of the Eq (5)

$$|g_{k+1}^T d_k| \geq -\sigma g_k^T d_k \quad (6)$$

The rule referred to as the strong Wolfe condition, which is a pivotal aspect of optimization algorithms, is being presented. This rule is characterized by the condition where $0 < \delta < \sigma < 1$. Moreover, the determination of the step-size is intricately linked to the calculation conducted along the trajectory of search direction $d_{k+1}$, given by

$$d_{k+1} = \begin{cases} -g_{k+1}, & \text{if } k = 0 \\ -g_{k+1} + \beta_k d_k, & \text{if } k \geq 1 \end{cases} \quad (7)$$

The CG updating parameters, which are denoted by the scalar $\beta_k$, play a significant role in the construction and determination of the choice of a CG method. Various $\beta_k$ formulas have been proposed by different researchers for these parameters. Fletcher and Revees (FR) [4], Polak, Ribière, Polyak (PRP) [5], Hestenes and Steifel (HS) [6], Liu and Storey (LS) [7], Dai and Yuan (DY) [8], Conjugate Descent (CD) [9] have all put forward their own formulas for the calculation of these parameters. Each of these formulas has its own distinct characteristics and advantages. The formulas presented by Fletcher and Revees, Polak, Ribière, Polyak, Hestenes and Steifel, Liu and Storey, Dai and Yuan, Conjugate Descent are expressed as follows respectively.

$$\beta_k^{FR} = \frac{\|g_{k+1}\|^2}{\|g_k\|^2} \quad (8)$$

$$\beta_k^{PRP} = \frac{g_{k+1}^T y_k}{\|g_k\|^2} \quad (9)$$

$$\beta_k^{HS} = \frac{g_{k+1}^T y_k}{d_k^T y_k} \quad (10)$$

$$\beta_k^{LS} = \frac{g_{k+1}^T y_k}{-d_k^T g_k} \quad (11)$$

$$\beta_k^{DY} = \frac{\|g_{k+1}\|^2}{d_k^T y_k} \tag{12}$$

$$\beta_k^{CD} = \frac{\|g_{k+1}\|^2}{d_k^T g_k} \tag{13}$$

where $y_k = g_{k+1} - g_k$ and $\|.\|$ denotes the Euclidean norm.

In the second section, we will derive a new method from the conjugate gradient algorithms based on the modified secant condition to overcome the slow convergence of the FR and HS algorithms, we will also prove the properties of descent and global convergence of the algorithm.

## 2. Our new formulas

In 2021, Basim et. al. [10] presented a modified secant condition:

$$s_k^T Q(u_k) s_k = (1+\vartheta) s_k^T y_k \tag{14}$$

where $\vartheta = 0,1,2,3$. In general derivatives difference $y_k$ defined the relation as:

$$y_k = g_{k+1} - g_k = Q(x_k) s_k \tag{15}$$

From Eq (14) and Eq (15), it can be deduced that:

$$s_k^T Q(u_k) s_k = \frac{1}{(1+\vartheta)} s_k^T y_k \tag{16}$$

Multiplying Eq (15) by $s_k^T$ thus:

$$s_k^T Q(x_k) s_k = s_k^T g_{k+1} - s_k^T g_k \tag{17}$$

Using Eq (16) in Eq (17), by benefit of Perry's conjugacy condition $d_{k+1}^T y_k = -s_k^T g_{k+1}$, furthermore:

$$d_{k+1}^T y_k = -\frac{1}{(1+\vartheta)} s_k^T y_k - s_k^T g_k \tag{18}$$

Putting $d_{k+1} = -g_{k+1} + \beta_k s_k$ in Eq (18) imply that:

$$\beta_k s_k^T y_k = g_{k+1}^T y_k - \frac{1}{(1+\vartheta)} s_k^T y_k - s_k^T g_k \tag{19}$$

Accordingly

$$\beta_k = \frac{g_{k+1}^T y_k}{s_k^T y_k} - \frac{1}{(1+\vartheta)} - \frac{s_k^T g_k}{s_k^T y_k} \tag{20}$$

Procedure Eq (20) will be called BBH method. Now we represent our algorithm for solving optimization problems. The algorithm is given as follows.

**New Algorithm 2.1:**

Step 1. Input, $x_1 \in R^n, d_1 = -g_1$.

