A VARIANT OF DAI-YUAN CONJUGATE GRADIENT METHOD FOR UNCONSTRAINED OPTIMIZATION AND ITS APPLICATION IN PORTFOLIO SELECTION

BASIM A. HASSAN1, MAULANA MALIK2,∗, IBRAHIM MOHAMMED SULAIMAN3

1Department of Mathematics, University of Mosul, Mosul 41002, Iraq
2Department of Mathematics, Universitas Indonesia, Depok 16424, Indonesia
3Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Terengganu 22200, Malaysia

Abstract. The quasi-Newton (QN) method are among the efficient variants of conjugate gradient (CG) method for solving unconstrained optimization problems. The QN method utilizes the gradients of the function while ignoring the available value information at every iteration. In this paper, we extended the Dai-Yuan [39] coefficient in designing a new CG method for large-scale unconstrained optimization problems. An interesting feature of our method is that its algorithm not only uses the available gradient value, but also consider the function value information. The global convergence of the proposed method was established under some suitable Wolfe conditions. Extensive numerical computation have been carried out which show that the average performance of this new algorithm is efficient and promising. In addition, the proposed method was extended to solve practical application problem of portfolio selection.

Keywords: conjugate gradient parameter; convergence analysis; optimization models; line search procedures.

2010 AMS Subject Classification: 65K10, 49M37, 90C06.

*Corresponding author
E-mail address: m.malik@sci.ui.ac.id
Received April 1, 2021
1. Introduction

The nonlinear conjugate gradient (CG) algorithms are among the efficient numerical algorithms for solving unconstrained optimization problems, especially, when the problems are of large dimension. The CG method are very popular among mathematicians, engineers, and many more because of its robustness and ability to solve large-scale optimization problems [22]. Consider an unconstrained optimization model

\[
\min_{x \in \mathbb{R}^n} f(x),
\]

where \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) is a smooth function and \( g \) denotes the gradient of \( f \). The CG algorithm generate a sequence of iterate \( \{x_k\} \) via the following recurrence formula:

\[
x_{k+1} = x_k + s_k,
\]

where \( k \geq 0, s_k = \alpha_k d_k \) [23]. The parameter \( \alpha_k > 0 \) is known as the step-size which is often computed along the search direction \( d_k \) with formula defined as

\[
d_k := \begin{cases} 
- g_k, & k = 0 \\
- g_k + \beta_k d_{k-1}, & k \geq 1 
\end{cases},
\]

where \( \beta_k \) represent the conjugate gradient parameter that characterize different CG methods. The classical formula for \( \beta_k \) are group into two. The first group include HS method [27], PRP method [9, 11], and LS method [37] with formula given as:

\[
\beta_{k}^{HS} = \frac{g_k^T(g_k - g_{k-1})}{d_{k-1}^T(g_k - g_{k-1})},
\]

\[
\beta_{k}^{PRP} = \frac{g_k^T(g_k - g_{k-1})}{\|g_{k-1}\|^2},
\]

\[
\beta_{k}^{LS} = -\frac{g_k^T(g_k - g_{k-1})}{d_{k-1}^T g_{k-1}}.
\]

This group is characterize by their restart properties and efficient numerical performance [21, 38]. Restart strategy is usually employed in conjugate gradient algorithms to improve their computational efficiency. However, the convergence of most of these methods are yet to be established under some line search conditions [24, 35]. The second group include the FR method [31], the CD method [30], and the DY method [39] with formula given as:
On the other hand, these methods FR, CD, and DY do not possess the restart strategy and thus perform poorly due to jamming phenomenon [3]. However, the convergence of these methods have been established under various line search methods [24, 35]. One of the most frequent used line search method is the inexact line search, particularly the Wolfe line search method [18]. For the Wolfe line search, the step-size $\alpha_k$ is computed such that:

\begin{align}
 f(x_k + \alpha_k d_k) &\leq f(x_k) + \phi \alpha_k g_k^T d_k, \\
 g(x_k + \alpha_k d_k)^T d_k &\geq \sigma g_k^T d_k,
\end{align}

where $0 < \phi < \sigma < 1$ [32]. Numerous researcher have studied the conjugate gradient method under the strong Wolfe line search [16]. For more references on advances in conjugate gradient method (see, [1, 4, 5, 6, 7, 8, 15, 17, 25, 26, 36]).

Motivated by the method of Dai and Yuan [39], we propose a modification of the conjugate gradient coefficient for solving unconstrained optimization models. The global convergence of the method is established under some mild conditions. Furthermore, the method was extended to solve practical application problem of portfolio selection.

The rest sections of this paper is structured as follows: In section 2, we present the derivation process of the new our method and its algorithm. The convergence result of the proposed method is discussed in section 3. We report preliminary results of the numerical computation carried out on some benchmark test problems in section 4. An application problem of portfolio selection was discussed in section 5. Lastly, section 6 present the conclusion of the paper.
2. A NEW CONJUGATE GRADIENT METHOD

Ataee et al. [34] a unified quasi-Newton equations those presented by several authors (see [12, 19, 41]), as follows:

\[ B_{k+1}s_k = y_k + \vartheta \frac{2(f_k - f_{k+1}) + (g_{k+1} + g_k)^T s_k}{s_k^T \mu_k} \mu_k, \]

where \( \vartheta \in [0, 1, 2, 3] \) and \( \mu_k \) is any vector satisfying \( s_k^T \mu_k \neq 0 \).

