Computational Optimization and Applications
https://doi.org/10.1007/s10589-022-00349-z

On the acceleration of the Barzilai–Borwein method

Yakui Huang[1] · **Yu-Hong Dai**[2] · **Xin-Wei Liu**[1] · **Hongchao Zhang**[3]

Received: 29 November 2020 / Accepted: 4 January 2022

Abstract
The Barzilai–Borwein (BB) gradient method is efficient for solving large-scale unconstrained problems to modest accuracy due to its ingenious stepsize which generally yields nonmonotone behavior. In this paper, we propose a new stepsize to accelerate the BB method by requiring finite termination for minimizing the two-dimensional strongly convex quadratic function. Based on this new stepsize, we develop an efficient gradient method for quadratic optimization which adaptively takes the nonmonotone BB stepsizes and certain monotone stepsizes. Two variants using retard stepsizes associated with the new stepsize are also presented. Numerical experiments show that our strategies of properly inserting monotone gradient steps into the nonmonotone BB method could significantly improve its performance and our new methods are competitive with the most successful gradient descent methods developed in the recent literature.

Keywords Barzilai–Borwein method · Gradient methods · Finite termination · Quadratic optimization

Mathematics Subject Classification 90C20 · 90C25 · 90C30

✉ Yakui Huang
huangyakui2006@gmail.com

Yu-Hong Dai
dyh@lsec.cc.ac.cn

Xin-Wei Liu
mathlxw@hebut.edu.cn

Hongchao Zhang
hozhang@math.lsu.edu

[1] Institute of Mathematics, Hebei University of Technology, Tianjin 300401, China

[2] LSEC, ICMSEC, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China

[3] Department of Mathematics, Louisiana State University, Baton Rouge, LA 70803-4918, USA

Published online: 13 January 2022

1 Introduction

Gradient descent methods have been widely used for solving smooth unconstrained nonlinear optimization

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

by generating a sequence of iterates

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

where $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable, $g_k = \nabla f(x_k)$ and $\alpha_k > 0$ is the stepsize along the negative gradient direction. Different gradient descent methods would have different rules for determining the stepsize α_k. The classic steepest descent (SD) method proposed by Cauchy [3] determines its stepsize by the so-called exact line search

$$\alpha_k^{SD} = \arg\min_{\alpha\in\mathbb{R}} f(x_k - \alpha g_k). \tag{3}$$

Although the SD method locally has the most function value reduction along the negative gradient direction, it often performs poorly in practice. Theoretically, when f is a strongly convex quadratic function, i.e.

$$f(x) = \frac{1}{2}x^T Ax - b^T x, \tag{4}$$

where $b \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n\times n}$ is symmetric and positive definite, the SD method converges Q-linearly [1] and will asymptotically perform zigzag between two orthogonal directions [17, 23].

In 1988, Barzilai and Borwein [2] proposed the following two novel stepsizes that significantly improve the performance of gradient descent methods:

$$\alpha_k^{BB1} = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}} \qquad \text{and} \qquad \alpha_k^{BB2} = \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}, \tag{5}$$

where $s_{k-1} = x_k - x_{k-1}$ and $y_{k-1} = g_k - g_{k-1}$. Clearly, when $s_{k-1}^T y_{k-1} > 0$, one has $\alpha_k^{BB1} \geq \alpha_k^{BB2}$. Hence, α_k^{BB1} is often called the *long* BB stepsize while α_k^{BB2} is called the *short* BB stepsize. When the objective function is quadratic (4), the stepsize α_k^{BB1} will be exactly the steepest descent stepsize but with one step retard, while α_k^{BB2} will be just the retard stepsize of minimal gradient (MG) method [9], that is

$$\alpha_k^{BB1} = \frac{g_{k-1}^T g_{k-1}}{g_{k-1}^T A g_{k-1}} = \alpha_{k-1}^{SD} \qquad \text{and} \qquad \alpha_k^{BB2} = \frac{g_{k-1}^T A g_{k-1}}{g_{k-1}^T A^2 g_{k-1}} = \alpha_{k-1}^{MG}.$$