Step 2. If $\|g_{k+1}\| \leq 10^{-6}$, then stop.

Step 3. Evaluate the $\alpha_k > 0$ based on Wolfe-condition Eq (4), Eq (5) and the $x_{k+1} = x_k + \alpha_k d_k$.

Step 4. Determine $\beta_k$ based on Eq (20).

Step 5. Evaluate $d_{k+1} = -g_{k+1} + \beta_k d_k$. Set $k = k + 1$and move to step2.

## 3. Convergence analysis

The discussion of the convergence analysis for the algorithm here based on the Lipschitz condition. The assumptions below are defined for the objective function.

*Assumption 3.1*

1. Level is set to $L_0 = \{x \in R^n : f(x) \leq f(x_0)\}$ be convex.
2. Gradient is Lipschitz continuous, hence there exists positive constant $L > 0$:

$$\left(\nabla f(\bar{x}) - \nabla f(x^+)\right) \leq L\left\|\bar{x} - x^+\right\|, \forall\ \bar{x}, x^+ \in L_0 \tag{21}$$

More details can be found in [10]-[13].

**Theorem 1:** *If $d_{k+1}$ is the solution of Eq (1), then $d_{k+1}^T g_{k+1} \leq -c\|g_{k+1}\|^2$ holds.*

***Proof:*** Since $d_0 = -g_0$, therefore, we obtain $g_0^T d_0 = -\|g_0\|^2 \leq 0$. Let $g_k^T d_k < 0$ for all $k \in n$. Multiply Eq (7) by $g_{k+1}^T$,, we get:

$$d_{k+1}^T g_{k+1} = -\left\|g_{k+1}\right\|^2 + \left[\frac{g_{k+1}^T y_k}{s_k^T y_k} - \frac{1}{(1+\vartheta)} - \frac{s_k^T g_k}{s_k^T y_k}\right] s_k^T g_{k+1} \tag{22}$$

Combining Eq (15) and Eq (20), we get:

$$d_{k+1}^T g_{k+1} = -\left\|g_{k+1}\right\|^2 + \left[\frac{g_{k+1}^T y_k}{s_k^T y_k} - \frac{s_k^T g_{k+1}}{s_k^T y_k}\right] s_k^T g_{k+1} \tag{23}$$

By Eq (21), we conclude that:

$$y_k^T g_{k+1} \leq L\, s_k^T g_{k+1} \tag{24}$$

From Eq (23) and Eq (24), we conclude that:

$$d_{k+1}^T g_{k+1} \leq -\left\|g_{k+1}\right\|^2 + \left[L\,\frac{s_k^T g_{k+1}}{s_k^T y_k} - \frac{s_k^T g_{k+1}}{s_k^T y_k}\right] s_k^T g_{k+1} \tag{25}$$

Thus, we get $$d_{k+1}^T g_{k+1} \leq -\left\|g_{k+1}\right\|^2 + \left[L - 1\right]\frac{(s_k^T g_{k+1})^2}{s_k^T y_k} \tag{26}$$

Moreover,

$$d_{k+1}^T g_{k+1} \leq -\left\|g_{k+1}\right\|^2 < 0 \tag{27}$$

So, the proof is complete.

Now, to prove the convergence of the BBH, we need to use the following result.

**Lemma 1:** *Any search direction in conjugate gradient, where $d_{k+1} = -g_{k+1} + \beta_k s_k$, it's descent and $\alpha_k$is compute by the Wolfe-condition, assuming assumptions (i) and (ii) holds. If:*

$$\sum_{k>1} \frac{1}{\left\|d_{k+1}\right\|^2} = \infty \tag{28}$$

then: $$\lim_{k\to\infty}(\inf\left\|g_{k+1}\right\|) = 0. \tag{29}$$

The proof of this theorem find in [14].

In next theorem, the convergence property result of the new Algorithm is shown.