Recently, Razieh et al. [29] defined the relation as:

\[ f_{k+1} - f_k = \int_0^1 \nabla f(x_k + ts_k) dt s_k \approx s_k^T g_k, \]

from inspired if \( ||s_k|| \) is small.

By using the unified quasi-Newton equation and above relation we derive a new coefficient conjugate gradient.

Multiplying both side equation (2) by \( s_k^T \), we have:

\[
\begin{align*}
    s_k^T B_{k+1} s_k &= s_k^T y_k + \vartheta 2(f_k - f_{k+1}) + \vartheta (g_{k+1} + g_k)^T s_k \\
    &= s_k^T y_k - \vartheta 2s_k^T g_k + \vartheta (g_{k+1} + g_k)^T s_k \\
    &= s_k^T y_k - \vartheta 2s_k^T g_k + \vartheta s_k^T g_{k+1} + \vartheta s_k^T g_k \\
    &= s_k^T y_k + \vartheta s_k^T g_{k+1} - \vartheta s_k^T g_k \\
    &= s_k^T y_k + \vartheta s_k^T y_k.
\end{align*}
\]

From above equation, we get:

\[ d_k^T B_{k+1} s_k = d_k^T y_k + \vartheta d_k^T y_k = (1 + \vartheta) d_k^T y_k \]

On the other hand, by using conjugacy condition, we obtain:

\[ d_k^T Gd_k = (-g_{k+1} + \beta_k d_k)^T Gd_k = -g_{k+1}^T Gd_k + \beta_k d_k^T Gd_k = 0, \]

where \( G \) is Hessian matrix. As a result,

\[ \beta_k = \frac{g_{k+1}^T Gs_k}{d_k^T Gs_k}. \]
Putting (5) in (6), which yields:

$$
\beta_k = \frac{y_k^T g_{k+1}}{(1 + \vartheta)d_k^Ty_k}.
$$

To create algorithms that have global convergence properties, we modify the above formula as follows:

$$
\beta_{BMS}^k = \frac{\|g_{k+1}\|^2}{(1 + \vartheta)d_k^Ty_k},
$$

where $\vartheta \in [0, 1, 2, 3]$ and $y_k = g_{k+1} - g_k$.

A complete algorithm of BMS method could be generated as follows:

**Algorithm 2.1. (BMS Method)**

1. Given an initial point $x_0 \in \mathbb{R}^n$, stopping criteria $\varepsilon = 10^{-6}$, parameter $\sigma = 0.001, \varphi = 0.0001$ and $\vartheta = 1$.
2. Calculate $\|g_k\|$, if $\|g_k\| \leq \varepsilon$ then stop, $x_k$ is optimal point. Else, go to Step 3.
3. Calculate $\beta_k$ using (7).
4. Calculate search direction $d_k$ (2).
5. Calculate step length $\alpha_k$ using the Wolfe line search (3) and (4).
6. Set $k := k + 1$, calculate the next iteration $x_{k+1}$ using (1), and go to Step 2.

**3. Convergence Analysis**

This section, we prove the global convergence of new Algorithm under the following assumption, which has often been used in the convergence analysis of conjugate gradient methods.

**Assumption 3.1.** (A1) The level set $L = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$ is bounded. (A2) In some neighborhood $U$ of $L$, $f(x)$ is continuously differentiable and its gradient is Lipschitz continuous, namely, there exists a constant $L > 0$ such that

$$
\|g(x_{k+1}) - g(x_k)\| \leq L\|x_{k+1} - x_k\|, \forall x_{k+1}, x_k \in U
$$

**Theorem 3.2.** Let $\{x_{k+1}\}$ is obtained as new Algorithm, then, we have $d^T_{k+1}g_{k+1} \leq 0$ for all $k.$
Proof. Since \( d_0 = -g_0 \), we obtained \( g_0^T d_0 = -\|g_0\|^2 \leq 0 \). Suppose that \( d_k^T g_k < 0 \). In [39], it follows from the definition of the direction generated by the Dai-Yuan (DY) method as:

\[
\beta_{k}^{DY} = \frac{g_{k+1}^T d_{k+1}}{g_k^T d_k}.
\]

By the using the new formula, we have:

\[
\beta_{k}^{BMS} = \frac{\|g_{k+1}\|^2}{(1 + \vartheta)d_k^T y_k} = \frac{1}{1 + \vartheta} \beta_{k}^{DY}.
\]

From (9) and (10), we obtained:

\[
\beta_{k}^{BMS} = \frac{1}{1 + \vartheta} \frac{g_{k+1}^T d_{k+1}}{g_k^T d_k}.
\]

The above relation can be rewritten as:

\[
g_{k+1}^T d_{k+1} = (1 + \vartheta) \beta_{k}^{BMS} g_k^T d_k
\]

Since \((1 + \vartheta)\) and \( \beta_{k}^{DY} \) are positive then \( \beta_{k}^{BMS} \) is always positive, now (11), this yields :

\[
g_{k+1}^T d_{k+1} = (1 + \vartheta) \beta_{k}^{BMS} g_k^T d_k < 0.
\]

This finishes the proof. \( \square \)

The formula (10) is very important in our convergence analysis. Due to playing an important role in analyzing the convergence property for conjugate gradient, Zoutendijk’s condition [13] will be proved to be a part of the proposed Algorithm in this study.

