A NEW SPECTRAL CONJUGATE GRADIENT METHOD WITH DESCENT CONDITION AND GLOBAL CONVERGENCE PROPERTY FOR UNCONSTRAINED OPTIMIZATION

MAULANA MALIK\textsuperscript{1,2,*}, MUSTAFA MAMAT\textsuperscript{1}, SITI S. ABAS\textsuperscript{1}, IBRAHIM M. SULAIMAN\textsuperscript{1}, SUKONO\textsuperscript{3}

\textsuperscript{1}Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin (UniSZA), Terengganu 22200, Malaysia
\textsuperscript{2}Department of Mathematics, Universitas Indonesia (UI), Depok 16424, Indonesia
\textsuperscript{3}Department of Mathematics, Universitas Padjadjaran (Unpad), Jatinangor 45361, Indonesia

Copyright © 2020 the author(s). This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract. The Spectral conjugate gradient method is an efficient method for solving large-scale unconstrained optimization problems. In this paper, we propose a new spectral conjugate gradient method in which performance is analyzed numerically. We establish the descent condition and global convergence property under some assumptions and the strong Wolfe line search. Numerical experiments to evaluate the method’s efficiency are conducted using 98 problems with various dimensions and initial points. The numerical results based on the number of iterations and central processing unit time show that the new method has a high performance computational.

Keywords: spectral conjugate gradient method; unconstrained optimization; descent condition; global convergence property; strong Wolfe line search.

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

1. INTRODUCTION

The conjugate gradient method is an efficient method and very interesting for solving large-scale optimization problems since it can be done with lower storage and easy calculation. For

\footnote{*Corresponding author}

E-mail address: m.malik@sci.ui.ac.id

Received July 4, 2020
good reference for studies application of the conjugate gradient method, see [1, 2, 3]. In this article, we consider the following problem to be unconstrained minimization:

\[ \text{(1)} \quad \min \{ f(x) \mid x \in \mathbb{R}^n \} \]

where \( \mathbb{R} \) is real number and \( f \) is a continuously differentiable function. Generally, conjugate gradient method is an iterative method with iterations formula defined as

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

where \( x_k \) is \( k \)th approximation to a solution of problem (1) with \( x_0 \) is starting point, and \( \alpha_k \) is step length obtained by some line search. In this article, we use strong Wolfe line search as follows

\[ \text{(3)} \quad f(x_k + \alpha_k d_k) \leq f(x_k) + \delta \alpha_k g_k^T d_k, \quad \left| g(x_k + \alpha_k d_k)^T d_k \right| \leq -\sigma g_k^T d_k \]

where \( g_k = g(x_k) = \nabla f(x_k) \) is a gradient of function \( f \) at point \( x_k \), \( g_k^T \) is transpose \( g_k \), and \( 0 < \delta < \sigma < 1 \) [4]. Search direction in conjugate gradient method \( d_k \) defined as

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

where \( \beta_k \) is a coefficient determining different formulas [5]. The most well-known conjugate gradient methods are the Hestenes-Stiefel (HS) method [6], Fletcher-Reeves (FR) method [7], Conjugate Descent (CD) [8], Polak-Ribiére-Polyak (PRP) [9], and Wei-Yao-Liu (WYL) method [10], where the formulas \( \beta_k \) for corresponding method respectively are

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

\[ \beta_k^{FR} = \frac{\|g_k\|^2}{\|g_{k-1}\|^2}, \]

\[ \beta_k^{CD} = \frac{\|g_k\|^2}{d_{k-1}^T g_{k-1}}, \]

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

\[ \beta_k^{WYL} = \frac{g_k^T (g_k - \frac{\|g_k\|}{\|g_{k-1}\|} g_{k-1})}{\|g_{k-1}\|^2}. \]
Another common way to solve the problem (1) is to use the spectral conjugate gradient method introduced by Raydan [11] and initially the idea of the spectral conjugate gradient method proposed by Barzilai and Borwein [12]. The main difference between spectral gradient method and gradient conjugate method lies in the calculate of the search direction. The search direction of the spectral gradient method is as follows:

\[ d_k = -\theta_k g_k + \beta_k s_{k-1} \]

where \( s_{k-1} = \alpha_{k-1} d_{k-1} \), and \( \theta_k \) is the spectral gradient parameter. In 2001, Birgin and Martinez [13] developed three kinds of the spectral methods which are a combination of spectral methods and conjugate gradient methods with the following parameters \( \beta_k \):

\[ \beta_k = \frac{(\theta_k y_{k-1} - s_{k-1})^T g_k}{s_{k-1}^T y_{k-1}}, \quad \beta_k = \frac{\theta_k g_k^T y_{k-1}}{\alpha_{k-1} \theta_{k-1} s_{k-1}^T g_k}, \quad \beta_k = \frac{\theta_k g_k^T g_k}{\alpha_{k-1} \theta_{k-1} g_{k-1}^T g_{k-1}}. \]

where

\[ y_{k-1} = g_k - g_{k-1}, \quad \theta_k = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}. \]

Based on numerical results, three methods above are quite efficient, but the descent direction is not necessarily fulfilled. Therefore, Zhang et al. [14] make a modification to the FR method (MFR) so that the method has been proven descent direction, and satisfies global convergence property under Armijo line search, where the search direction is defined as

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

and

\[ \beta_k = \beta_{k-1}^{FR}, \quad \theta_k = \frac{d_{k-1}^T y_{k-1}}{||g_{k-1}||^2}, \]

which search direction of the MFR method can be written as follows:

\[ d_k = \left(1 + \beta_{k-1}^{FR} \frac{g_k^T d_{k-1}}{||g_k||^2}\right) g_k + \beta_k^{FR} d_{k-1}. \]

As well as in 2012, Liu and Jiang [15] has proposed a new kind of the spectral conjugate gradient method (SCD), where the coefficient \( \beta_k \), and spectral gradient parameter \( \theta_k \) determined
by
\[
\beta_k = \begin{cases} 
\beta_k^{CD}, & \text{if } g_k^T d_{k-1} \leq 0 \\
0, & \text{else}
\end{cases}
\]

\[
\theta_k = 1 - \frac{g_k^T d_{k-1}}{g_{k-1}^T d_{k-1}}.
\]

Recently, Jian et al. [16] proposed a new class of the spectral conjugate gradient method, which they are choice for spectral parameter \( \theta_k \) as follows:

\[
\theta_k^{JYJLL} = 1 + \frac{|g_k^T d_{k-1}|}{g_{k-1}^T d_{k-1}}
\]

and conjugate gradient parameter in form

\[
\beta_{k}^{JYJLL} = \frac{\|g_k\|^2 - (g_k^T d_{k-1})^2}{\max \{ \|g_{k-1}\|^2, d_{k-1}^T \left( g_k - g_{k-1} \right) \}}
\]

The JYJLL spectral conjugate gradient method (JYJLL-SCGM) always fulfills the descent condition without depending any line search and the global convergence properties under Wolfe line search are met also. The numerical experiments of the JYJLL-SCGM in comparison with AN1 [17], KD [18], HZ [19], and LFZ [20] methods show that JYJLL-SCGM is most effective.