**Theorem 2:** *If $f$ is a uniformly convex function and a constant $\mu > 0$ exists such that:*

$$(\nabla f(z) - \nabla f(u))^T \geq \mu\left\|z-u\right\|^2 \ , \forall\ z, u \in R^n \tag{30}$$

then $$\lim_{k\to\infty}(\inf\left\|g_{k+1}\right\|) = 0 \tag{31}$$

***Proof:*** Correspondingly, the $\beta_k$ is generated by:

$$\left|\beta_k\right| = \left|\left[\frac{g_{k+1}^T y_k}{s_k^T y_k} - \frac{1}{(1+\vartheta)} - \frac{s_k^T g_k}{s_k^T y_k}\right]\right| = \left|\frac{g_{k+1}^T y_k}{s_k^T y_k} - \frac{s_k^T g_{k+1}}{s_k^T y_k}\right| \tag{32}$$

Through Eq (21), we obtained $\|y_k\| \leq L\|s_k\|$, while $s_k^T y_k \geq \mu\|s_k\|^2$ is obtained by using Eq (17). By apply the Cauchy inequality to the previous inequalities, then (30), we get:

$$\left|\beta_k\right| \leq \frac{L\left\|s_k\right\|\left\|g_{k+1}\right\|}{\mu\left\|s_k\right\|^2} + \frac{\left\|s_k\right\|\left\|g_{k+1}\right\|}{\mu\left\|s_k\right\|^2} \leq \left(\frac{L+1}{\mu}\right)\frac{¥}{\left\|s_k\right\|} \tag{33}$$

If f(x) satisfies Assumptions, there exists a constant $¥ > 0$such that $\|g(x)\| \leq ¥$ for all x. From Eq (33) in Eq (7), we get:

$$\left\|d_{k+1}\right\| \leq \left\|g_{k+1}\right\| + \left|\beta_k\right|\left\|s_k\right\| \leq \frac{¥(\mu+L+1)}{\mu} \tag{34}$$

demonstrating that Eq (28) is true.

## 4. Numerical results

In this section, experiments are conducted to verify the effectiveness of BBH (0.25), BBH (0.3), BBH (0.5) compare to the original algorithms HS, FR. For experiments, the same 46 unconstrained problems are tested and compared, in which the problems 1-49 are taken from the CUTE library [15]-[17]. The test problems that are being considered in this study exhibit a wide range of dimensions, ranging from as small as 11 to as large as 1000. In order to ensure a fair and consistent comparison among different methods, all of the comparison methods utilized in this research employ the weak Wolfe line search technique (as described in reference [8]) to compute the step-length, denoted as ak. In addition, the relevant parameters for this line search method, namely the values of $\delta$ and $\sigma$, are set to 0.01 and 0.1, respectively. As for our own proposed

methods, we have chosen to adopt the strategy outlined in reference [18] for computing the initial step-length. Lastly, the termination criterion for these optimization algorithms are:$\|g_k\| \leq 10^{-6}$ or Iterations > 2000, Iterations (Itr) will reflect the total number of times the process is repeated. In the event that condition (2) is satisfied, we assert that the relevant algorithm cannot solve the related test problem, we refer to this assertion as fail(F). All of the scripts are written in Matlab 2022b, they are executed on a Windows 10 operating system-based HP computer with a 2.60 GHz CPU processor, 8 GB RAM memory, the Matlab 2022b programming language. In the experiments, we report the number of iterations (Iterations (Itr)), the amount of CPU time (Tcpu), the final value for $\|g_k\|$ ($\|g_*\|$) in Table 1. We also make use of the performance profiles that were proposed by Dolan and More [19] to visually describe the performance of these algorithms in terms of the amount of Tcpu and Itr, respectively. In terms of the interpretation of the performance profiles, the top curve indicates, in a more general sense, that the applicable approach is the winner; see [19-24] for additional information on this.

