Journal of Decision Science and Optimization
Vol. 2 (2026), No. 2, pp. 141-150
DOI: https://doi.org/10.55578/jdso.2610.010

ARTICLE

Two Scaling Conjugate Gradient Methods Using the Approximate Optimal Stepsizes for Unconstrained Optimization Problem

Yongjin Kim[1], Yunchol Jong[1,*], Pyongil Choe[1]

[1] *Department of Mathematics, University of Science, Pyongyang, DPR Korea*

*Corresponding author. Email: jyc1963@star-co.net.kp

Received: 29 June 2026, Revised: 11 Sept 2026, Accepted: 20 Sept 2026, Published: 13 Oct 2026

Abstract

This paper proposes a new method, which combines the gradient method using the approximately optimal stepsizes with the Barzilai and Borwein scaling conjugate gradient method, and propose two scaling conjugate gradient methods for nonlinear unconstrained optimization problems. And then, the sufficient descent property and the global convergence are proved under some reasonable conditions and the standard Wolfe line search. We give some preliminary numerical results to show the effectiveness of the proposed method.

Keywords: Unconstrained optimization, approximate optimal stepsize, Barzilai-Borwein method, scaling conjugate gradient method, global convergence

2000 MSC: 2020 90C06, 65K05

1. INTRODUCTION

Consider the unconstrained optimization problem

$$\min_{x\in R^n} f(x) \tag{1.1}$$

where $f: R^n \to R$ is continuously differentiable and bounded below.

We first give a review of the gradient method for the strictly convex quadratic minimization problem in which

$$f(x) = \frac{1}{2}x^T Ax - b^T x,$$

where $A \in R^{n\times n}$ is a symmetric and positive definite matrix and $x, b \in R^n$ are vectors. Let x_k be the current iteration point, g_k the gradient of $f(x)$ at x_k, and $s_{k-1} = x_k - x_{k-1}, y_{k-1} = g_k - g_{k-1}$.

The gradient method for (1.1) is of the form

$$x_{k+1} = x_k - \alpha_k g_k, k = 0,1, ..., \tag{1.2}$$

where α_k is a stepsize depending on the line search. In the classical gradient method, the stepsize α_k is determined by

$$\alpha_k^{SD} = \frac{g_k^T g_k}{g_k^T A g_k}.$$

However, it shows poor performance and is badly affected by ill-conditioning. Therefore, many researchers have made great effort to overcome these drawbacks [1-4].

An efficient gradient method called BB method was proposed by Barzilai and Borwein [1] which makes use of some quasi-Newton properties.

https://doi.org/10.55578/jdso.2610.010
© 2026 The Authors. Published by Nexus Press B.V.
This is an open access article under the CC BY 4.0 license (https://creativecommons.org/licenses/by/4.0/).

A stepsize α_k of BB method for solving (1.1) is

$$\alpha_k^{BB1} = \frac{\| s_{k-1} \|^2}{s_{k-1}^T y_{k-1}}, \text{ or } \alpha_k^{BB2} = \frac{s_{k-1}^T y_{k-1}}{\| y_{k-1} \|^2},$$

which was obtained by solving

$$\min_{\alpha>0} \left\| \frac{1}{\alpha} s_{k-1} - y_{k-1} \right\|^2, \text{ or } \min_{\alpha>0} \| \alpha y_{k-1} - s_{k-1} \|^2.$$

We can see without any difficulty that the idea of the BB method is to approximate the Hessian matrix by the scalar matrix $\frac{1}{\alpha_k^{BB1}} I$ or $\frac{1}{\alpha_k^{BB2}} I$. We know that

$$\lambda_{\min}(A) \leq \frac{1}{\alpha_k^{BB1}} \leq \frac{1}{\alpha_k^{BB2}} \leq \lambda_{\max}(A),$$

where $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ are the smallest and largest eigenvalues of A, respectively [17]. The important feature of the BB gradient method is that it does not need the Hessian matrix computation but possesses some approximate quasi-Newton property. Numerical experiments demonstrated that the BB method requires less computational efforts than general gradient methods but it has better performance [1,5-8]. The BB method was proved to have global convergence and linear convergence rate [3,9,24] for strictly convex quadratic function minimization. Raydan [6] proposed a global BB method for general nonlinear unconstrained optimization by incorporating the nonmonotone line search (GLL line search) [10].

The numerical results of [6] showed that the BB method has superior performance to some classical conjugate gradient methods. Since then, some modified BB stepsizes have been developed for unconstrained optimization (1.1). Dai et al. [11] presented the cyclic BB method for unconstrained optimization. By using the interpolation procedure, Dai et al. [12] presented two modified BB stepsizes for gradient methods. Xiao et al. [13] designed four modified BB stepsizes for gradient methods and their BB method is based on some modified secant equations. However, it is still worth while designing more robust and more efficient gradient methods for the problem (1.1) [14].

The authors of [15,16] introduced a concept of approximate optimal stepsize for the gradient method and proposed a new gradient method using the approximate optimal stepsize for large unconstrained optimization.