The main objective of this paper is to propose a new spectral conjugate gradient method and compare its performance with the MFR, SCD, JYJLL, and NPRP method (see Zhang [21]). This paper will be organized as follows: In section 2, a new spectral conjugate gradient formula, and the algorithm will be presented. In section 3, we will show the descent condition and global convergence property of our new method. Numerical experiments will be presented in section 4. Finally, our conclusion will be written in section 5.

### 2. New Spectral Conjugate Gradient Formula

In this section, we first propose a new conjugate gradient coefficient based on the NPRP conjugate gradient formula in Ref. [21]. NPRP method is a modification of the PRP method, and development of WYL method. The coefficient \( \beta_k \) of NPRP method is defined as:

\[
\beta_k^{NPRP} = \frac{\|g_k\|^2 - \|g_k - g_{k-1}\| \cdot |g_k^T g_{k-1}|}{\|g_{k-1}\|^2},
\]
and our new coefficient defined as:

\[
\beta_k^{\text{MMSMS}} = \begin{cases} 
\frac{\|g_k\|^2 - \|g_k\| \|g_{k-1}\| - \|g_k^T g_{k-1}\|}{(1-\mu)\|d_{k-1}\|^2 + \mu \|g_{k-1}\|^2}, & \text{if } \|g_k\|^2 > \left( \frac{\|g_k\|}{\|g_{k-1}\|} + 1 \right) \|g_k^T g_{k-1}\|, \\
0, & \text{otherwise}
\end{cases}
\]

that is, we add a negative \(|g_k^T g_{k-1}|\) in the numerator \(\beta_k^{\text{NPRP}}\), extend the denominator by \((1-\mu)\|d_{k-1}\|^2 + \mu \|g_{k-1}\|^2\), and prevent negative value, where \(\mu = 0.9\).

Secondly, we propose a new spectral parameter which is quite same to MFR formula (5) but different coefficient of \(\beta_k\) as follows:

\[
\theta_k^{\text{MMSMS}} = 1 + \beta_k^{\text{MMSMS}} \frac{g_k^T d_{k-1}}{\|g_k\|^2}.
\]

The MMSMS is symbolizes Malik, Mustafa, Sabariah, Mohammed, Sukono.

In the following, we describe the algorithm spectral MMSMS (SpMMSMS) method for solving unconstrained optimization problems.

**Algorithm 2.1. (SpMMSMS Method)**

**Step 1.** Choose an initial point \(x_0 \in \mathbb{R}^n\). Given the stopping criteria \(\varepsilon > 0\), parameter \(\sigma\), and \(\delta\).

**Step 2.** Compute \(\|g_k\|\), if \(\|g_k\| \leq \varepsilon\) then stop, \(x_k\) is optimal point. Else, go to Step 3.

**Step 3.** Compute \(\beta_k\) using (6).

**Step 4.** Compute search direction \(d_k\) using (4) with \(\beta_k^{\text{MMSMS}}\) and \(\theta_k^{\text{MMSMS}}\).

**Step 5.** Compute step length \(\alpha_k\) using the strong Wolfe line search (3).

**Step 6.** Set \(k := k + 1\), calculate the next iteration \(x_{k+1}\) using (2), and go to Step 2.

**3. Descent Condition and Global Convergence Property**

In this section, we analyze the descent condition and global convergence property of the SpMMSMS method. Therefore, we need the following definition.

**Definition 3.1.** [22] Let \(d_k\) is the search direction. If

\[
g_k^T d_k < 0, \text{ for all } k \geq 0.
\]

then the descent condition holds.
Definition 3.2. [22] We say that a conjugate gradient method is global convergence if

\[
\lim_{k \to \infty} \inf \| g_k \| = 0.
\]

The theorem below shows that the SpMMSMS method always fulfills the descent condition.

Theorem 3.3. Suppose that the search direction \( d_k \) is generated by SpMMSMS method, then

\[
g_k^T d_k = -\| g_k \|^2 < 0
\]

holds for any \( k \geq 0 \).

Proof. If \( k = 0 \), then \( d_0 = -g_0 \) and we obtain \( g_0^T d_0 = -g_0^T g_0 = -\| g_0 \|^2 < 0 \). So, for \( k = 0 \), the SpMMSMS method satisfies the descent condition. Now, for \( k \geq 1 \), we have

\[
d_k = -\theta_k g_k + \beta_k d_{k-1}.
\]

Substituting \( \theta_k \) by \( \theta_k^{MMSMS} \) and \( \beta_k \) by \( \beta_k^{MMSMS} \), then we get

(8) \[
d_k = -\theta_k^{MMSMS} g_k + \beta_k^{MMSMS} d_{k-1} = -\left( 1 + \beta_k^{MMSMS} \frac{g_k^T d_{k-1}}{\| g_k \|^2} \right) g_k + \beta_k^{MMSMS} d_{k-1}.
\]

Based on value of \( \beta_k^{MMSMS} \), we have two cases for \( d_k \):

Case 1. For \( \| g_k \|^2 \leq \left( \| g_k \| \| g_{k-1} \| + 1 \right) \| g_k^T g_{k-1} \| \), then from (6), and (8), we have

\[
d_k = -g_k.
\]

Multiplying both sides by \( g_k^T \), we get \( g_k^T d_k = -g_k^T g_k = -\| g_k \|^2 < 0 \). Hence, the descent condition holds.