**Table 1**
**Numerical results and comparisons between BH(0.25), BH(0.3), BH(0.5), HS, FR**

| Problems | | BH (0.25) | | BH (0.3) | | BH (0.5) | | HS | | FR |
|---|---|---|---|---|---|---|---|---|---|---|
| Name/Dim | Iter | time cpu | Iter | time cpu | Iter | time cpu | Iter | time cpu | Iter | time cpu |
| cosine/100 | 37 | 0.065 | 70 | 0.020 | 103 | 0.024 | 253 | 0.057 | 52 | 0.015 |
| cosine/1000 | 494 | 0.362 | 245 | 0.213 | 63 | 0.047 | NAN | NAN | NAN | NAN |
| dixmaana/90 | 15 | 0.049 | 18 | 0.010 | 18 | 0.012 | 33 | 0.030 | 21 | 0.017 |
| dixmaana/900 | 21 | 0.062 | 16 | 0.050 | 21 | 0.061 | 27 | 0.106 | 21 | 0.051 |
| dixmaanb/90 | 16 | 0.07 | 16 | 0.017 | 17 | 0.020 | 31 | 0.032 | 13 | 0.017 |
| dixmaanb/900 | 20 | 0.094 | 21 | 0.066 | 22 | 0.055 | 20 | 0.060 | 20 | 0.052 |
| dixmaanc/90 | 19 | 0.051 | 15 | 0.018 | 22 | 0.018 | 26 | 0.025 | 23 | 0.018 |
| dixmaanc/900 | 21 | 0.06 | 17 | 0.063 | 20 | 0.050 | 32 | 0.118 | 25 | 0.064 |
| dixmaand/90 | 20 | 0.055 | 19 | 0.017 | 23 | 0.024 | 27 | 0.027 | 27 | 0.019 |
| dixmaand/900 | 19 | 0.063 | 20 | 0.050 | 24 | 0.065 | 39 | 0.185 | 26 | 0.057 |
| dixmaane/90 | 93 | 0.107 | 99 | 0.031 | 171 | 0.067 | 88 | 0.038 | 196 | 0.072 |
| dixmaane/900 | 234 | 0.361 | 274 | 0.363 | 520 | 0.655 | 230 | 0.204 | 642 | 0.606 |
| dixmaanf/90 | 83 | 0.071 | 88 | 0.059 | 181 | 0.080 | 77 | 0.029 | 226 | 0.086 |
| dixmaanf/900 | 227 | 0.312 | 321 | 0.384 | 597 | 0.611 | 179 | 0.138 | 658 | 0.636 |
| dixmaang/90 | 83 | 0.064 | 88 | 0.034 | 135 | 0.068 | 79 | 0.038 | 253 | 0.080 |
| dixmaang/900 | 225 | 0.3 | 190 | 0.231 | 540 | 0.620 | 218 | 0.155 | NAN | NAN |
| dixmaanh/90 | 84 | 0.076 | 91 | 0.036 | 142 | 0.054 | 81 | 0.030 | 147 | 0.048 |
| dixmaanh/900 | 241 | 0.365 | 216 | 0.263 | 382 | 0.436 | 240 | 0.200 | 497 | 0.449 |
| dixmaani/90 | 991 | 0.367 | 718 | 0.282 | 1458 | 0.496 | 847 | 0.163 | 1310 | 0.363 |
| dixmaanj/90 | 329 | 0.155 | 262 | 0.118 | 1798 | 0.646 | 602 | 0.144 | NAN | NAN |
| dixmaank/90 | 631 | 0.265 | 775 | 0.243 | 1174 | 0.442 | 605 | 0.133 | NAN | NAN |
| dixmaanl/90 | 622 | 0.29 | 850 | 0.347 | 1103 | 0.390 | 695 | 0.147 | NAN | NAN |
| dixmaanl/900 | .NAN | .NAN | 1423 | 1.312 | .NAN | .NAN | .NAN | .NAN | .NAN | .NAN |
| dixon3dq/30 | 255 | 0.042 | 353 | 0.033 | 350 | 0.028 | 332 | 0.041 | 1272 | 0.190 |