**Lemma 3.3.** Let that \( d_{k+1} \) is generated by (2) and step size \( \alpha_K \) fulfills (3) and (4), if \( f(x) \) satisfies the Assumption 3.1, then :

\[
\sum_{k=1}^{\infty} \frac{(g_{k+1}^T d_{k+1})^2}{\|d_{k+1}\|^2} < \infty
\]

holds.

**Theorem 3.4.** Suppose that assumptions holes. Let \( \{x_k\} \) be generated by Algorithm, where the step length satisfies the Wolfe line search conditions. Then :

\[
\lim_{k \to \infty} \inf \|g_k\| = 0.
\]
Proof. By contradiction, we suppose that the conclusion is not true. Then there exists a constant \( \xi > 0 \) such that \( \|g_{k+1}\|^2 \geq \xi^2 \). From search direction, it follows that \( d_{k+1} + g_{k+1} = \beta_k d_k \). Implies that:

\[
\|d_{k+1}\|^2 + \|g_{k+1}\|^2 + 2d_{k+1}^Tg_{k+1} = (\beta_k)^2 \|d_k\|^2
\]

Dividing both sides of this inequality by \( (d_{k+1}^Tg_{k+1})^2 \), that:

\[
\frac{\|d_{k+1}\|^2}{(d_{k+1}^Tg_{k+1})^2} = -\frac{2}{d_{k+1}^Tg_{k+1}} - \|g_{k+1}\|^2 + (\beta_k)^2 \|d_k\|^2 \\
= -\left( \|g_{k+1}\| + \frac{1}{\|g_{k+1}\|} \right)^2 + \frac{1}{\|g_{k+1}\|^2} + \frac{1}{(1 + \vartheta)} \left( \frac{d_{k+1}^Tg_{k+1}}{d_k^Tg_k} \right)^2 \\
\leq \frac{1}{\|g_{k+1}\|^2} + \frac{1}{(1 + \vartheta)} \left( \frac{d_{k+1}^Tg_{k+1}}{d_k^Tg_k} \right)^2
\]

Since \( \vartheta \in [0, 1, 2, 3] \), then \( \frac{1}{1 + \vartheta} < 1 \). Hence, we obtain:

\[
\frac{\|d_{k+1}\|^2}{(d_{k+1}^Tg_{k+1})^2} \leq \frac{1}{\|g_{k+1}\|^2} + \frac{\|d_k\|^2}{(d_k^Tg_k)^2}.
\]

The above inequality implies

\[
\frac{\|d_{k+1}\|^2}{(d_{k+1}^Tg_{k+1})^2} \leq \sum_{i=1}^{k+1} \frac{1}{\|g_k\|^2}
\]

Thus,

\[
\frac{\|d_{k+1}\|^2}{(d_{k+1}^Tg_{k+1})^2} < \frac{k+1}{\xi^2}
\]

this implies that

\[
\sum_{k=1}^{\infty} \frac{(g_k^Td_k)^2}{\|d_k\|^2} = \infty.
\]

This contradicts Lemma 3.3. Therefore, (13) holds. \( \square \)

4. Numerical Experiments

In this section, we report the numerical results of a comparison between the proposed method (denoted by BMS) and RMIL+ method [40]. Our experiments have been used 49 test functions selected from Andrei [28] and Jamil-Yang [20], as listed in Table 1, with variation dimensions from 2 to 50,000. Attempts to complete each test function are limited to the termination criterion \( \|g_k\| \leq 10^{-6} \) or to a maximum of 10,000 iterations. All the algorithms are coded in
MATLAB R2019a by using a personal laptop with specifications; processor Intel Core i7, 16 GB RAM, Windows 10 Pro 64 bit and the algorithms implemented the Wolfe line search conditions with $\sigma = 0.001$ and $\varphi = 0.0001$.

**Table 1. List of Test Functions.**

| No | Function                          | No | Function            |
|----|-----------------------------------|----|---------------------|
| F1 | Extended White & Holst            | F26| POWER               |
| F2 | Extended Rosenbrock               | F27| Quadratic QF1       |
| F3 | Extended Freudenstein & Roth      | F28| Quartic             |
| F4 | Extended Beale                    | F29| Matyas              |
| F5 | Raydan 1                          | F30| Colville            |
| F6 | Extended Tridiagonal 1            | F31| Dixon and Price     |
| F7 | Diagonal 4                        | F32| Sphere              |
| F8 | Extended Himmelblau               | F33| Sum Squares         |
| F9 | FLETCHCR                          | F34| DENSCHNA            |
| F10| NONSCOMP                          | F35| DENSCHNF            |
| F11| DENSCHNB                          | F36| Staircase S1        |
| F12| Extended Penalty                  | F37| Staircase S2        |
| F13| Hager                             | F38| Staircase S3        |
| F14| BIGGSB1                           | F39| Extended Block-Diagonal BD1 |
| F15| Extended Maratos                  | F40| HIMMELBH            |
| F16| Six Hump Camel                    | F41| Tridiagonal White and Holst |
| F17| Three Hump Camel                  | F42| ENGVAL1             |
| F18| Booth                             | F43| Linear Perturbed    |
| F19| Trecanni                          | F44| QUARTICCM           |
| F20| Zettl                             | F45| Brent               |
| F21| Shallow                           | F46| Deckkers-Aarts      |
| F22| Generalized Quartic               | F47| El-Attar-Vidyasagar-Dutta |