Definition 1. *([15,16]) Let $\phi(\alpha)$ be an approximate model of $f(x_k - \alpha g_k)$. A positive constant $\alpha^* = arg\ min_{\alpha>0} \phi(\alpha)$ is called approximate optimal stepsize associated to $\phi(\alpha)$ for gradient method.*

The approximate optimal stepsize is easy to calculate, but α_k^{SD} results in expensive computational cost. The stepsizes $\alpha_k^{SD}, \alpha_k^{BB1}$, and α_k^{BB2} are easily verified as approximate optimal stepsizes. In any gradient method for strictly convex quadratic minimization problems, the stepsize α_k can also be obtained by minimizing the following quadratic approximation model:

$$\phi(\alpha) = f(x_k) - \alpha \| g_k \|^2 + \frac{1}{2} \alpha^2 g_k^T \left(\frac{1}{\mu_k} I \right) g_k,$$

where $\frac{1}{\mu_k} I$ is an approximation to the Hessian matrix of f at x_k. The obtained stepsize α_k is the approximate optimal stepsize associated with the above mentioned $\phi_k(\alpha)$ for gradient method.

Let us consider the BB gradient method. Suppose that $s_{k-1}^T y_{k-1} > 0$ and we take

$$\phi(\alpha) = f(x_k) - \alpha \| g_k \|^2 + \frac{1}{2} \alpha^2 g_k^T \frac{s_{k-1}^T y_{k-1}}{\| s_{k-1} \|^2} I g_k, \tag{1.3}$$

as the approximation model of $f(x_k - \alpha g_k)$. In (1.3), $\frac{s_{k-1}^T y_{k-1}}{\|s_{k-1}\|^2} I$ is an approximation to the Hessian matrix. By solving the equation $\frac{d\phi}{d\alpha} = 0$ with α, we also obtain the BB stepsize α_k^{BB1}, which satisfies $\alpha_k^{BB1} =$

$\arg\min_{\alpha>0}\phi(\alpha)$. Therefore, the BB stepsize α_k^{BB1} is the approximate optimal stepsize associated to the approximation model $\phi(\alpha)$ in (1.3).

From the definition of approximate optimal stepsize, we see that its effectiveness depends heavily on approximation model, and approximation model is essential factor to improve the numerical performance of gradient method using approximate optimal stepsize.

Let us consider the quadratic model:

$$\phi(\alpha) = f(x_k) - \alpha \parallel g_k \parallel^2 + \frac{1}{2}\alpha^2 g_k^T B_k g_k, \tag{1.4}$$

where B_k is a symmetric positive definite matrix. To reduce the storage and computational amount, Liu et al.[15,16] updated a scalar matrix approximating the Hessian matrix by the BFGS formula. By taking the scalar matrix $D_k = \bar{\xi}\frac{y_{k-1}^T y_{k-1}}{s_{k-1}^T y_{k-1}} I$ with $\bar{\xi} \geq 1$, Zhang et al [18] determined the B_k by the following formula:

$$B_k = D_k - \frac{D_k s_{k-1} s_{k-1}^T D_k}{s_{k-1}^T D_k s_{k-1}} + \frac{\bar{y}_{k-1}\bar{y}_{k-1}^T}{s_{k-1}^T \bar{y}_{k-1}},$$

where $\bar{y}_{k-1} = y_{k-1} + \frac{r_k}{\|s_{k-1}\|^2} s_{k-1}$ and $r_k = 3(g_k + g_{k-1})^T s_{k-1} + 6(f_{k-1} - f_k)$. Also, they replaced r_k with

$$\bar{r}_k = \min\{\max,\{\bar{r}_k, -\bar{\eta} s_{k-1}^T y_{k-1}\}\bar{\eta} s_{k-1}^T y_{k-1}\},$$

in order to improve the numerical performance, where $0 < \bar{\eta} < 0.1$. The above expression implies that $s_{k-1}^T \bar{y}_{k-1} = s_{k-1}^T y_{k-1} + \bar{r}_k \geq (1-\bar{\eta}) s_{k-1}^T y_{k-1}$ when $s_{k-1}^T y_{k-1} > 0$, from which we have the following lemma.

Lemma 1. ([15]) If $s_{k-1}^T y_{k-1} > 0$, we have that $s_{k-1}^T \bar{y}_{k-1} > 0$ and B_k is symmetric positive definite.

The optimality condition $\frac{d\phi}{d\alpha} = 0$ results in

$$\alpha_k = \frac{g_k^T g_k}{g_k^T B_k g_k} \tag{1.5}$$

$$= \frac{g_k^T g_k}{\frac{\bar{\xi} \parallel y_{k-1} \parallel^2}{s_{k-1}^T y_{k-1}}\left(\parallel g_k \parallel^2 - \frac{(g_k^T s_{k-1})^2}{\parallel s_{k-1} \parallel^2}\right) + \left(\frac{g_k^T y_{k-1}}{(s_{k-1}^T \bar{y}_{k-1})^{1/2}} + \frac{\bar{r}_k g_k^T s_{k-1}}{(s_{k-1}^T \bar{y}_{k-1})^{1/2} \parallel s_{k-1} \parallel^2}\right)^2}$$

By $s_{k-1}^T y_{k-1} > 0$ and Lemma 1 the stepsize α_k satisfies $\alpha_k = \arg\min_{\alpha>0}\phi(\alpha)$ and then α_k is the approximate optimal stepsize associated to $\phi(\alpha)$.