| dqdrtic/30 | 71 | 0.028 | 91 | 0.016 | 86 | 0.020 | 79 | 0.020 | 110 | 0.013 |
| dqdrtic/300 | 65 | 0.023 | 99 | 0.017 | 75 | 0.024 | 72 | 0.023 | 187 | 0.032 |
| edensch/100 | 47 | 0.03 | 37 | 0.010 | 57 | 0.024 | 68 | 0.051 | 144 | 0.083 |
| edensch/1000 | 71 | 0.166 | 75 | 0.181 | 54 | 0.102 | 56 | 0.087 | 540 | 1.046 |
| eg2/100 | 85 | 0.036 | 363 | 0.074 | 365 | 0.154 | 158 | 0.042 | NAN | NAN |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| fletchcr/100 | 47 | 0.028 | 72 | 0.019 | 60 | 0.014 | 100 | 0.056 | 129 | 0.036 |
| fletchcr/1000 | 676 | 0.417 | 331 | 0.169 | 892 | 0.509 | 798 | 0.366 | NAN | NAN |
| genrose/100 | 972 | 0.109 | 927 | 0.086 | 939 | 0.086 | 923 | 0.087 | NAN | NAN |
| himmelbg/100 | 2 | 0.016 | 2 | 0.001 | 2 | 0.001 | 2 | 0.001 | 2 | 0.001 |
| himmelbg/1000 | 2 | 0.003 | 2 | 0.003 | 2 | 0.003 | 2 | 0.003 | 2 | 0.002 |
| liarwhd/100 | 25 | 0.025 | 31 | 0.011 | 27 | 0.013 | 33 | 0.014 | 71 | 0.016 |
| liarwhd/1000 | 45 | 0.018 | 54 | 0.030 | 46 | 0.015 | 40 | 0.014 | 118 | 0.027 |
| penalty1/100 | 22 | 0.022 | 22 | 0.007 | 14 | 0.006 | NAN | NAN | 20 | 0.013 |
| penalty1/1000 | 39 | 0.37 | 39 | 0.205 | 179 | 1.251 | NAN | NAN | NAN | NAN |
| quartc/100 | NAN | NAN | NAN | NAN | NAN | NAN | 29 | 0.008 | 30 | 0.008 |
| quartc/1000 | NAN | NAN | NAN | NAN | NAN | NAN | 48 | 0.061 | 36 | 0.065 |
| tridia/100 | 331 | 0.049 | 280 | 0.040 | 390 | 0.036 | 278 | 0.024 | 884 | 0.093 |
| tridia/1000 | 1472 | 0.15 | 1886 | 0.178 | 1459 | 0.141 | 1626 | 0.161 | NAN | NAN |
| bdexp/100 | 2 | 0.023 | 2 | 0.001 | 2 | 0.001 | NAN | NAN | 2 | 0.001 |
| bdexp/1000 | 2 | 0.006 | 2 | 0.006 | 2 | 0.006 | NAN | NAN | 2 | 0.002 |
| exdenschnf/100 | 33 | 0.027 | 32 | 0.012 | 35 | 0.009 | 28 | 0.010 | 46 | 0.008 |
| exdenschnf/1000 | 34 | 0.021 | 35 | 0.021 | 37 | 0.020 | 30 | 0.019 | 36 | 0.007 |
| Exdenschnb/100 | 28 | 0.019 | 19 | 0.004 | 15 | 0.004 | 32 | 0.010 | 25 | 0.004 |
| exdenschnb/1000 | 31 | 0.007 | 22 | 0.006 | 21 | 0.007 | 27 | 0.011 | 43 | 0.009 |
| genquartic/100 | 24 | 0.028 | 30 | 0.005 | 30 | 0.005 | 25 | 0.011 | 36 | 0.008 |
| genquartic/1000 | 28 | 0.015 | 28 | 0.016 | 24 | 0.015 | 39 | 0.031 | 220 | 0.076 |
| biggsb1/100 | 565 | 0.077 | 484 | 0.053 | 539 | 0.067 | 516 | 0.066 | 922 | 0.145 |