It is proved that the Barzilai–Borwein (BB) method converges R-superlinearly for minimizing two-dimensional strongly convex quadratic functions [2]. Moreover, for the general n-dimensional case, the BB method is globally convergent [27] and the rate is R-linear [8]. Although the BB method does not decrease the objective

function value monotonically, extensive numerical experiments show that it performs much better than the SD method [14, 28, 34]. And it is commonly accepted that when a loose accuracy is required, BB-type methods could be even competitive with nonlinear conjugate gradient (CG) methods for solving smooth unconstrained optimization [14, 28]. Furthermore, by combining with gradient projection techniques, BB-type methods have a great advantage of easy extension to solve a wide class of constrained optimization, for example the bound or simplex constrained optimization [5]. Hence, BB-type methods enjoy many important applications, such as image restoration [31], signal processing [26], eigenvalue problems [25], nonnegative matrix factorization [24], sparse reconstruction [32], machine learning [29], etc.

Recently, Yuan [33, 34] proposed a gradient descent method which combines a new stepsize

$$\alpha_k^Y = \frac{2}{\frac{1}{\alpha_{k-1}^{SD}} + \frac{1}{\alpha_k^{SD}} + \sqrt{\left(\frac{1}{\alpha_{k-1}^{SD}} - \frac{1}{\alpha_k^{SD}}\right)^2 + \frac{4\|g_k\|^2}{(\alpha_{k-1}^{SD}\|g_{k-1}\|)^2}}}, \tag{6}$$

in the SD method so that the new method enjoys finite termination when minimizing a two-dimensional strongly convex quadratic function. Based on this new stepsize α_k^Y, Dai and Yuan [10] further developed the DY method, which alternately employs α_k^{SD} and α_k^Y stepsizes as follows:

$$\alpha_k^{DY} = \begin{cases} \alpha_k^{SD}, & \text{if } \mathrm{mod}(k,4) < 2; \\ \alpha_k^Y, & \text{otherwise.} \end{cases} \tag{7}$$

It is easy to see that $\alpha_k^Y \le \alpha_k^{SD}$. Hence, the DY method (7) is a monotone method. Moreover, it is shown that the DY method (7) performs even better than the nonmonotone BB method [10].

The property of nonmonotonically reducing objective function values is an intrinsic feature that causes the efficiency of the BB method. However, it is also pointed out by Fletcher [15] that retaining monotonicity is important for a gradient method, especially for minimizing general objective functions. On the other hand, although the monotone DY method performs well, using α_k^{SD} and α_k^Y in a nonmonotone fashion may yield better performance, see [11] for example. Moreover, it is usually difficult to compute the exact monotone stepsize α_k^{SD} in general optimization. Hence, in this paper, motivated by the great success of the BB method and the previous considerations, we want to further improve and accelerate the *nonmonotone* BB method by incorporating some *monotone* steps. For a more general and uniform analysis, we consider to accelerate the class of gradient descent methods (2) for quadratic function (4) using the following stepsize:

$$\alpha_k(\Psi(A)) = \frac{g_{k-1}^T \Psi(A) g_{k-1}}{g_{k-1}^T \Psi(A) A g_{k-1}}, \tag{8}$$

where $\Psi(\cdot)$ is a real analytic function on $[\lambda_1, \lambda_n]$ that can be expressed by a Laurent series

$$\Psi(z) = \sum_{k=-\infty}^{\infty} c_k z^k, \quad c_k \in \mathbb{R},$$

such that $0 < \sum_{k=-\infty}^{\infty} c_k z^k < +\infty$ for all $z \in [\lambda_1, \lambda_n]$. Here, λ_1 and λ_n are the smallest and largest eigenvalues of A, respectively. Clearly, the class of gradient methods (8) are generally nonmonotone and the two BB stepsizes α_k^{BB1} and α_k^{BB2} can be obtained by setting $\Psi(A) = I$ and $\Psi(A) = A$ in (8), respectively.