Case 2. For \( \| g_k \|^2 > \left( \| g_k \| \| g_{k-1} \| + 1 \right) \| g_k^T g_{k-1} \| \), then from (8) and multiplying both sides by \( g_k^T \), we obtain

\[
g_k^T d_k = -\left( 1 + \beta_k^{MMSMS} \frac{g_k^T d_{k-1}}{\| g_k \|^2} \right) \| g_k \|^2 + \beta_k^{MMSMS} g_k^T d_{k-1} = -\left( 1 + \frac{\| g_k \|^2 - \| g_k \| \| g_k^T g_{k-1} \| - \| g_k^T g_{k-1} \|}{(1 - \mu) \| d_{k-1} \|^2 + \mu \| g_{k-1} \|^2} \right) \| g_k \|^2 + \beta_k^{MMSMS} g_k^T d_{k-1}.
\]
\[ -\|g_k\|^2 - \left( \frac{\|g_k\|^2 - \|g_{k-1}\| \left| g_k^T g_{k-1} \right| - \|g_k^T g_{k-1}\|}{(1 - \mu)\|d_{k-1}\|^2 + \mu\|g_{k-1}\|^2} \right) g_k^T d_{k-1} \]

\[ + \left( \frac{\|g_k\|^2 - \|g_{k-1}\| \left| g_k^T g_{k-1} \right| - \|g_k^T g_{k-1}\|}{(1 - \mu)\|d_{k-1}\|^2 + \mu\|g_{k-1}\|^2} \right) g_k^T d_{k-1} \]

\[ = -\|g_k\|^2 < 0. \]

So that the descent condition holds. Hence, for any \( k \geq 0 \), the descent condition \( g_k^T d_k < 0 \) always satisfies. The proof is completed.

The following lemma is essential to prove the global convergence of the SpMMSMS method.

**Lemma 3.4.** The relation

\( 0 \leq \beta_k^{\text{MMSMS}} \leq \frac{10}{9} \frac{\|g_k\|^2}{\|g_{k-1}\|^2} \)

always holds for any \( k \geq 0 \).

**Proof.** Clearly, from (6) \( \beta_k^{\text{MMSMS}} \) can be 0 or

\[ \beta_k^{\text{MMSMS}} = \frac{\|g_k\|^2 - \|g_{k-1}\| \left| g_k^T g_{k-1} \right| - \|g_k^T g_{k-1}\|}{(1 - \mu)\|d_{k-1}\|^2 + \mu\|g_{k-1}\|^2} \]

\[ = \frac{\|g_k\|^2 - \left( \frac{\|g_k\|}{\|g_{k-1}\|} + 1 \right) \left| g_k^T g_{k-1} \right|}{(1 - \mu)\|d_{k-1}\|^2 + \mu\|g_{k-1}\|^2}. \]

Since \( \left( \frac{\|g_k\|}{\|g_{k-1}\|} + 1 \right) \left| g_k^T g_{k-1} \right| > 0 \), (6), \( (1 - \mu)\|d_{k-1}\|^2 > 0 \), and \( \mu = 0.9 \), then

\[ \beta_k^{\text{MMSMS}} \leq \frac{10}{9} \frac{\|g_k\|^2}{\|g_{k-1}\|^2}, \text{ and } \beta_k^{\text{MMSMS}} > 0. \]

Hence, the relation (9) is true. The proof is finished.

To investigate the global convergence property of the SpMMSMS method, we need the following assumption.

**Assumption 3.5.**

(A1) For any initial point \( x_0 \), the set \( \Omega = \{ x \in \mathbb{R}^n : f(x) \leq f(x_0) \} \) is bounded.

(A2) In any neighborhood \( \Omega_0 \) of \( \Omega \), \( f \) is continuous and differentiable, and its gradient \( g(x) \) is Lipschitz continuous; in other words, there exists a Lipschitz constant \( L > 0 \) such that

\[ \|g(x) - g(y)\| \leq L\|x - y\|, \text{ for all } x, y \in \Omega_0 \]
In the lemma below, we discuss the well-known condition of lemma Zoutendijk [23], which plays an essential role in the conjugate gradient method about convergence analysis.

**Lemma 3.6.** Suppose that Assumption 3.5 holds, let $x_k$ be generated by iterative method $x_{k+1} = x_k + \alpha_k d_k$, where $\alpha_k$ is a step length which calculated by the strong Wolfe line search (3), and the search direction $d_k$ satisfies the descent direction such that $g_k^T d_k < 0$. Then

\[
\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < \infty.
\]

The theorem below, we establish the global convergence property of our new method.

**Theorem 3.7.** Suppose that Assumption 3.5 holds, and let the sequences $\{g_k\}$ and $\{d_k\}$ be generated by the SpMMSMS method with the strong Wolfe line search (3). Then

\[
\lim_{k \to \infty} \inf \|g_k\| = 0.
\]

**Proof.** The proof is done by contradiction. Suppose that (11) is not true. Then there exist a positive constant $\phi > 0$ such that $\|g_k\| \geq \phi$, $\forall k \geq 0$, which means

\[
\frac{1}{\|g_k\|^2} \leq \frac{1}{\phi^2}.
\]

From (8), we have $d_k + \theta_k^{MMSMS} g_k = \beta_k^{MMSMS} d_{k-1}$, and squaring the both sides, we get

\[
\|d_k\|^2 = \left(\beta_k^{MMSMS}\right)^2 \|d_{k-1}\|^2 - 2\theta_k^{MMSMS} g_k^T d_k \left(\theta_k^{MMSMS}\right)^2 \|g_k\|^2.
\]
Dividing both sides of (13) by \((g_k^T d_k)^2\), and combining with Theorem 3.3, and (9), we obtain

\[
\frac{\|d_k\|^2}{(g_k^T d_k)^2} = (\beta_k^{MMSMS})^2 \frac{\|d_{k-1}\|^2}{\|g_k\|^4} + \frac{2\theta_k^{MMSMS}}{\|g_k\|^2} - \left(\frac{\theta_k^{MMSMS}}{\|g_k\|^2}\right)^2
\]