| sine/100 | 38 | 0.023 | 34 | 0.009 | 62 | 0.022 | NAN | NAN | NAN | NAN |
| sine/1000 | 217 | 0.079 | 513 | 0.181 | 214 | 0.102 | NAN | NAN | 160 | 0.049 |
| nonscomp/100 | 64 | 0.023 | 52 | 0.008 | 60 | 0.008 | 52 | 0.008 | NAN | NAN |
| nonscomp/1000 | 203 | 0.032 | 112 | 0.018 | NAN | NAN | 149 | 0.028 | NAN | NAN |
| power1/100 | 1431 | 0.123 | 1315 | 0.095 | 1154 | 0.089 | 1430 | 0.111 | NAN | NAN |
| raydan1/100 | 89 | 0.025 | 91 | 0.013 | 76 | 0.010 | 84 | 0.011 | 238 | 0.039 |
| raydan2/100 | 14 | 0.019 | 14 | 0.006 | 14 | 0.005 | NAN | NAN | 14 | 0.004 |
| raydan2/1000 | 18 | 0.006 | 17 | 0.006 | 17 | 0.006 | 16 | 0.007 | 31 | 0.006 |
| diagonal1/100 | 258 | 0.109 | 222 | 0.097 | 105 | 0.020 | 213 | 0.070 | NAN | NAN |
| diagonal2/100 | 190 | 0.038 | 363 | 0.039 | 669 | 0.073 | 82 | 0.010 | 271 | 0.033 |
| diagonal2/1000 | 733 | 0.186 | 1508 | 0.329 | NAN | NAN | 218 | 0.044 | 518 | 0.201 |
| diagonal3/100 | 185 | 0.091 | 136 | 0.038 | 189 | 0.078 | 190 | 0.063 | NAN | NAN |
| bv/1000 | 61 | 0.559 | 85 | 0.655 | 78 | 0.629 | 103 | 1.561 | NAN | NAN |
| ie/100 | 18 | 0.27 | 18 | 0.199 | 16 | 0.201 | 19 | 0.305 | 33 | 0.280 |
| ie/1000 | 16 | 19.543 | 15 | 19.684 | 15 | 18.057 | 21 | 25.874 | 34 | 28.277 |
| singx/100 | 119 | 0.073 | 139 | 0.087 | 136 | 0.066 | 165 | 0.142 | 234 | 0.111 |
| singx/1000 | 169 | 2.916 | 95 | 1.591 | 607 | 9.518 | 196 | 3.167 | 236 | 3.128 |
| lin/100 | 13 | 0.123 | 13 | 0.037 | 13 | 0.042 | NAN | NAN | 13 | 0.051 |
| lin/1000 | 19 | 93.679 | 19 | 89.969 | 19 | 89.849 | 39 | 258.422 | 19 | 93.106 |
| osb2/11 | 758 | 0.381 | 669 | 0.255 | 660 | 0.152 | 687 | 0.151 | NAN | NAN |
| pen1/100 | 128 | 0.09 | 212 | 0.129 | 1127 | 0.455 | NAN | NAN | NAN | NAN |
| pen1/1000 | 211 | 13.503 | 173 | 7.895 | 715 | 19.667 | 223 | 10.301 | NAN | NAN |
| pen2/100 | 638 | 1.059 | 176 | 0.165 | 251 | 0.328 | 224 | 0.223 | NAN | NAN |
| rosex/100 | 50 | 0.053 | 43 | 0.030 | 40 | 0.030 | 44 | 0.037 | 131 | 0.066 |
| rosex/1000 | 59 | 1.928 | 49 | 1.552 | 48 | 1.472 | 51 | 1.729 | 113 | 2.218 |
| trid/100 | 85 | 0.075 | 81 | 0.052 | 86 | 0.029 | 88 | 0.042 | 355 | 0.150 |
| trid/1000 | 50 | 0.746 | 51 | 0.666 | 54 | 0.838 | 49 | 0.750 | 329 | 3.337 |
| Sum | 59333 | 361.171 | 59385 | 339.236 | 68661 | 389.248 | 73738 | 596.577 | 93763 | 545.591 |