*(Continued on next page)*
Table 1 – Continued

| No | Function                  | No  | Function                  |
|----|---------------------------|-----|---------------------------|
| F23| Quadratic QF2             | F48 | Rotated Ellipse 2         |
| F24| Generalized Tridiagonal 1 | F49 | Zirilli or Aluffi-Pentini’s|
| F25| Generalized Tridiagonal 2 |     |                           |

TABLE 2. Numerical Results.

| Functions | Dimensions | Initial Points | BMS | RMIL+ |
|-----------|------------|----------------|-----|-------|
|           |            |                | NoI | NOF   | CPU  |       |
|           |            |                |     |       |      |       |
| F1        | 1,000      | (-1.2,1,-1.2,1)| -   | -     | 16   | 102  0.0582 |
| F1        | 10,000     | (-1.2,1,-1.2,1)| -   | -     | 16   | 102  0.3839 |
| F2        | 1,000      | (-1.2,1,-1.2,1)| 6574| 20205 | 5.0919| 27   | 176  0.0401 |
| F2        | 10,000     | (-1.2,1,-1.2,1)| 7056| 21651 | 51.639| 32   | 192  0.3618 |
| F3        | 10         | (0.5,2,0.5,-2) | -   | -     | 12   | 62   0.0155 |
| F3        | 100        | (0.5,-2,0.5,-2)| -   | -     | -    | -    |       |
| F4        | 1,000      | (1,0.8,1,0.8)  | 425 | 1326  | 0.6103| 52   | 191  0.0889 |
| F4        | 10,000     | (1,0.8,1,0.8)  | 458 | 1425  | 6.4805| 54   | 199  1.0128 |
| F5        | 10         | (1,....1)      | 31  | 143   | 0.0318| 27   | 105  0.0027 |
| F5        | 100        | (1,....1)      | 218 | 1211  | 0.0549| 102  | 629  0.0331 |
| F6        | 500        | (2,...,2)      | 9883| 23068 | 6.1981| 6    | 37   0.0117 |
| F6        | 1,000      | (2,...,2)      | -   | -     | 7    | 40   0.0364 |
| F7        | 500        | (1,...,1)      | 187 | 560   | 0.1136| -    | -    |       |
| F7        | 1,000      | (1,...,1)      | 193 | 578   | 0.1435| -    | -    |       |
| F8        | 1,000      | (1,...,1)      | 21  | 74    | 0.0318| 11   | 44   0.015 |
| F8        | 10,000     | (1,...,1)      | 22  | 77    | 0.1625| 12   | 47   0.1184 |
| F9        | 10         | (0,...,0)      | 142 | 525   | 0.1252| 72   | 311  0.0052 |
| F9        | 100        | (0,...,0)      | 7508| 23301 | 1.0162| 3030 | 9841 | 0.4331 |
| F10       | 5          | (3,...,3)      | 226 | 693   | 0.0167| -    | -    |       |
| F10       | 9          | (3,...,3)      | -   | -     | -    | -    | -    |       |
| F11       | 1,000      | (10,...,10)    | 16  | 67    | 0.0154| 8    | 37   0.0145 |
| F11       | 10,000     | (10,10)        | 16  | 67    | 0.1473| 7    | 34   0.0887 |
| F12       | 10         | (1,...,10)     | 50  | 168   | 0.0049| 27   | 112  0.0047 |
| F12       | 100        | (1,...,100)    | 279 | 938   | 0.0455| 30   | 137  0.0236 |
| F13       | 50         | (1,...,1)      | 28  | 128   | 0.0297| 20   | 91   0.0052 |
| F13       | 100        | (1,...,1)      | 37  | 217   | 0.1116| 25   | 138  0.0142 |
| F14       | 3          | (0.1,...,0.1)  | 1   | 3     | 0.002 | 1    | 3    0.0117 |
| F14       | 3          | (1,...,1)      | 0   | 0     | 2.79E-04| 0 | 0  0.0053 |
| F15       | 10         | (1.1,0.1,1.1,0.1)| 1773| 5628  | 0.0853| 207  | 923  0.0204 |
| F15       | 50         | (1.1,0.1,1.1,0.1)| -   | -     | 48   | 292  0.0163 |
| F16       | 2          | (-1,2)         | 12  | 48    | 0.0014| 8    | 36   8.67E-04 |