\[
= (\beta_k^{MMSMS})^2 \frac{\|d_{k-1}\|^2}{\|g_k\|^4} - \frac{1}{\|g_k\|^2} \left(\left(\frac{\theta_k^{MMSMS}}{\|g_k\|^2}\right)^2 - 2\theta_k^{MMSMS}\right)
\]

\[
= (\beta_k^{MMSMS})^2 \frac{\|d_{k-1}\|^2}{\|g_k\|^4} - \frac{1}{\|g_k\|^2} \left(\left(\frac{\theta_k^{MMSMS}}{\|g_k\|^2} - 1\right)^2 - 1\right)
\]

\[
= (\beta_k^{MMSMS})^2 \frac{\|d_{k-1}\|^2}{\|g_k\|^4} - \frac{\left(\theta_k^{MMSMS} - 1\right)^2}{\|g_k\|^2} + \frac{1}{\|g_k\|^2}
\]

\[
\leq (\beta_k^{MMSMS})^2 \frac{\|d_{k-1}\|^2}{\|g_k\|^4} + \frac{1}{\|g_k\|^2}
\]

\[
\leq \left(\frac{100}{9}\right) \frac{\|g_k\|^2}{\|g_{k-1}\|^2} \frac{\|d_{k-1}\|^2}{\|g_k\|^4} + \frac{1}{\|g_k\|^2} = \frac{100}{81} \frac{\|d_{k-1}\|^2}{\|g_{k-1}\|^4} + \frac{1}{\|g_k\|^2}.
\]

We know that \(\frac{\|d_0\|^2}{\|g_0\|^2} = \frac{1}{\|g_0\|^2}\), applying (12) to above relation, we have

\[
\frac{\|d_k\|^2}{(g_k^T d_k)^2} \leq \frac{100}{81} \frac{\|d_{k-1}\|^2}{\|g_{k-1}\|^4} + \frac{1}{\|g_k\|^2}
\]

\[
\leq \left(\frac{100}{81}\right)^2 \frac{\|d_{k-2}\|^2}{\|g_{k-2}\|^4} + \frac{1}{\|g_{k-1}\|^2} + \frac{1}{\|g_k\|^2}
\]

\[
\leq \ldots \leq \left(\frac{100}{81}\right)^k \frac{1}{\|g_0\|^2} + \sum_{i=1}^{k} \frac{1}{\|g_i\|^2}
\]

\[
\leq \left(\frac{100}{81}\right)^k \frac{1}{\|g_0\|^2} + \frac{k}{\phi^2} = Z,
\]

where \(Z\) is an arbitrary scalar. So, we have \(\frac{(g_k^T d_k)^2}{\|d_k\|^2} \geq \frac{1}{Z}\). Furthermore,

\[
\left(\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2}\right) = \lim_{n \to \infty} \sum_{k=0}^{n} \frac{(g_k^T d_k)^2}{\|d_k\|^2} \geq \left(\sum_{k=0}^{\infty} \frac{1}{Z}\right) = \lim_{n \to \infty} \sum_{k=0}^{n} \frac{1}{Z} = \lim_{n \to \infty} \frac{n+1}{Z} = \infty.
\]

Hence \(\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} \geq \infty\), which contradicts (10) in Lemma 3.6. So, base on Definition 3.2, we say that SpMMSMS fulfills global convergence property. The proof is completed. \(\Box\)

### 4. Numerical Experiments and Discussion

In this section, we present the computational results of SpMMSMS, MFR, SCD, JYJLL, and NPRP method. Some test functions considered in Andrei [24] to analyze the efficiency of each
method. The comparison is made using 98 problems with various initial points and dimensions, as in Table 1. Most of the initial points used are suggestions from Andrei [24], and dimensional variations by Malik et al. [25, 26], namely 2, 3, 4, 10, 50, 100, 500, 1000, 5000 and 10000. The function that is used is an artificial function. Artificial functions are used to detect algorithmic behavior under different conditions such as local optimal functions, valley-shaped function, unimodal functions, bowl-shaped functions, plate-shaped functions, and other functions.

Comparison of numerical results based on the number of iterations (NOI) and central processing unit (CPU) times (in second). The results are said to fail if the number of iterations exceeds 10,000, and no solution is reached. To get entries in Table 2, we use the algorithm code written in Matlab R2019a and ran on a personal laptop with specifications; processor Intel Core i7, 16 GB RAM memory, and operating system Windows 10 Pro 64 bit. All results have the same stopping criteria with $\varepsilon = 10^{-6}$ and implemented under strong Wolfe line search with $\sigma = 0.001$ and $\delta = 0.0001$. 
### Table 1. List of the test functions, dimension, and initial point.