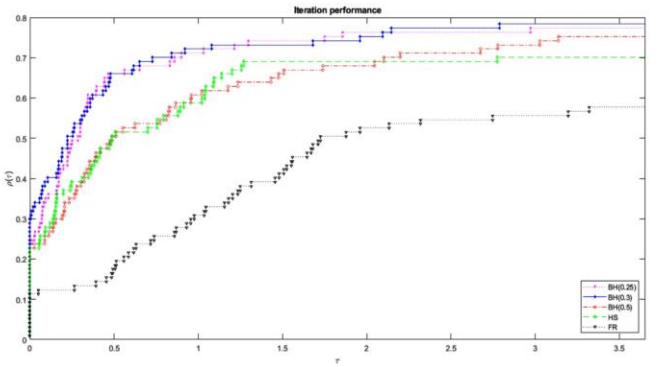



**Figure 1**
**Performance profile on iteration**

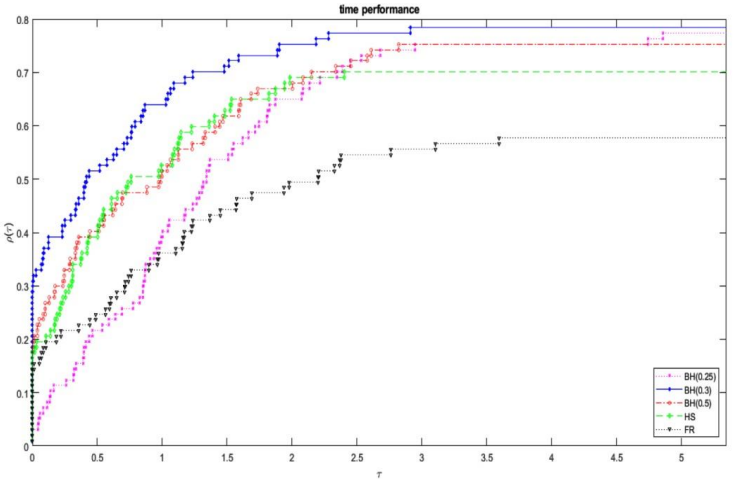



**Figure 2**
**Performance profile on time cpu.**

Table 1 and Figs. 1 and 2 show that the numerical performance of BBH (0.25), BBH(0.3) and BBH(0.5) is better than the corresponding original algorithm.

## 5. **Conclusion**

We have presented a conjugate gradient CG algorithm to solve unconstrained optimization problems. Derivation of an algorithm based on the modified categorical condition Under some assumptions, the global convergence and sufficient descent properties of this algorithm have been proven. Using some unconstrained optimization test functions, the numerical results showed that our algorithm (BH) is the most efficient compared to the HS and FR algorithms, as shown in Table 1 and Figures 1 and 2.

**References**

[1] H. M. Khudhur and A. A. M. Fawze. An improved conjugate gradient method for solving unconstrained optimisation and image restoration problems. *Int. J. Math. Model. Numer. Optim.*, vol. 13, no. 3, pp. 313-325 (2023), https://doi.org/10.1504/IJMMNO.2023.132286.

[2] Y. A. Laylani, B. A. Hassan and H. M. Khudhur. Enhanced spectral conjugate gradient methods for unconstrained optimization. *Comput. Sci.*, vol. 18, no. 2, pp. 163-172 (2023).

[3] H. N. Jabbar and B. A. Hassan. Two-versions of descent conjugate gradient methods for large-scale unconstrained optimization. *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 22, no. 3 (2021), https://doi.org/10.11591/ijeecs.v22.i3.pp1643-1649.

[4] R. Fletcher. Function minimization by conjugate gradients. *Comput. J.*, vol. 7, no. 2, pp. 149-154, Feb. (1964), https://doi.org/10.1093/comjnl/7.2.149.

[5] S. M. Kadham, "Acute interstitial pneumonia image enhancement using fuzzy partial transforms," *Applied Geomatics,* vol. 16, no. 1, pp. 35–39 (May 2023), doi: https://doi.org/10.1007/s12518-023-00509-8.

[6] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Natl. Bur. Stand.* (1934)., vol. 49, no. 6, pp. 409-436, Dec. (1952), https://doi.org/10.6028/jres.049.044.

[7] L. Mahdi, H.W. Mahdi, and Q. L. Abdulrasool, "Mechanical Properties of Waste Aggregate-Steel Fiber Reinforced Concrete Modified with Water prof Admixture," *International Journal of Civil Engineering and Technology* (IJCIET), vol. 9, no. 11, pp. 934–947, 2018, Accessed: Jan. 28 (2025).

[8] Y. H. Dai and Y. Yuan. A nonlinear conjugate gradient method with a strong global convergence property. *SIAM J. Optim.*, vol. 10, no. 1, pp. 177-182 (1999), https://doi.org/10.1137/S1052623497318992.

[9] R. Fletcher, Unconstrained Optimization Practical Methods of Optimization, vol. 1. New York:Wiley (1987).

[10] B.A. Hassan, M. Malik and I.M. Sulaiman. A variant of Dai-Yuan conjugate gradient method for unconstrained optimization and its application in portfolio selection. *J. Math. Comput. Sci.* (2021), https://doi.org/10.28919/jmcs/5798.

[11] B. A. Hassan, Z. M. Abdullah and H. N. Jabbar. A descent extension of the Dai - Yuan conjugate gradient technique. *Indonesian Journal*

*of Electrical Engineering and Computer Science*, vol. 16, no. 2 (2019), https://doi.org/10.11591/ijeecs.v16.i2.pp661-668.

[12] I. M. Sulaiman, N. A. Bakar, M. Mamat, B. A. Hassan, M. Malik and A. M. Ahmed. A new hybrid conjugate gradient algorithm for optimization models and its application to regression analysis. *Indones. J. Electr. Eng. Comput. Sci.,* vol. 23, no. 2, pp. 1100, Aug. (2021), https://doi.org/10.11591/ijeecs.v23.i2.pp1100-1109.