(Continued on next page)
Table 2 – Continued

| Functions | Dimensions | Initial Points | NOI | NOF | CPU   | NOI | NOF | CPU   |
|-----------|------------|----------------|-----|-----|-------|-----|-----|-------|
| F16       | 2          | (-5,10)        | 9   | 46  | 5.56E-04 | 11  | 66  | 0.0068 |
| F17       | 2          | (0.5,0.5)      | 21  | 398 | 0.0039  | -   | -   | -     |
| F17       | 2          | (0.5,0)        | 23  | 434 | 0.0049  | -   | -   | -     |
| F18       | 2          | (5.5)          | 16  | 48  | 0.0016  | 2   | 6   | 1.99E-04 |
| F18       | 2          | (10,10)        | 7   | 21  | 3.18E-04 | 2   | 6   | 0.0026 |
| F19       | 2          | (-1,0.5)       | 1   | 3   | 4.52E-04 | 1   | 3   | 1.15E-04 |
| F19       | 2          | (-5,10)        | 9   | 36  | 8.78E-04 | 5   | 23  | 0.0086 |
| F20       | 2          | (-1.2)         | 84  | 262 | 0.0071  | 16  | 69  | 0.0038 |
| F20       | 2          | (10,10)        | 43  | 134 | 0.0033  | -   | -   | -     |
| F21       | 1,000      | (2,...,2)      | 99  | 307 | 0.2032  | 8   | 39  | 0.0146 |
| F21       | 5,000      | (2,...,2)      | 104 | 322 | 0.2904  | 8   | 39  | 0.0535 |
| F22       | 1,000      | (-0.5,...,-0.5) | 507 | 8992 | 1.3872 | -   | -   | -     |
| F22       | 7,000      | (-0.5,...,-0.5) | -   | -   | -     | -   | -   | -     |
| F23       | 50         | (0.5,...,0.5)  | 157 | 536 | 0.0234  | 78  | 280 | 0.0148 |
| F23       | 500        | (0.5,...,0.5)  | 1391| 4695| 0.6698  | 581 | 2031| 0.2735 |
| F24       | 10         | (2,...,2)      | 31  | 102 | 0.0065  | 22  | 74  | 0.0031 |
| F24       | 100        | (2,...,2)      | 32  | 116 | 0.0136  | 23  | 78  | 0.0162 |
| F25       | 4          | (1,...,1)      | 14  | 38  | 0.0035  | 7   | 21  | 0.001 |
| F25       | 500        | (1,...,1)      | 59  | 350 | 0.054   | 34  | 190 | 0.047 |
| F26       | 10         | (1,...,1)      | 278 | 834 | 0.0137  | 123 | 369 | 0.0114 |
| F26       | 100        | (1,...,1)      | -   | -   | -     | -   | -   | -     |
| F27       | 50         | (1,...,1)      | 143 | 429 | 0.0148  | 69  | 207 | 0.0065 |
| F27       | 500        | (1,...,1)      | 1241| 3723| 0.4803  | 447 | 1341| 0.1815 |
| F28       | 4          | (20,20,20,20)  | 2404| 8112| 0.3912  | 803 | 2809| 0.0523 |
| F28       | 4          | (1,1,1,1)      | 2372| 7827| 0.1092  | 766 | 2532| 0.0679 |
| F29       | 2          | (1,1)          | 7   | 49  | 8.77E-04 | -   | -   | -     |
| F29       | 2          | (20,20)        | 9   | 63  | 0.0011  | -   | -   | -     |
| F30       | 4          | (2,2,2,2)      | 3845| 12774| 0.1636 | 1032| 4339| 0.0562 |
| F30       | 4          | (10,10,10,10)  | 5991| 21091| 0.2244 | 669 | 2819| 0.1592 |
| F31       | 3          | (1,1,1)        | 73  | 232 | 0.0057  | -   | -   | -     |
| F31       | 3          | (2,2,2)        | 96  | 329 | 0.0388  | -   | -   | -     |
| F32       | 100        | (1,...,1)      | 1   | 3   | 3.15E-04 | 1   | 3   | 7.10E-04 |
| F32       | 5,000      | (1,...,1)      | 1   | 3   | 0.0061  | 1   | 3   | 0.0123 |
| F33       | 50         | (0,1,0,1)      | 128 | 384 | 0.6765  | 46  | 138 | 0.0111 |
| F33       | 5,000      | (0,1,0,1)      | -   | -   | -     | 2659| 7977| 6.8666 |
| F34       | 10,000     | (7,...,7)      | 52  | 345 | 1.4298  | 12  | 66  | 0.2259 |
| F34       | 50,000     | (7,...,7)      | 54  | 351 | 5.3625  | 12  | 66  | 0.9863 |
| F35       | 5,000      | (100,-100)     | 24  | 140 | 0.12    | 11  | 91  | 0.0832 |
| F35       | 10,000     | (100,-100)     | 24  | 140 | 0.4653  | 11  | 91  | 0.197 |
| F36       | 2          | (1,1)          | 4   | 207 | 9.59E-04 | 4   | 208 | 0.0019 |
| F36       | 2          | (-1,-1)        | 4   | 208 | 0.0049  | 4   | 208 | 0.0086 |