Numerical experiments showed that the bound $[\alpha_k^{BB2}, \alpha_k^{BB1}]$ for α_k in (1.5) is desirable. Therefore, [15] used the following stepsize in case of $s_{k-1}^T y_{k-1} > 0$:

$$\alpha_k^{(1)} = \max\{\min\{\alpha_k, \alpha_k^{BB1}\}, \alpha_k^{BB2}\}. \tag{1.6}$$

Liu et al. [16] suggested another approximate optimal stepsize based on the following BFGS formula [19]:

$$B_k = D_k - \frac{D_k s_{k-1} s_{k-1}^T D_k}{s_{k-1}^T D_k s_{k-1}} + t_k \frac{y_{k-1} y_{k-1}^T}{s_{k-1}^T y_{k-1}}, \tag{1.7}$$

where $D_k = \bar{\xi}\frac{y_{k-1}^T y_{k-1}}{s_{k-1}^T y_{k-1}} I$ with $1 \leq \bar{\xi} \leq 1.5$, and $t_k = 2(f_{k-1} - f_k + g_k^T s_{k-1})/s_{k-1}^T y_{k-1}$.

For the positive definiteness of B_k and the desirable numerical performance, t_k is usually taken as

$$\bar{t}_k = \min\{\max,\{t_k, 0.05\}10\} \tag{1.8}$$

in practice. Again, consider the quadratic approximate model (1.4). In (1.7), the positive definiteness B_k is preserved [16]. They substituted (1.7) into (1.4) and obtained the approximate optimal stepsize by $\frac{d\phi}{d\alpha} = 0$,

(1.9)

$$\alpha_k = \frac{g_k^T g_k}{\frac{\bar{\xi} \parallel y_{k-1} \parallel^2}{s_{k-1}^T y_{k-1}} \left(\parallel g_k \parallel^2 - \frac{(g_k^T s_{k-1})^2}{\parallel s_{k-1} \parallel^2} \right) + \bar{t}_k \frac{(g_k^T y_{k-1})^2}{s_{k-1}^T y_{k-1}}},$$

In practice, the stepsize is taken similarly to (1.6):

$$\alpha_k^{(2)} = \max\{\min\{\alpha_k, \alpha_k^{BB1}\}, \alpha_k^{BB2}\}. \tag{1.10}$$

On the other hand, the simplicity and numerical efficiency of the BB method and the conjugate gradient method encouraged to develop the methods that combine the conjugate gradient method with the BB gradient method [17,19,21]. Dai and Kou [21] developed some Barzilai-Borwein conjugate gradient (BBCG) methods for strictly convex quadratic minimization, and showed that BBCG is the most efficient through numerical results. Liu et al.[17] proposed an efficient BBCG method for general unconstrained optimization problem (1.1). They proved that the proposed method has global convergence and R-linear convergence. Some numerical results showed that the method is competitive to CG-DESCENT:

$$d_{k+1} = -g_{k+1} + \bar{\beta}_k^N d_k, \; d_0 = -g_0, \tag{1.11}$$

where

$$\bar{\beta}_k^N = \max\{\beta_k^N, \eta_k\}, \; \eta_k = \frac{-1}{\parallel d_k \parallel \min\{\eta, \parallel g_k \parallel\}}, \tag{1.12}$$

η is a constant, and

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

and CGOPT.

Wang et al.[20] proposed a BB scaling conjugate gradient method as another BBCG for the problem (1.1). It seems that the new approach will be superior to both of the BB method and the CG method. BB scaling conjugate gradient method proposed in [20] is as follows,

$$d_{k+1} = -\lambda_k g_{k+1} + \bar{\beta}_k^N d_k, \; d_0 = -g_0,$$

where $\bar{\beta}_k^N$ is defined by (1.12) and λ_k is computed by the BB gradient method, i.e. $\lambda_k = \frac{1}{\theta_k}$ and $\theta_k = \frac{s_{k-1}^T y_{k-1}}{s_{k-1}^T s_{k-1}}$.

Inspired by the BB scaling conjugate gradient method and the approximately optimal stepsize, in this paper, we propose two scaling conjugate gradient methods that possess the approximate optimal stepsizes, and establish the global convergence under weak conditions.

The rest of this paper is organized as follows. Section 2 presents two scaling conjugate gradient methods with the approximate optimal stepsizes. In Section 3 we consider global convergence of the algorithms proposed in Sect. 2 under the Wolfe line search. Numerical experiments are carried out to demonstrate the efficiency of our algorithms in Section 4, and finally, Section 5 gives some conclusions.

2. SCALING CONJUGATE GRADIENT METHODS WITH THE APPROXIMATE OPTIMAL STEPSIZES (AOS-CG)

Scaling conjugate gradient methods with the approximate optimal stepsizes, in short, AOS-CG is as follows,

$$d_{k+1} = -\alpha_{k+1}^{(i)} g_{k+1} + \bar{\beta}_k^N d_k, i = 1,2, d_0 = -g_0, \tag{2.1}$$

where $\bar{\beta}_k^N$ is defined by (1.12) and $\alpha_{k+1}^{(i)} (i, = 12)$ is computed by (1.6) or (1.10).