| Prob | Function            | Dim    | Initial point          | Prob | Function            | Dim    | Initial point          |
|------|---------------------|--------|------------------------|------|---------------------|--------|------------------------|
| 1    | Ext. White & Holst  | 1000   | (-1.2,1,...,-1.2,1)    | 50   | Ext. Maratos        | 10     | (-1,...,-1)            |
| 2    | Ext. White & Holst  | 1000   | (10,...,10)            | 51   | Six hump camel      | 2      | (-1.2)                 |
| 3    | Ext. White & Holst  | 10000  | (-1.2,1,...,-1.2,1)    | 52   | Six hump camel      | 2      | (-5,10)                |
| 4    | Ext. White & Holst  | 10000  | (5,...,5)              | 53   | Three hump camel    | 2      | (-1.2)                 |
| 5    | Ext. Rosenbrock     | 1000   | (-1.2,1,...,-1.2,1)    | 54   | Three hump camel    | 2      | (2,-1)                 |
| 6    | Ext. Rosenbrock     | 1000   | (10,...,10)            | 55   | Booth               | 2      | (5,5)                  |
| 7    | Ext. Rosenbrock     | 10000  | (-1.2,1,...,-1.2,1)    | 56   | Booth               | 2      | (10,10)                |
| 8    | Ext. Rosenbrock     | 10000  | (5,...,5)              | 57   | Trecanni            | 2      | (-1.0,5)               |
| 9    | Ext. Freudenstein & Roth | 4     | (0.5,-2,0.5,-2)       | 58   | Trecanni            | 2      | (-5,10)                |
| 10   | Ext. Freudenstein & Roth | 4   | (5,5,5,5)             | 59   | Zettl               | 2      | (-1.2)                 |
| 11   | Ext. Beale          | 1000   | (1.0,8,...,1.0,8)      | 60   | Zettl               | 2      | (10,10)                |
| 12   | Ext. Beale          | 1000   | (0.5,...,0.5)          | 61   | Shallow             | 1000   | (0,...,0)              |
| 13   | Ext. Beale          | 10000  | (-1,...,-1)            | 62   | Shallow             | 1000   | (10,...,10)            |
| 14   | Ext. Beale          | 10000  | (0.5,...,0.5)          | 63   | Shallow             | 10000  | (-1,...,-1)            |
| 15   | Ext. Wood           | 4      | (-3,-1.3,-1)          | 64   | Shallow             | 10000  | (-10,...,-10)          |
| 16   | Ext. Wood           | 4      | (5,5,5,5)             | 65   | Generalized Quartic | 1000   | (1,...,1)              |
| 17   | Raydan 1            | 10     | (1,...,1)             | 66   | Generalized Quartic | 1000   | (20,...,20)            |
| 18   | Raydan 1            | 10     | (10,...,10)           | 67   | Quadratic QF2       | 50     | (0.5,...,0.5)          |
| 19   | Raydan 1            | 100    | (-1,...,-1)           | 68   | Quadratic QF2       | 50     | (30,...,30)            |
| 20   | Raydan 1            | 100    | (-10,...,-10)         | 69   | Leon                | 2      | (2,2)                  |
| 21   | Ext. Tridiagonal 1  | 500    | (2,...,2)             | 70   | Leon                | 2      | (8,8)                  |
| 22   | Ext. Tridiagonal 1  | 500    | (10,...,10)           | 71   | Gen. Tridiagonal 1  | 10     | (2,...,2)              |
| 23   | Ext. Tridiagonal 1  | 1000   | (1,...,1)             | 72   | Gen. Tridiagonal 1  | 10     | (10,...,10)            |
| 24   | Ext. Tridiagonal 1  | 1000   | (-10,...,-10)         | 73   | Gen. Tridiagonal 2  | 4      | (1,1,1,1)              |
| 25   | Diagonal 4          | 500    | (1,...,1)             | 74   | Gen. Tridiagonal 2  | 4      | (10,10,10,10)          |
| 26   | Diagonal 4          | 500    | (-20,...,-20)         | 75   | POWER               | 10     | (1,...,1)              |
| 27   | Diagonal 4          | 1000   | (1,...,1)             | 76   | POWER               | 10     | (10,...,10)            |
| 28   | Diagonal 4          | 1000   | (-30,...,-30)         | 77   | Quadratic QF1       | 50     | (1,...,1)              |
| 29   | Ext. Himmelblau     | 1000   | (1,...,1)             | 78   | Quadratic QF1       | 50     | (10,...,10)            |
| 30   | Ext. Himmelblau     | 1000   | (20,...,20)           | 79   | Quadratic QF1       | 500    | (1,...,1)              |
| 31   | Ext. Himmelblau     | 10000  | (-1,...,-1)           | 80   | Quadratic QF1       | 500    | (-5,...,-5)            |
| 32   | Ext. Himmelblau     | 10000  | (50,...,50)           | 81   | Ext.quad.pen.QP2    | 100    | (1,...,1)              |
| 33   | FLETCHCR            | 10     | (0,...,0)             | 82   | Ext.quad.pen.QP2    | 100    | (10,...,10)            |
| 34   | FLETCHCR            | 10     | (10,...,10)           | 83   | Ext.quad.pen.QP2    | 500    | (10,...,10)            |
| 35   | Ext. Powel          | 100    | (3,...,1,0,1,...,1)    | 84   | Ext.quad.pen.QP2    | 500    | (50,...,50)            |
| 36   | Ext. Powel          | 100    | (5,...,5)             | 85   | Ext.quad.pen.QP1    | 4      | (1,1,1,1)              |
| 37   | NONSCOMP            | 2      | (3,3)                 | 86   | Ext.quad.pen.QP1    | 4      | (10,10,10,10)          |