(Continued on next page)
Numerical results are provided in Table 2 in the form number of iterations (NOI), number of function evaluations (NOF), and central processing unit (CPU) time. We symbolizes '-' if the NOI of methods exceeds 10,000 or never reaches the optimal value. We also use a performance profile suggested by Dolan and Moré [10], to describe the performance profile of the BMS and RMIL+ methods on NOI, NOF and CPU time. Suppose that $V$ is a set of solvers with $n_s$ solvers and $M$ is a set of problems set with $n_m$ test problems. For each problem $m \in M$ and solver $s \in V$, we denotes $c_{m,s}$ as NOI or NOF or CPU required to solve problem $m \in M$ by solver $v \in V$. Then comparison of the solvers is defined as follows:

$$z_{m,s} = \frac{c_{m,s}}{\min\{c_{m,s} : m \in M \text{ and } s \in V\}}.$$
Thus, the overall performance appraisal of the solver is obtained from the performance profile function given by

$$\rho_s(\tau) = \frac{1}{n_m} \text{size}\{m : 1 \leq m \leq n_m, \log_2 z_{m,s} \leq \tau\},$$

where $\tau \geq 0$.

**Figure 1.** Performance profiles based on NOI.

**Figure 2.** Performance profiles based on NOF.
According to performance profiles are plotted in Figs. 1-3, where Fig. 1 shows the performance profiles based on NOI, Fig. 2 shows the performance profiles based on NOF and lastly Fig. 3 is the performance profiles based on CPU time. From the all figures we can see that the BMS method is very competitive with RMIL+ conjugate gradient method.

5. **APPLICATION IN PORTFOLIO SELECTION**

Portfolio selection plays an important role in financial mathematics, risk management and economics. Portfolio selection is useful for assessing the combination of available alternative securities. It aims to maximize the investment return of investors which can be done by maximizing return or minimizing risk [14].

In this article, we only consider a securities of stock and choose four blue chip stocks in Indonesia, namely, PT Bank Central Asia Tbk (BBCA.JK), PT Ace Hardware Indonesia Tbk (ACES.JK), PT Adaro Energy Tbk (ADRO.JK) and PT Gudang Garam Tbk (GGRM.JK). We use the weekly closing price which taken from http://finance.yahoo.com, with a period of three years (Jan 1, 2018-Dec 31, 2020). For return ($T_i$), expected of return ($E(T_i)$), and variance of
TABLE 3. Mean and variance of return for four Stocks

| Stocks | BBCA  | ACES  | ADRO  | GGRM  |
|--------|-------|-------|-------|-------|
| Mean   | -0.00204 | -0.00072 | 0.00497 | 0.00577 |
| Variance | 0.00134 | 0.00264 | 0.00594 | 0.00222 |

TABLE 4. Covariance of return for four stocks

| Covariance | BBCA  | ACES  | ADRO  | GGRM  |
|------------|-------|-------|-------|-------|
| BBCA       | 0.00134 | 0.00071 | 0.00132 | 0.00064 |
| ACES       | 0.00071 | 0.00266 | 0.00115 | 0.00051 |
| ADRO       | 0.00132 | 0.00115 | 0.00597 | 0.00103 |
| GGRM       | 0.00064 | 0.00051 | 0.00103 | 0.00224 |

The return ($\sigma_i^2$) of each stock can be obtained by formula as follows:

\[ T_t = \frac{I_t - I_{t-1}}{I_{t-1}}, \]
\[ E(T_i) = \frac{\sum_{t=1}^{n} T_t}{n}, \]
\[ \sigma_i^2 = Var(T_i) = \frac{\sum_{t=1}^{n} (T_t - E(T_i))^2}{n-1}, \]

where $T_t$ is a stock return in period $t$, $I_t$ is a closing stock price in period $t$, $I_{t-1}$ is a closing stock price one period before $t$ and $N$ is number of observation periods.

By using data in [http://finance.yahoo.com](http://finance.yahoo.com), (15), (16) and (17), we get the mean, variance and covariance of each return stock as in Tables 3 and 4.
Since our portfolio consists four stocks then we have formula for expected of return and variance of return of portfolio as follows [33]:

\[ \mu = E \left( \sum_{i=1}^{4} b_i T_i \right) = \sum_{i=1}^{4} b_i E(T_i), \]

\[ \sigma^2 = \text{Var} \left( \sum_{i=1}^{4} b_i T_i \right) = \sum_{i=1}^{4} \sum_{j=1}^{4} b_i b_j \text{Cov}(T_i, T_j), \]

where \( b_1, b_2, b_3, b_4 \) are proportion of the BBCA, ACES, ADRO and GGRM stocks respectively and \( \text{Cov}(T_i, T_j) \) is the covariance of return between two stocks. Since what we want here is risk avoidance, therefore, we want a small variance of the return (i.e. low risk), so that our problem about portfolio selection can be written by

\[
\begin{aligned}
\text{minimize} : \sigma^2 &= \sum_{i=1}^{4} \sum_{j=1}^{4} b_i b_j \text{Cov}(T_i, T_j). \\
\text{subject to} : \sum_{i=1}^{4} b_i &= 1.
\end{aligned}
\]