Algorithm 1(Scaling conjugate gradient method with the approximate optimal stepsize $\alpha_k^{(1)}$ (AOS-CG1))

Step 0. Given $x_0 \in R^n$, $\varepsilon > 0, 0 < \delta \leq \sigma < 1, \eta > 0, \frac{1}{8} < \xi < 1, 0 < \tau < 8, 0 < \bar{\eta} < 0.1, 1 \leq \bar{\xi} \leq 1.5$.

Step 1. Compute $g_0 = \nabla f(x_0)$. Set $d_0 = -g_0, k = 0$.

Step 2. If $\| g_k \| \leq \varepsilon$, stop.

Step 3. Compute λ_k by the Wolfe line search:

$$f(x_k + \lambda_k d_k) \leq f(x_k) + \delta\lambda_k \nabla f(x_k)^T d_k, \tag{2.2}$$
$$\nabla f(x_k + \lambda_k d_k)^T d_k \geq \sigma \nabla f(x_k)^T d_k. \tag{2.3}$$

and set $x_{k+1} = x_k + \lambda_k d_k$.

Step 4. Compute

$$\beta_k^N = \frac{1}{d_k^T y_k}\left(y_k - 2d_k \frac{\| y_k \|^2}{d_k^T y_k}\right)^T g_{k+1}, \eta_k = \frac{-1}{\| d_k \| \min\{\eta, \| g_k \|\}},$$
$$\bar{\beta}_k^N = \max\{\beta_k^N, \eta_k\}, \alpha_{k+1}^{BB1} = \frac{\| s_k \|^2}{s_k^T y_k}, \alpha_{k+1}^{BB2} = \frac{s_k^T y_k}{\| y_k \|^2}.$$

Step 5. Compute the scaling parameter $\boldsymbol{\alpha_{k+1}^{(1)}}$:

$$\alpha_{k+1}^{(1)} = \max\{\min\{\alpha_{k+1}, \alpha_{k+1}^{BB1}\}, \alpha_{k+1}^{BB2}\},$$

where α_{k+1} is computed by (1.5) and

$$\bar{y}_k = y_k + \frac{\bar{r}_{k+1}}{\| s_k \|^2} s_k, \bar{r}_{k+1} = \min\{\max,\{r_{k+1}, -\bar{\eta} s_k^T y_k\}\bar{\eta} s_k^T y_k\},$$
$$r_{k+1} = 3(g_{k+1} + g_k)^T s_k + 6(f_k - f_{k+1}), 0 < \bar{\eta} < 0.1.$$

If the condition $\alpha_{k+1}^{(1)} \leq \xi$ or $\alpha_{k+1}^{(1)} \geq \frac{1}{\xi}$ holds, then set $\alpha_{k+1}^{(1)} = \frac{1}{\tau}$.

Step 6. Compute the search direction d_{k+1}:

$$d_{k+1} = -\alpha_{k+1}^{(1)} g_{k+1} + \bar{\beta}_k^N d_k.$$

Step 7. If the condition $| g_{k+1}^T g_k | > 0.2 \| g_{k+1} \|^2$ holds, then set $d_{k+1} = -g_{k+1}$.

Step 8. Set $k = k + 1$ and go to Step 2.

Algorithm 2(Scaling conjugate gradient method with the approximate optimal stepsize $\boldsymbol{\alpha_k^{(2)}}$ (AOS-CG2))

This algorithm is constructed by modifying Step 5 and Step 6 of Algorithm 1 as following:

Step 5. Compute the scaling parameter $\alpha_{k+1}^{(2)}$:

$$\alpha_{k+1}^{(2)} = \max\{\min\{\alpha_{k+1}, \alpha_{k+1}^{BB1}\}, \alpha_{k+1}^{BB2}\},$$

where α_{k+1} is computed by (1.9).

If the condition $\alpha_{k+1}^{(2)} \leq \xi$ or $\alpha_{k+1}^{(2)} \geq \frac{1}{\xi}$ holds, then set $\alpha_{k+1}^{(2)} = \frac{1}{\tau}$.

Step 6. Compute the search direction d_{k+1}:

$$d_{k+1} = -\alpha_{k+1}^{(2)} g_{k+1} + \bar{\beta}_k^N d_k.$$
$$\bar{\mu}_k = \begin{cases} -g_k, & k = 0, \\ -g_k + \beta_k d_{k-1}, & k > 0, \end{cases} \tag{2.4}$$

3. CONVERGENCE ANALYSIS

In this section, we consider global convergence of algorithms AOS-CG1 and AOS-CG2 at a time. The convergence analysis is similar to [20] and is based on [23] and [22] for the scheme (1.11). The global convergence is considered under the following assumptions:

Assumption 1. *A1.* $f: R^n \to R$ is continuously differentiable and bounded below. *A2.* The level set $\mathcal{L} = \{x \in R^n : f(x) \leq f(x_0)\}$ is bounded. *A3.* ∇f satisfies the Lipschitz condition

$$\forall x, y \in \mathcal{L}, \| \nabla f(x) - \nabla f(y) \| \leq L \| x - y \|,$$

where L is the Lipschitz constant.

By a slight modification of lemma 3.1 of [20], we can show the sufficient descent property of d_k.