| 38   | NONSCOMP            | 2      | (10,10)               | 87   | Quartic             | 4      | (10,10,10,10)          |
| 39   | Ext. DENSCHNB       | 10     | (1,...,1)             | 88   | Quartic             | 4      | (15,15,15,15)          |
| 40   | Ext. DENSCHNB       | 10     | (10,...,10)           | 89   | Matyas              | 2      | (1,1)                  |
| 41   | Ext. DENSCHNB       | 100    | (10,...,10)           | 90   | Matyas              | 2      | (20,20)                |
| 42   | Ext. DENSCHNB       | 100    | (-50,...,-50)         | 91   | Colville            | 4      | (2,2,2,2)              |
| 43   | Ext. Penalty        | 10     | (1,2,3,...,10)        | 92   | Colville            | 4      | (10,10,10,10)          |
| 44   | Ext. Penalty        | 10     | (-10,...,-10)         | 93   | Dixon and Price     | 3      | (1,1,1)                |
| 45   | Ext. Penalty        | 100    | (5,...,5)             | 94   | Dixon and Price     | 3      | (10,10,10)             |
| 46   | Ext. Penalty        | 100    | (10,...,10)           | 95   | Sphere              | 5000   | (1,...,1)              |
| 47   | Hager               | 10     | (1,...,1)             | 96   | Sphere              | 5000   | (10,...,10)            |
| 48   | Hager               | 10     | (-10,...,-10)         | 97   | Sum Squares         | 50     | (0,1,...,0,1)          |
| 49   | Ext. Maratos        | 10     | (1.1,0.1)             | 98   | Sum Squares         | 50     | (10,...,10)            |
| Problem | SpM | MMS | SMS | JYJL | MFR | SCD | NPRP |
|---------|-----|-----|-----|------|-----|-----|------|
|         | NOI | CPU | NOI | CPU | NOI | CPU | NOI | CPU |
| 1       | 16  | 0.0597 | 49  | 0.1059 | 15  | 0.0625 | 17  | 0.0574 |
| 2       | 54  | 0.1786 | 210 | 0.9704 | 105 | 0.252  | 55  | 0.2006 |
| 3       | 16  | 0.3636 | 50  | 0.8048 | 15  | 0.389  | 17  | 0.4105 |
| 4       | 40  | 1.1356 | 130 | 4.4795 | 382  | 5.0925 | 42  | 1.1005 |
| 5       | 24  | 0.0538 | 59  | 0.1179 | 5954 | 3.6831 | 35  | 0.0587 |
| 6       | 33  | 0.0562 | 104 | 0.1882 | 150  | 0.1317 | 25  | 0.0392 |
| 7       | 16  | 0.3636 | 81  | 0.1376 | 335  | 0.4867 | 25  | 0.0639 |
| 8       | 34  | 0.858 | 87  | 1.2204 | 87   | 1.2208 | 14  | 0.32  |
| 9       | 8   | 0.0011 | 21  | 0.0066 | 21  | 0.0029 | 9   | 0.0183 |
| 10      | 11  | 0.0175 | fail | fail | fail | fail | fail | fail |
| 11      | 13  | 0.0469 | 75  | 0.1352 | 75   | 0.1322 | 17  | 0.0732 |
| 12      | 12  | 0.0431 | 81  | 0.1376 | 81   | 0.1436 | 35  | 0.3867 |
| 13      | 14  | 0.2908 | 87  | 1.2204 | 87   | 1.2208 | 14  | 0.32  |
| 14      | 8   | 0.0024 | 19  | 0.0065 | 19   | 0.0047 | 20  | 0.015 |
| 15      | 33  | 0.0043 | 2350 | 0.1437 | 2620 | 0.1278 | 50  | 0.0087 |
| 16      | 91  | 0.0343 | 93  | 0.0311 | 95   | 0.0243 | 253  | 0.0509 |
| 17      | 149 | 0.0472 | 801 | 0.349  | fail | fail | 390  | 0.0782 |
| 18      | 12  | 0.0294 | 452 | 0.3883 | 452  | 0.4074 | 19   | 0.0373 |
| 19      | 8   | 0.0235 | 9   | 0.0169 | 9    | 0.0225 | 15   | 0.0258 |
| 20      | 12  | 0.0382 | 517 | 0.7546 | 517  | 0.767 | 19   | 0.0458 |
| 21      | 8   | 0.0376 | 8   | 0.0236 | 9    | 0.0265 | 17   | 0.0391 |
| 22      | 2   | 0.0002 | 2   | 0.0021 | 2    | 0.0017 | 5    | 0.0116 |
| 23      | 2   | 0.0021 | 2   | 0.0021 | 2    | 0.0021 | 3    | 0.0022 |
| 24      | 2   | 0.0037 | 2   | 0.0035 | 2    | 0.0025 | 4    | 0.0057 |
| 25      | 2   | 0.0034 | 2   | 0.0034 | 2    | 0.0031 | 4    | 0.0034 |
| 26      | 8   | 0.0148 | 14  | 0.0199 | 15   | 0.0208 | 21   | 0.042 |
| 27      | 6   | 0.0152 | 9   | 0.0164 | 9    | 0.0211 | 10   | 0.0149 |
| 28      | 11  | 0.1136 | 22  | 0.2385 | 17   | 0.2096 | 15   | 0.1613 |
| 29      | 7   | 0.0862 | 13  | 0.1242 | 13   | 0.1156 | 18   | 0.1403 |
| 30      | 37  | 0.0046 | 1142 | 0.0615 | 1208 | 0.0461 | 153  | 0.0217 |
| 31      | 82  | 0.0094 | 403  | 0.0373 | 299  | 0.0214 | 148  | 0.0105 |
| 32      | 109 | 0.0384 | 5487 | 1.1151 | 5589 | 1.0281 | fail | fail |
| 33      | 98  | 0.0525 | 6066 | 1.2068 | 6019 | 1.0706 | fail | fail |
| 34      | 12  | 0.0012 | 86  | 0.0081 | 156  | 0.0052 | 28   | 0.0071 |
| 35      | 14  | 0.0018 | 88  | 0.0049 | 93   | 0.0043 | 22   | 0.0025 |
| 36      | 7   | 0.0013 | 9   | 0.0011 | 9    | 0.00734 | 10   | 0.0105 |
| 37      | 9   | 0.0014 | 11  | 0.0014 | 11   | 0.0014 | 21   | 0.0027 |
| 38      | 10  | 0.0044 | 11  | 0.0026 | 11   | 0.0045 | 22   | 0.0087 |
| 39      | 7   | 0.0031 | 63  | 0.0132 | 63   | 0.0099 | 11   | 0.0055 |
| 40      | 14  | 0.0071 | 10  | 8.84E-04 | 11   | 0.0011 | 45   | 0.0151 |
| 41      | 9   | 0.0024 | 19  | 0.0024 | 19   | 0.0014 | 54   | 0.0058 |
| 42      | 9   | 0.0053 | 28  | 0.0087 | 28   | 0.0109 | 10   | 0.0044 |
| 43      | 9   | 0.0047 | fail | fail | 28   | 0.1552 | 17   | 0.0121 |
| 44      | 13  | 0.0014 | 11  | 0.0039 | 11   | 0.0031 | 12   | 0.0107 |
| 45      | 20  | 0.0038 | 96  | 0.0096 | 97   | 0.0076 | 18   | 0.0081 |
| 46      | 36  | 0.0057 | 3527 | 0.5159 | fail | fail | 1229 | 0.0732 |