Suppose \( b_4 = 1 - b_1 - b_2 - b_3 \) and by using values in Table 4 then the problem (20) will become an unconstrained optimization problem as follows:

\[
\min_{(b_1, b_2, b_3) \in \mathbb{R}^3} \left[ (0.70e - 3b_1 + 0.7e - 4b_2 + 0.68e - 3b_3 + 0.64e - 3)b_1 + (0.20e - 3b_1 \\
+ 0.215e - 2b_2 + 0.64e - 3b_3 + 0.51e - 3)b_2 + (0.29e - 3b_1 + 0.12e - 3b_2 \\
+ 0.494e - 2b_3 + 0.103e - 2)b_3 + (-0.160e - 2b_1 - 0.173e - 2b_2 \\
- 0.121e - 2b_3 + 0.224e - 2)(1 - b_1 - b_2 - b_3) \right]
\]

The next step we solve the above problem by using BMS method with randomly initial point \( (b_1, b_2, b_3) = (0.3, 0.3, 0.4) \), thus we will get \( b_1 = 0.57, b_2 = 0.19, b_3 = -0.03 \) and \( b_4 = 0.27 \).

Furthermore, based on Table 3, Table 4, (18) and (19), we have \( \mu = 0.0001 \) and \( \sigma^2 = 0.001 \). Hence, the weight of the proportion of each stock that makes up the optimal portfolio with minimal risk is 57% for BBCA, 19% ACES, −3% ADRO and 27% GGRM with expected portfolio return is 0.0001 and the portfolio risk is 0.001. In this case, investor is allowed to do short selling as in ADRO stock. Another consideration regarding the application of the CG method in portfolio selection can be seen in [2].
6. Conclusion

The conjugate gradient methods have recently been explored by many researchers. This is due to their nice convergence properties, low memory requirement, efficient numerical result, in addition to real-life practical application. In this paper, we have derive a new conjugate gradient parameter for unconstrained optimization problems. The global convergence properties of the proposed method is established under some mild conditions. An interesting feature of our method is the ability to reduce to the classical DY method. Numerical results have been presented to illustrate the performance of the method especially for the large-scale problems. The proposed method was further extended to solve real-life application problem of portfolio selection.

Conflict of Interests

The author(s) declare that there is no conflict of interests.

References

[1] A.B. Abubakar, P. Kumam, H. Mohammad, A.M. Awwal, An efficient conjugate gradient method for convex constrained monotone nonlinear equations with applications, Mathematics, 7 (9) (2019), 767.

[2] A.B. Abubakar, P. Kumam, M. Malik, P. Chaipunya, A.H. Ibrahim, A hybrid FR-DY conjugate gradient algorithm for unconstrained optimization with application in portfolio selection, AIMS Math. 6 (6) (2021), 6506-6527.

[3] A.O. Umar, I.M. Sulaiman, M. Mamat, M.Y. Waziri, H.M. Foziah, A.J. Rindengan, D.T. Salaki, New hybrid conjugate gradient method for solving fuzzy nonlinear equations, J. Adv. Res. Dyn. Control Syst. 12 (2) (2020), 1585-590.

[4] B.A. Hassan, A modified quasi-Newton methods for unconstrained Optimization, J. Pure Appl. Math. 2019 (42) (2019), 504-511.

[5] B.A. Hassan, A new type of quasi-newton updating formulas based on the new quasi-newton equation, Numer. Algebra Control Optim. 10 (2020), 227–235.

[6] B.A. Hassan, H.N. Jabbar, A New Transformed Biggs ’s Self-Scaling Quasi-Newton Method for Optimization, ZANCO J. Pure Appl. Sci. 31 (2018), 1-5.

[7] B. A. Hassan, W.T. Mohammed, A New Variants of Quasi-Newton Equation Based on the Quadratic Function for Unconstrained Optimization, Indonesian J. Electric. Eng. Computer Sci. 19 (2) (2020), 701-708.
[8] B. Baluch, Z. Salleh, A. Alhawarat, U.A.M. Roslan, A new modified three-term conjugate gradient method with sufficient descent property and its global convergence, J. Math. 2017 (2017), 2715854.

[9] B.T. Polyak, The conjugate gradient method in extremal problems, USSR Comput. Math. Math. Phys. 9 (4) (1969), 94-112.

[10] E.D. Dolan, J.J. Moré, Benchmarking optimization software with performance profiles, Math. Program. 91 (2) (2002), 201-213.

[11] E. Polak, G. Ribiere, Note sur la convergence de méthodes de directions conjuguées, ESAIM: Math. Model. Numer. Anal.-Mod. Math. Anal. Numér. 3 (R1) (1969), 35-43.

[12] F. Biglari, M.A. Hassan, W.J. Leong, New quasi-Newton methods via higher order tensor models, J. Comput. Appl. Math. 235 (8) (2011), 2412-2422.

[13] G. Zoutendijk, Nonlinear programming, computational methods, in: Integer and Nonlinear Programming, ed. J. Abadie North-Holland, Amsterdam, (1970), pp. 37–86.

[14] H.M. Markowitz, Portfolio selection, J. Finance, 7 (1) (1952), 77-91.