Lemma 2. *If $d_k^T y_k \neq 0$ and d_k is generated by (2.1), then we have $g_{k+1}^T d_{k+1} \leq -q_1 \parallel g_{k+1} \parallel^2$, where $q_1 = \min\left\{\xi, \frac{1}{\tau}\right\} - \frac{1}{8}$.*

Proof. Since $d_0 = -g_0$, we have $g_0^T d_0 = -\parallel g_0 \parallel^2 < 0$. In case of $k \geq 1$, it follows from (2.1) that

$$g_{k+1}^T d_{k+1} = -\alpha_{k+1}^{(i)} \parallel g_{k+1} \parallel^2 + \bar{\beta}_k^N g_{k+1}^T d_k$$
$$= -\left(\alpha_{k+1}^{(i)} - 1\right) \parallel g_{k+1} \parallel^2 - \parallel g_{k+1} \parallel^2 + \bar{\beta}_k^N g_{k+1}^T d_k, i = 1,2.$$

By Theorem 1.1 of [22], we have

and then

$$g_{k+1}^T d_{k+1} \leq -\left(\alpha_{k+1}^{(i)} - 1\right) \parallel g_{k+1} \parallel^2 - \frac{7}{8} \parallel g_{k+1} \parallel^2$$
$$= -\left(\alpha_{k+1}^{(i)} - \frac{1}{8}\right) \parallel g_{k+1} \parallel^2, i = 1,2.$$

From Step 5 of Algorithm 1(or Algorithm 2), it follows that

$$\boldsymbol{\alpha_{k+1}^{(i)} \geq \min\left\{\xi, \frac{1}{\tau}\right\}, i = 1,2.}$$

If $\xi > \frac{1}{\tau}$, then $\alpha_{k+1}^{(i)} \geq \frac{1}{\tau} > \frac{1}{8}, i = 1,2$. If $\frac{1}{\tau} > \xi$, then $\alpha_{k+1}^{(i)} \geq \xi > \frac{1}{8}, i = 1,2$. Thus it holds always that $\alpha_{k+1}^{(i)} > \frac{1}{8}, i = 1,2$, which implies that

$$g_{k+1}^T d_{k+1} \leq -\left(\alpha_{k+1}^{(i)} - \frac{1}{8}\right) \parallel g_{k+1} \parallel^2 \leq -\left(\min\left\{\xi, \frac{1}{\tau}\right\} - \frac{1}{8}\right) \parallel g_{k+1} \parallel^2$$

Hence,

$$g_{k+1}^T d_{k+1} \leq -q_1 \parallel g_{k+1} \parallel^2,$$

which completes the proof.

Remark 1. *From Lemma 2 we can see that the degree of sufficient descent property depends on ξ and τ. This shows the importance of choice of ξ and τ.*

Lemma 3. [20]. *Suppose that the conditions of Lemma 2 are satisfied and the Assumption A3 holds. If λ_k is determined by the Wolfe conditions (2.2) and (2.3), then we have*

$$\lambda_k \geq \frac{1-\sigma}{L} \frac{| g_k^T d_k |}{\parallel d_k \parallel^2}.$$

Lemma 4. [20]. *If the assumptions A1-A3 of Assumption 1 are satisfied, then for the schemes (1.12), (1.13) and (2.1), we have*

$$d_k \neq 0 \ \forall k \text{ and } \sum_{k=0}^{\infty} \parallel u_{k+1} - u_k \parallel^2 < \infty$$

with $u_k = \frac{d_k}{\|d_k\|}$, whenever $\inf\{\parallel g_k \parallel : k \geq 0\} > 0$.

Theorem 1. *If the assumptions A1-A3 hold, then, for the scheme (1.12), (1.13) and (2.1), there exists some k such that $g_k = 0$, otherwise, $\liminf_{k\to\infty} \parallel g_k \parallel = 0$.*

Proof. This theorem is a consequent of Lemma 2 - Lemma 4. Since the proof is similar to Theorem 3.2 in [22], it is omitted.

4. NUMERICAL EXPERIMENTS

In this section is given some numerical experiment results to show the practical efficiency of our proposed AOS-CG1 and AOS-CG2 methods. Numerical experiments were performed using MATLAB R2013a on Intel Core i5 CPU U560@ 1.33GHz, 1.33 GHz, 1.86 GB RAM personal computer.

The test problems used in the numerical experiments are given in Table 1. Our algorithms AOS-CG1 and AOS-CG2 are compared with the algorithms AOS-Gradient1 [15], AOS-Gradient2 [16], LMYCD1 [25], LMYCD2 [25], MS-SSML-BFGS [26], SSML-BFGS [27] and BBSCG [20]. Each problem was tested with the different values of n.

We set parameters of the algorithms AOS-CG1 and AOS-CG2 as follows:

$$\varepsilon = 10^{-6},\ \delta = 0.1,\ \sigma = 0.9,\ \eta = 0.01,\ \bar{\eta} = 0.01,\ \xi = 0.25,\ \tau = 4,\ \bar{\xi} = 1.07,$$

which are regarded as the most appropriate values through several numerical experiments with various settings of the parameters. In the numerical experiments, parameters for the compared algorithms are set as reported in relevant references. For each test problem, all these algorithms is terminated if $\| g_k \| \leq \varepsilon$ or if iteration number exceeds 10000. Table 2 and Table 3 show the iteration numbers and CPU times for the nine algorithms, respectively.