(Continued on next page)
Table 2 – Continued

| Problem | SpMM | SM | MS | MS | JYJJ | MFR | SCD | NPRP |
|---------|------|----|----|----|------|-----|-----|------|
|         | NOI  | CPU| NOI| CPU| NOI | CPU | NOI | CPU  |
| 50      | 32   | 0.0087 | 165 | 0.0245 | 128 | 0.0121 | 47 | 0.0084 |
| 51      | 7    | 8.44E-04 | 27   | 0.0027 | 27 | 8.09E-04 | 13 | 0.0021 |
| 52      | 10   | 9.07E-04 | 264  | 0.0181 | 536 | 0.0154 | 13 | 0.0019 |
| 53      | 13   | 0.0036 | 11   | 0.0033 | 11 | 0.0024 | 13 | 0.0032 |
| 54      | 12   | 0.0033 | 11   | 0.0024 | 12 | 0.0039 | 13 | 0.0065 |
| 55      | 2    | 2.53E-04 | 2    | 2.63E-04 | 2 | 2.63E-04 | 2 | 2.93E-04 |
| 56      | 2    | 2.92E-04 | 2    | 2.03E-04 | 2 | 1.29E-04 | 2 | 0.0041 |
| 57      | 1    | 2.11E-04 | 1    | 1.82E-04 | 1 | 1.64E-04 | 1 | 2.40E-04 |
| 58      | 5    | 0.0013 | 7    | 8.31E-04 | 7 | 4.69E-04 | 5 | 0.0083 |
| 59      | 11   | 0.0011 | 11   | 0.0081 | 11 | 6.34E-04 | 105 | 0.0064 |
| 60      | 12   | 0.0081 | 16   | 0.0022 | 16 | 9.00E-04 | 68 | 0.0081 |
| 61      | 8    | 0.0157 | 18   | 0.0348 | 18 | 0.0261 | 10 | 0.0109 |
| 62      | 13   | 0.0252 | 78   | 0.0692 | 96 | 0.076 | 50 | 0.0393 |
| 63      | 9    | 0.0849 | 47   | 0.3093 | 47 | 0.2884 | 18 | 0.1548 |
| 64      | 10   | 0.0995 | 10   | 0.0883 | 9  | 0.0753 | 11 | 0.1134 |
| 65      | 5    | 0.0459 | 7    | 0.0345 | 7  | 0.0384 | 5  | 0.0311 |
| 66      | 10   | 0.0454 | 16   | 0.0646 | 40 | 0.2119 | 10 | 0.0477 |
| 67      | 79   | 0.0153 | 116  | 0.017 | 116 | 0.0106 | 200 | 0.0256 |
| 68      | 67   | 0.0149 | 1306 | 0.3108 | fail | fail | 187 | 0.0315 |
| 69      | 22   | 0.0025 | 180  | 0.0157 | 194 | 0.0062 | 8117 | 0.0352 |
| 70      | 49   | 0.0077 | 735  | 0.0449 | 736 | 0.0229 | 4249 | 0.1963 |
| 71      | 21   | 0.0028 | 27   | 0.002 | 27  | 0.0027 | 30  | 0.0046 |
| 72      | 29   | 0.0049 | 43   | 0.0038 | 43  | 0.005 | 34  | 0.0111 |
| 73      | 4    | 4.54E-04 | 5    | 5.72E-04 | 5 | 3.36E-04 | 5 | 6.25E-04 |
| 74      | 11   | 0.0013 | 4710 | 0.205 | 6315 | 0.1435 | 11 | 0.0086 |
| 75      | 101  | 0.0157 | 10   | 0.0013 | 10 | 6.97E-04 | 77 | 0.0089 |
| 76      | 114  | 0.0121 | 10   | 0.0012 | 10 | 6.85E-04 | 105 | 0.0102 |
| 77      | 67   | 0.0133 | 38   | 0.0072 | 38 | 0.0054 | 67  | 0.0087 |
| 78      | 74   | 0.0139 | 40   | 0.0041 | 40 | 0.0056 | 75  | 0.015 |
| 79      | 284  | 0.1225 | 131  | 0.0532 | 131 | 0.0504 | 234 | 0.0883 |
| 80      | 286  | 0.1324 | 137  | 0.0615 | 137 | 0.0489 | 255 | 0.1001 |
| 81      | 30   | 0.0204 | 255  | 0.141 | 388 | 0.1794 | 46  | 0.0189 |
| 82      | 33   | 0.0226 | 3690 | 0.997 | 490 | 0.1991 | 40  | 0.0205 |
| 83      | 60   | 0.1051 | 1149 | 3.5478 | 1217 | 3.475 | 87  | 0.1231 |
| 84      | 58   | 0.105 | 1763 | 4.102 | 1132 | 3.246 | 120 | 0.1653 |
| 85      | 9    | 9.53E-04 | 20   | 0.0022 | 20 | 0.001 | 8  | 8.67E-04 |
| 86      | 9    | 0.0014 | 51   | 0.0056 | 51 | 0.0025 | 12  | 0.0092 |
| 87      | 81   | 0.0108 | 272  | 0.0166 | 272 | 0.0111 | 4334 | 0.2208 |
| 88      | 91   | 0.0119 | 273  | 0.0181 | 273 | 0.0138 | 1230 | 0.0778 |
| 89      | 1    | 2.97E-04 | 1    | 2.58E-04 | 1 | 0.0011 | 1  | 0.0011 |
| 90      | 1    | 0.007 | 1    | 0.003 | 1   | 0.0015 | 1  | 0.0056 |
| 91      | 214  | 0.0227 | fail | fail | fail | fail | 4295 | 0.2079 |
| 92      | 87   | 0.0149 | 33   | 0.0029 | 33 | 0.0015 | 1346 | 0.0801 |
| 93      | 13   | 0.0014 | 16   | 0.0015 | 16 | 8.83E-04 | 55 | 0.006 |
| 94      | 23   | 0.0044 | 24   | 0.0027 | 25 | 0.002 | 105 | 0.0119 |
| 95      | 1    | 0.0065 | 1    | 0.0083 | 1  | 0.0075 | 1  | 0.0052 |
| 96      | 1    | 0.0109 | 1    | 0.0072 | 1  | 0.0043 | 1  | 0.0151 |
| 97      | 45   | 0.0137 | 25   | 0.005 | 25 | 0.0054 | 45  | 0.0065 |
| 98      | 77   | 0.0161 | 41   | 0.0048 | 41 | 0.0055 | 77  | 0.0181 |
Based on Table 2, we can summarize the results for each method, the total number of iterations, the total of CPU Times, and the problem that was successfully resolved in Table 3.

| Methods   | TNOI  | TCPU          | Successful |
|-----------|-------|---------------|------------|
| SpMMSMS   | 3,756 | 4.8720114     | 100%       |
| JYJLL     | 38,483| 30.2860924    | 94%        |
| MFR       | 31,480| 25.9044023    | 93%        |
| SCD       | 46,778| 66.3459239    | 96%        |
| NPRP      | 9,625 | 7.1376309     | 96%        |