[13] S. Aji, P. Kumam, P. Siricharoen, A. B. Abubakar, M. M. Yahaya. A Modified Conjugate Descent Projection Method for Monotone Nonlinear Equations and Image Restoration. *IEEE Access,* vol. 8, pp. 158656-158665 (2020), https://doi.org/10.1109/ACCESS.2020.3020334.

[14] Y. Dai, J. Han, G. Liu, D. Sun, H. Yin and Y. Yuan. Convergence Properties of Nonlinear Conjugate Gradient Methods. *SIAM J. Optim.,* vol. 10, no. 2, pp. 345-358, Jan. (2000), DOI:10.1137/S1052623494268443.

[15] N. I. M. Gould, D. Orban and P. L. Toint. CUTEr and SifDec:A constrained and unconstrained testing environment, revisited. *ACM Trans. Math. Softw.,* vol. 29, no. 4, pp. 373-394, Dec. (2003), https://doi.org/10.1145/962437.962439.

[16] J. J. Moré, B. S. Garbow and K. E. Hillstrom. Testing unconstrained optimization software. *ACM Trans. Math. Softw.*, vol. 7, no. 1, pp. 17-41 (1981).

[17] R. A. Kamil, H. M. Atiea, A. A. Kadhem, Layth Abdulrasool Alasadi, and Q. A. Jabal, "Improvement of the mechanical properties of polymer-modified concrete by using PC superplasticizer," *Pollack Periodica*, vol. 18, no. 3, pp. 14–19 (Jul. 2023), doi: https://doi.org/10.1556/606.2023.00742.

[18] B. Sellami, Y. Laskri and R. Benzine. A new two-parameter family of nonlinear conjugate gradient methods. *Optimization,* vol. 64 (2015).

[19] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Math. Program. Ser. B,* vol. 91, no. 2, pp. 201-213 (2002), https://doi.org/10.1007/s101070100263.

[20] W. Gao and M. R. Farahani. The hyper-zagreb index for an infinite family of nanostar dendrimer," *J. Discrete Math. Sci. Cryptography,* vol. 20, no. 2, pp. 2017, https://doi.org/10.1080/09720529.2016.1220088.

[21] M. Imran, A. A. E. Abunamous, D. Adi, S. H. Rafique, A. Q. Baig and M. R. Farahani. Eccentricity based topological indices of honeycomb networks.," *J. Discrete Math. Sci. Cryptography*, vol. 22, no. 7, pp. 2019, https://doi.org/10.1080/09720529.2019.1691326.

[22] W. Gao and M. R. Farahani. The hyper-zagreb index for an infinite family of nanostar dendrimer," *J. Discrete Math. Sci. Cryptography*, vol. 20, no. 2, pp. 2017, https://doi.org/10.1080/09720529.2016.1220088.

[23] M. Imran, A. A. E. Abunamous, D. Adi, S. H. Rafique, A. Q. Baig and M. R. Farahani. Eccentricity based topological indices of honeycomb networks.," *J. Discrete Math. Sci. Cryptography,* vol. 22, no. 7, pp. 2019, https://doi.org/10.1080/09720529.2019.1691326.

[24] W. Gao, L. Shi and M. R. Farahani. Szeged Related Indices of TUAC6[p,q]," *J. Discrete Math. Sci. Cryptography,* vol. 20, no. 2, pp. 2017, https://doi.org/10.1080/09720529.2016.1228312.

[25] S. Ahmad, H. M. A. Siddiqui, A. Ali, M. R. Farahani, M. Imran and I. N. Cangul. On Wiener index and Wiener polarity index of some polyomino chains," *J. Discrete Math. Sci. Cryptography,* vol. 22, no. 7, pp. 2019, https://doi.org/10.1080/09720529.2019.1688965.

[26] M. Nadeem, S. Ahmad, M. K. Siddiqui, M. A. Ali, M. R. Farahani and A. J. M. Khalaf. On some applications related with algebraic structures through different well known graphs," *J. Discrete Math. Sci. Cryptography,* vol. 24, no. 2, pp. 2021, https://doi.org/10.1080/09720529.2021.1885806.

*Received June, 2024*