In the tables below, let x_0 be the initial point and we regard any algorithm as failed if the algorithm did not stop within CPU time of 1200 seconds or did not find any global solution.

Table 1. Test problems(dimension, initial point) and its optimal values

No	Problem	Optimal Value
1	Trigonometric [7] (n = 100, 300, 500, 1000, 3000, $x0$ = $(1, ..., 1)^T/(5n)$)	0
2	n-Dimensions Sine-Square II [31] (n = 100, 300, 500, $x0$ = 2.5 × $(1, ..., 1)^T/(5n)$)	0
3	Rastrigin [31](n = 500, 1000, 2000, $x0$ = $(1, ..., 1)^T$)	0
4	Cube [7]($n = 2, x0 = (-1.2, -1)^T$)	0
5	Rosenbrock [7]($n = 2, x0 = (-1.2, 1)^T$)	0
6	Powell Singular [7]($n = 4, x0 = (3, -1, 0, -1)^T$)	0
7	Helical Valley [7]($n = 3, x0 = (-1, 0, 0)^T$)	0
8	Shekel [29](n = 500, 1000, $x0$ = $(e/5n)^T$)	0
9	Wood [7]($n = 4, x0 = (-3, -1, -3, -1)^T$	0
10	Six-hump camel back [31]($n = 2, x0 = (-1, 1)^T$)	-1.0316
11	Extended Rosenbrock [7](n = 100, 500, $x0$ = $(1.05, 1, ..., 1.05, 1)^T$)	0
12	Goldstein Price [31]($n = 2, x0 = (-0.2, -0.9)^T, (-0.2, -1)^T$)	3
13	Ackley [31](n = 100, $x0$ = 0.1 × $(1, 1, ..., 1)^T$)	0
14	Freudenstein-Roth [30](n = 100, $x0$ = $(-6, -1, ..., -6, -1)^T$)	0
15	Shubert [31]($n = 2, x0 = (-1.25, -1)^T$)	-186.7309

The algorithms in Tables 2 and 3 are labeled as follows.

1=MS-SSML-BFGS; 2=AOS-CG1; 3=AOS-CG2; 4=BBSCG; 5=AOS-Gradient1; 6=AOS-Gradient2; 7=LMYCD1; 8=LMYCD2; 9=SSML-BFGS.

Table 2. Iteration numbers for the problems 1-15

Problems	Methods								
No(n)	1	2	3	4	5	6	7	8	9
1(100)	22	6	6	19	5	5	5	5	5
1(300)	22	5	5	15	5	5	5	5	5
1(500)	19	7	7	13	5	5	5	5	5
1(1000)	21	7	7	9	5	5	5	5	5
1(3000)	13	5	5	5	5	5	5	5	5
2(100)	25	367	380	fail	91	101	23	23	23
2(300)	110	435	393	fail	33	12	24	24	24
2(500)	28	12	12	fail	fail	fail	fail	fail	fail
3(500)	fail	fail	fail	16	fail	fail	fail	fail	fail
3(1000)	fail	fail	fail	16	fail	fail	fail	fail	fail
3(2000)	fail	fail	fail	15	fail	fail	fail	fail	fail
4(2)	4679	2390	2390	2678	18505	27822	1107	2917	fail
5(2)	5320	470	470	1235	5386	575	793	486	fail
6(4)	753	647	647	2230	125	121	5836	672	2671

7(3)	1203	413	413	1178	fail	74	423	392	36
8(500)	37	27	27	126	fail	fail	fail	fail	fail
8(1000)	39	25	25	fail	fail	fail	fail	fail	fail
9(4)	3137	1304	1304	3484	4301	fail	fail	fail	fail
10(4)	26	15	15	81	15	fail	29	29	29
11(100)	52	49	49	188	31	34	fail	fail	fail
11(500)	58	46	46	184	62	63	fail	fail	fail
12(2)	24	17	17	65	14	10	20	23	25
13(100)	53	30	30	132	fail	fail	fail	fail	fail
14(100)	fail	2432	2432	1732	fail	fail	fail	fail	fail
15(2)	35	55	55	47	7	6	26	26	26