Figure 1 and Figure 2, respectively, display the performance profiles of each method using a performance profile introduced by Dolan and Moré [27]. The formulas used to describe the outcome of the profile will be explained as follows:

\[
    r_{p,s} = \frac{\tau_{p,s}}{\min\{\tau_{p,s} : p \in P \text{ and } s \in S\}}, \quad \rho_s(\tau) = \frac{1}{n_p} \text{size}\{p \in P : r_{p,s} \leq \tau\}
\]

where

- $S$ is the set solvers,
- $P$ is the test set of problems,
- $n_s$ is the number of solvers,
- $n_p$ is the number of problems,
- $\tau(p,s)$ is computing time (NOI or others) needed to solve problem $p$ by solver $s$,
- $r_{p,s}$ is performance profile ratio,
- $\rho_s(\tau)$ is the probability for solvers,

$r_{p,s}$ used to compare the performance method by solver $s$ with the best performance by any solver on problem $p$, and $\rho_s(\tau)$ is a factor of the best possible ratio. In general, solvers with high vaules of $\rho_s(\tau)$ or in the upper right of the image represent the best solver.
Figure 1 and Figure 2 plots the performance profiles of the SpMMSMS, JYJLL, MFR, SCD, and NPRP methods based on the number of iteration and CPU time, respectively. From the pictures below, we can be seen that from the left side of Figure 1 and Figure 2; the SpMMSMS method is the high-performance method in solving of 98 the test problems. We can also see in Table 3; the SpMMSMS method has the best performance in a total of NOI and the total number of CPU times compared to the other methods. SpMMSMS method has the ability to solve all problems, so the percentage reaches 100%. All comparisons for performance profiles, the total number of NOI, the total number of CPU, and the successful percentage indicate that the SpMMSMS method has a high performance computational compared to the other methods.
5. Conclusion

In this article, we propose a new spectral conjugate gradient method, namely SpMMSMS method. The SpMMSMS method’s performance was tested by comparing it to the previous methods (JYJLL, MFR, SCD, and NPRP). Our new method fulfills the descent condition and global convergence property under the strong Wolfe line search. Through 98 test problems, the SpMMSMS method has a high performance computational compared with the JYJLL, MFR, SCD, and NPRP method.

Conflict of Interests

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

References

[1] J. Guo, Z. Wan, A Modified Spectral PRP Conjugate Gradient Projection Method for Solving Large-Scale Monotone Equations and Its Application in Compressed Sensing, Math. Probl. Eng. 2019 (2019), Article ID 5261830.

[2] A.M. Awwal, P. Kumam, A.B. Abubakar, Spectral modified Polak-Ribiére-Polyak projection conjugate gradient method for solving monotone systems of nonlinear equations, Appl. Math. Comput. 362 (2019), 124514.

[3] G. Yuan, T. Li, W. Hu, A conjugate gradient algorithm for large-scale nonlinear equations and image restoration problems, Appl. Numer. Math. 147 (2020), 129-141.

[4] R. Pytlak, Conjugate Gradient Algorithms in Nonconvex Optimization, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[5] J. Nocedal, S.J. Wright, Numerical optimization, 2nd ed, Springer, New York, 2006.

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

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

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

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

[10] Z. Wei, S. Yao, L. Liu, The convergence properties of some new conjugate gradient methods, Appl. Math. Comput. 183 (2) (2006), 1341-1350.

[11] M. Raydan, The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem, SIAM J. Optim. 7 (1) (1997), 26-33.
[12] J. Barzilai, J. M. Borwein, Two-point step size gradient methods, IMA J. Numer. Anal. 8 (1988), 141-148.
[13] E.G. Birgin, J.M. Martínez, A spectral conjugate gradient method for unconstrained optimization, Appl. Math. Optim. 43 (2) (2001), 117-128.
[14] L. Zhang, W. Zhou, D. Li, Global convergence of a modified Fletcher-Reeves conjugate gradient method with Armijo-type line search, Numer. Math. 104 (4) (2006), 561-572.
[15] J. Liu, Y. Jiang, Global convergence of a spectral conjugate gradient method for unconstrained optimization, Abstr. Appl. Anal. 2012 (2012), Article ID 758287.
[16] J. Jian, L. Yang, X. Jiang, P. Liu, M. Liu, A Spectral Conjugate Gradient Method with Descent Property, Mathematics, 8 (2) (2020), 280.
[17] N. Andrei, New acceleration conjugate gradient algorithms as a modification of Dai-Yuan’s computational scheme for unconstrained optimization, J. Comput. Appl. Math. 234 (2010), 3397-3410.
[18] C.X. Kou, Y.H. Dai, A modified self-scaling memoryless Broyden-Fletcher-Goldfarb-Shanno method for unconstrained optimization, J. Optimiz. Theory. Appl. 165 (2015), 209-224.
[19] W.W. Hager, H. Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM J. Optim. 16 (2005), 170-192.
[20] J.K. Liu Y.M. Feng, L.M. Zou, A spectral conjugate gradient method for solving large-scale unconstrained optimization, Comput. Math. Appl. 77 (2019), 731-739.
[21] L. Zhang, An improved Wei-Yao-Liu nonlinear conjugate gradient method for optimization computation, Appl. Math. Comput. 215 (6) (2009), 2269-2274.
[22] Y.-H. Dai, Nonlinear Conjugate Gradient Methods, in: Wiley Encyclopedia of Operations Research and Management Science, John Wiley & Sons, Inc., Hoboken, NJ, USA, 2011: p. eorms0183.
[23] G. Zoutendijk, Nonlinear programming, computational methods, in Integer and Nonlinear Programming, J. Abadie (Ed.), North-Holland: Amsterdam, 1970, pp. 37-86.
[24] N. Andrei, An unconstrained optimization test functions collection, Adv. Model. Optim. 10 (1) (2008), 147-161.
[25] M. Malik, M. Mamat, S.S. Abas, Sukono, Convergence analysis of a new coefficient conjugate gradient method under exact line search, Int. J. Adv. Sci. Technol. 29 (5) (2020), 187-198.
[26] M. Malik, S.S. Abas, M. Mamat, Sukono, I.S. Mohammed, A new hybrid conjugate gradient method with global convergence properties, Int. J. Adv. Sci. Technol. 29 (5) (2020), 199-210.
[27] E.D. Dolan, J.J. Moré, Benchmarking optimization software with performance profiles, Math. Program. 91 (2) (2002), 201-213.