[15] I.M. Sulaiman, M. Mamat, M.Y. Waziri, U.A. Yakubu, M. Malik, The performance analysis of a new modification of conjugate gradient parameter for unconstrained optimization models, Math. Stat. 9 (1) (2021), 16-23.

[16] I.M. Sulaiman, M. Mamat, A.E. Owoyemi, P.L. Ghazali, M. Rivaie, M. Malik, The convergence properties of some descent conjugate gradient algorithms for optimization models, J. Math. Computer Sci. 22 (3) (2020), 204-215.

[17] J.K. Liu, Y.X. Zhao, X.L. Wu, Some three-term conjugate gradient methods with the new direction structure, Appl. Numer. Math. 150 (2020), 433-443.

[18] J. Nocedal, S.J. Wright, Numerical optimization, Springer Science & Business Media, New York, 2006.

[19] L.H. Chen, N.Y. Deng, J.Z. Zhang, A modified quasi-Newton method for structured optimization with partial information on the Hessian, Comput. Optim. Appl. 35 (1) (2006), 5-18.

[20] M. Jamil, X-S. Yang A literature survey of benchmark functions for global optimisation problems, Int. J. Math. Model. Numer. Optim. 4 (2) (2013), 150-194.

[21] M.J.D. Powell, Restart procedures for the conjugate gradient method, Math. Program. 12 (1977), 241-254.

[22] M. Malik, M. Mamat, S.S. Abas, I.M. Sulaiman, Sukono, A new coefficient of the conjugate gradient method with the sufficient descent condition and global convergence properties, Eng. Lett. 28 (3) (2020), 704-714.

[23] M. Malik, M. Mamat, S.S. Abas, I.M. Sulaiman, Sukono, A new spectral conjugate gradient method with descent condition and global convergence property for unconstrained optimization, J. Math. Comput. Sci. 10 (5) (2020), 2053-2069.

[24] M. Malik, S.S. Abas, M. Mamat, Sukono, I.M. Sulaiman, A new hybrid conjugate gradient method with global convergence properties, Int. J. Adv. Sci. Technol. 29 (5) (2020), 199-210.
[25] M. Malik, M. Mamat, S.S. Abas, I.M. Sulaiman, Sukono, Performance analysis of new spectral and hybrid conjugate gradient methods for solving unconstrained optimization problems, IAENG Int. J. Computer Sci. 48 (1) (2021), 66-79.

[26] M. Mamat, I.M. Sulaiman, M. Malik, Z.A. Zakaria, An efficient spectral conjugate gradient parameter with descent condition for unconstrained optimization, J. Adv. Res. Dyn. Control Syst. 12 (2) (2020), 2487-2493.

[27] M.R. Hestenes, E. Stiefel, Methods of conjugate gradients for solving linear systems, J. Res. Nat. Bureau Standards, 49 (6) (1952), 409-436.

[28] N. Andrei, Nonlinear conjugate gradient methods for unconstrained optimization, Springer International Publishing, 2020.

[29] R. Dehghani, N. Bidabadi, M.M. Hosseini, A new modified BFGS method for solving systems of nonlinear equations, J. Interdiscip. Math. 22 (1) (2019), 75-89.

[30] R. Fletcher, Practical methods of optimization, John Wiley and Sons, New York, 2013.

[31] R. Fletcher, C.M. Reeves, Function minimization by conjugate gradients, Computer J. 7 (2) (1964), 149-154.

[32] R. Pytlak, Conjugate gradient algorithms in nonconvex optimization, Springer, New York, 2008.

[33] S. Roman, Introduction to the mathematics of finance: from risk management to options pricing, Springer, New York, 2004.

[34] Tarzanagh, D. Ataee, M. Reza Peyghami, A new regularized limited memory BFGS-type method based on modified secant conditions for unconstrained optimization problems, J. Glob. Optim. 63 (4) (2015), 709-728.

[35] U.A. Yakubu, I.M. Sulaiman, M. Mamat, P.L. Ghazali, K. Khalid, The global convergence properties of a descent conjugate gradient method, J. Adv. Res. Dyn. Control Syst. 12 (2) (2020), 1011-1016.

[36] W. W. Hager, H. Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search, SIAM J. Optim. 16 (1) (2005), 170-192.

[37] Y. Liu, C. Storey, Efficient generalized conjugate gradient algorithms, Part 1: theory, J. Optim. Theory Appl. 69 (1) (1991), 129-137.

[38] Y.H. Dai, L.Z. Liao, D. Li, On Restart procedures for the conjugate gradient method, Numer. Algorithms, 35 (2004), 249-260.

[39] Y.H. Dai, Y. Yuan, A nonlinear conjugate gradient method with a strong global convergence property, SIAM J. Optim. 10 (1) (1999), 177-182.

[40] Z. Dai, Comments on a new class of nonlinear conjugate gradient coefficients with global convergence properties, Appl. Math. Comput. 276 (2016), 297-300.

[41] Z. Wei, G. Li, L. Qi, New quasi-Newton methods for unconstrained optimization problems, Appl. Math. Comput. 175 (2) (2006), 1156-1188.