Table 3. CPU times for the problems 1-15

Problems	Methods								
$No(n)$	1	2	3	4	5	6	7	8	9
1(100)	0.4606	0.4142	0.1894	0.3840	0.0886	0.0868	0.1090	0.0958	0.1106
1(300)	2.0180	0.8040	0.8040	1.3633	0.4066	0.3935	0.4307	0.4276	0.4145
1(500)	3.6705	1.5155	1.4934	2.5379	0.8585	0.8332	1.1016	0.9088	0.9115
1(1000)	13.4214	5.0422	5.0106	5.7854	2.7367	2.7163	2.9879	2.9167	2.9573
1(3000)	65.6902	30.0019	33.4737	26.855	22.7034	25.7002	21.9682	23.5867	26.0803
2(100)	0.4669	7.9221	8.0995	fail	1.3748	1.5558	1.4854	0.4657	0.4649
2(300)	10.4968	54.63	39.8243	fail	2.2139	0.8666	2.1147	2.0432	2.0378
2(500)	5.6713	2.8778	2.8418	fail	fail	fail	fail	fail	fail
3(500)	fail	fail	fail	2.1151	fail	fail	fail	fail	fail
3(1000)	fail	fail	fail	5.8883	fail	fail	fail	fail	fail
3(2000)	fail	fail	fail	16.9867	fail	fail	fail	fail	fail
4(2)	5.3461	2.2939	2.1715	3.7249	5.1664	6.4652	0.4296	0.3595	fail
5(2)	5.8259	0.3848	0.3642	1.6650	1.4746	0.0182	0.3212	0.1946	fail
6(4)	0.8756	9.8525	9.4510	2.9068	0.0549	0.0450	3.2646	0.3726	1.5463
7(3)	1.4131	0.4071	0.3912	1.6225	fail	0.0250	0.2270	0.1946	0.0197
8(500)	77.0793	60.7186	61.4492	265.392	fail	fail	fail	fail	fail
8(1000)	318.4062	240.8221	233.7582	fail	fail	fail	fail	fail	fail
9(4)	3.9565	1.6929	1.3060	5.2033	1.6925	fail	fail	fail	fail
10(4)	0.0195	0.0167	0.0091	0.0712	0.0084	fail	0.1015	0.0273	0.0188
11(100)	4.7156	4.4575	4.4394	18.1966	2.0552	2.2247	fail	fail	fail
11(500)	109.508	90.5251	92.9638	361.0688	60.5119	61.6521	fail	fail	fail
12(2)	0.0253	0.0186	0.0148	0.0938	0.0082	0.0036	0.0165	0.0095	0.0111
13(100)	5.3104	3.1615	3.1586	13.8898	fail	fail	fail	fail	fail
14(100)	fail	131.5021	133.8927	95.0773	fail	fail	fail	fail	fail
15(2)	0.0488	0.0575	0.0588	0.0778	0.0072	0.0045	0.0255	0.0177	0.0185

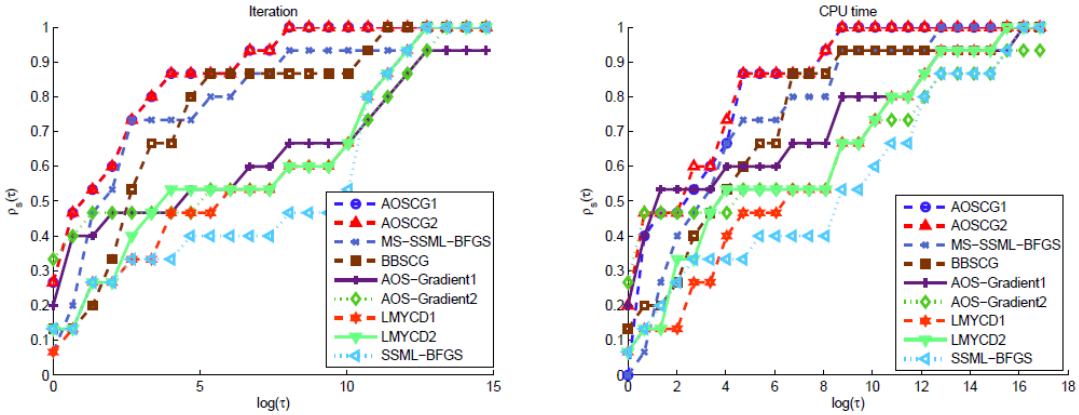


Figure 1. Performance profiles for iteration numbers and CPU times of tested algorithms

Table 2 and Table 3 show that our algorithms AOS-CG1 and AOS-CG2 outperform other algorithms for 8 problems among 15 test problems in both aspect of iteration number and CPU time. We employed the performance profiles of Dolan and Mor´e [28] to compare the performance of these 9 algorithms with respect to needed iteration numbers and CPU times. Figure 1 demonstrates that algorithms AOS-CG1 and

AOS-CG2 have the better performance compared with the other algorithms. From the figures, we can see that performances of the AOS-CG1 and AOS-CG2 are very similar for the given test problems.

5. CONCLUSIONS

In this paper, we presented two scaling conjugate gradient methods using the approximate optimal stepsizes given in [15,16] as scaling parameters for unconstrained optimization problem (1.1). The methods employ the Wolfe line search and have global convergence under reasonable assumptions. Numerical results show that our methods are prospective.

Acknowledgements: The authors would like to thank the anonymous referee for his/her deep insight, useful comments and suggestions, which led to the improvement of this paper.

Conflict of Interests: The authors declare no conflict of interests.

Data Availability Statement: The authors declare that all data in the paper are available.

REFERENCES

[1] Barzilai, J., & Borwein, J. (1988). Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, *8*(1), 141–148.
[2] Raydan, M., & Svaiter, B. (2002). Relaxed steepest descent and Cauchy-Barzilai-Borwein method. *Computational Optimization and Applications*, *21*(2), 155–167.
[3] Dai, Y., & Liao, L. (2002). R-linear convergence of the Barzilai and Borwein gradient method. *IMA Journal of Numerical Analysis*, *22*(1), 1–10.
[4] Dai, Y., & Yuan, Y. (2003). Alternate minimization gradient method. *IMA Journal of Numerical Analysis*, *23*, 377–393.
[5] Fletcher, R. (2005). On the Barzilai-Borwein method. *Optimization and Control Applications*, *56*, 235–256.
[6] Raydan, M. (1997). The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. *SIAM Journal on Optimization*, *7*(1), 26–33.
[7] Sun, W., & Yuan, Y. (2006). *Optimization theory and methods: Nonlinear programming*. Springer.
[8] Zhang, Y., Sun, W., & Qi, L. (2010). A nonmonotone filter Barzilai-Borwein method for optimization. *Asia Pacific Journal of Operations Research*, *27*(1), 55–69.
[9] Raydan, M. (1993). On the Barzilai and Borwein choice of steplength for the gradient method. *IMA Journal of Numerical Analysis*, *13*(3), 321–326.
[10] Grippo, L., Lampariello, F., & Lucidi, S. (1986). A nonmonotone line search technique for Newton's method. *SIAM Journal on Numerical Analysis*, *23*(4), 707–716.
[11] Dai, Y. H., Hager, W. W., & Schittkowski, K. (2006). The cyclic Barzilai-Borwein method for unconstrained optimization. *IMA Journal of Numerical Analysis*, *26*(3), 604–627.
[12] Dai, Y. H., Yuan, J. Y., & Yuan, Y. X. (2002). Modified two-point stepsize gradient methods for unconstrained optimization problems. *Computational Optimization and Applications*, *22*(1), 103–109.

[13] Xiao, Y. H., Wang, Q. Y., & Wang, D. (2010). Notes on the Dai-Yuan-Yuan modified spectral gradient method. *Journal of Computational and Applied Mathematics*, *234*(10), 2986–2992.
[14] Zhou, B., Gao, L., & Dai, Y. H. (2006). Gradient methods with adaptive stepsizes. *Computational Optimization and Applications*, *35*(1), 69–86.
[15] Liu, Z., & Liu, H. (2018). An efficient gradient method with approximate optimal stepsize for large-scale unconstrained optimization. *Numerical Algorithms*, *78*(1), 21–39. https://doi.org/10.1007/s11075-017-0365-2
[16] Liu, Z., & Liu, H. (2019). An efficient gradient method with approximately optimal stepsize based on tensor model for unconstrained optimization. *Journal of Optimization Theory and Applications*. https://doi.org/10.1007/s10957-019-01475-1
[17] Liu, H., & Liu, Z. (2018). An efficient Barzilai-Borwein conjugate gradient method for unconstrained optimization. *Journal of Optimization Theory and Applications*. https://doi.org/10.1007/s10957-018-1393-3
[18] Zhang, J. Z., Deng, N. Y., & Chen, L. H. (1999). New quasi-Newton equation and related methods for unconstrained optimization. *Journal of Optimization Theory and Applications*, *102*(1), 147–167.
[19] Yuan, Y. X. (1991). A modified BFGS algorithm for unconstrained optimization. *IMA Journal of Numerical Analysis*, *11*(3), 325–332.
[20] Wang, L., Sun, W., de Sampaio, R. J. B., & Yuan, J. (2015). A Barzilai and Borwein scaling conjugate gradient method for unconstrained optimization problems. *Applied Mathematics and Computation*, *262*, 136–144.
[21] Dai, Y. H., & Kou, C. X. (2016). A Barzilai-Borwein conjugate gradient method. *Science China Mathematics*, *59*(8), 1511–1524.
[22] Hager, W. W., & Zhang, H. C. (2005). A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM Journal on Optimization*, *16*(1), 170–192. https://doi.org/10.1137/030601880
[23] Gilbert, J. C., & Nocedal, J. (1992). Global convergence properties of conjugate gradient methods for optimization. *SIAM Journal on Optimization*, *2*(1), 21–42.
[24] Zheng, Y., & Zheng, B. (2017). A new modified Barzilai-Borwein gradient method for the quadratic minimization problem. *Journal of Optimization Theory and Applications*, *172*(1), 179–186.
[25] Liu, M., et al. (2020). Two new conjugate gradient methods for unconstrained optimization. *Complexity*, *2020*, Article 9720653. https://doi.org/10.1155/2020/9720653
[26] Kim, Y. J., Jong, Y. C., & Kim, Y. (2024). A self-scaling memoryless BFGS based conjugate gradient method using multi-step secant condition for unconstrained minimization. *Applied Mathematics*, *69*(6), 847–866.
[27] Lv, J., Deng, S., & Wan, Z. (2020). An efficient single-parameter scaling memoryless Broyden-Fletcher-Goldfarb-Shanno algorithm for solving large scale unconstrained optimization problems. *IEEE Access*, *8*. https://doi.org/10.1109/ACCESS.2020.2992340
[28] Dolan, E. D., & Moré, J. J. (2002). Benchmarking optimization software with performance profiles. *Mathematical Programming*, *91*(2), 201–213.
[29] Peng, Z., Wu, D., & Zheng, Q. (2013). A level-value estimation method and stochastic implementation for global optimization. *Journal of Optimization Theory and Applications*, *156*(3), 493–523.
[30] Hu, P., & Liu, X. Q. (2013). A nonmonotone line search slackness technique for unconstrained optimization. *Journal of Optimization Theory and Applications*, *158*(3), 773–786.
[31] Ali, M. M., Khompatraporn, C., & Zabinsky, Z. B. (2005). A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. *Journal of Global Optimization*, *31*(4), 635–672.