More particularly, we will derive a new stepsize, say $\tilde{\alpha}_k(\Psi(A))$, which together with the stepsize $\alpha_k(\Psi(A))$ in (8) can give the minimizer of a two-dimensional convex quadratic function in no more than 5 iterations. To the best of our knowledge, this is the first nonmonotone gradient method with finite termination property. We will see that $\tilde{\alpha}_k(I) \le \alpha_k^{SD}$ and $\tilde{\alpha}_k(A) \le \alpha_k^{MG}$. Hence, this finite termination property is essentially obtained by inserting monotone steps into the generally nonmonotone class (8). In fact, we show that this finite termination property can be maintained even when the algorithm uses different function Ψ's during its iteration. Based on this observation, to achieve good numerical performance, we propose an adaptive nonmonotone gradient method (ANGM), which adaptively takes some nonmonotone steps involving the long and short BB stepsizes (5), and some monotone steps using $\tilde{\alpha}_k(A)$. Moreover, we propose two variants of ANGM, called ANGR1 and ANGR2, using certain retard stepsizes. Our numerical experiments show that the proposed methods significantly accelerate the BB method and perform much better on minimizing quadratic function (4) than the most successful gradient descent methods developed in the recent literature, such as the DY [10], ABBmin2 [18] and SDC methods [11]. In addition, our proposed methods are competitive with the CG method [16] for solving some quadratic optimization whose Hessian has a large condition number and the required accuracy is low.

The paper is organized as follows. In Sect. 2, we derive the new stepsize $\tilde{\alpha}_k(\Psi(A))$ by requiring finite termination on minimizing the two-dimensional strongly convex quadratic function. In Sect. 3, we develop the ANGM, ANGR1 and ANGR2 methods for quadratic optimization. Our numerical experiments are presented in Sect. 4. We finally draw some conclusion remarks in Sect. 5.

2 Derivation of new stepsize

In this section, we derive a new monotone stepsize by applying the class of nonmonotone gradient methods (8) to minimize the quadratic function (4). This new stepsize is motivated by requiring finite termination for minimizing the two-dimensional strongly convex quadratic function. Such an idea was originally proposed by Yuan [33] to accelerate the SD method. However, new techniques need to be developed for accelerating the class (8) since the key orthogonal property of successive two gradients generated by the SD method no longer holds for the class (8).

Notice that the class (8) is invariant under translations and rotations when minimizing quadratics. Hence, without loss of generality, we may simply assume that the matrix A is diagonal, i.e.

$$A = \mathrm{diag}\{\lambda_1, \ldots, \lambda_n\}, \tag{9}$$

where $0 < \lambda_1 \leq \ldots \leq \lambda_n$.

2.1 Motivation

Let us investigate the behavior of the class (8) with $\Psi(A) = I$ (i.e. the BB1 method). Particularly, we applied it to the non-random quadratic minimization problem in [11], which has the form (4) with a diagonal matrix A given by

$$A_{jj} = 10^{\frac{ncond}{n-1}(n-j)}, \quad j = 1, \ldots, n, \tag{10}$$

and b being the null vector. Here, $ncond = \log_{10} \kappa$ and $\kappa > 0$ is the condition number of A. We set $n = 10$, $\kappa = 10^3$ and used $(10, 10, \ldots, 10)^T$ as the initial point. The iteration was stopped once the gradient norm is reduced by a factor of 10^{-6}. Denote the i-th component of g_k by $g_k^{(i)}$ and the indices of the components of g_k with the first two largest magnitudes by i_1 and i_2, respectively. Then, the percentage of the magnitudes of the first two largest components to that of the whole gradient can be computed by

$$\Upsilon(g_k) = \frac{|g_k^{(i_1)}| + |g_k^{(i_2)}|}{\sum_{i=1}^{n} |g_k^{(i)}|}.$$

This $\Upsilon(g_k)$ is plotted in Fig. 1 (left), where we can see that $\Upsilon(g_k) \geq 0.8$ holds for more than half of the iterations (145 out of 224 total iterations). Hence, roughly speaking, the searches of the BB1 method are often dominated in some

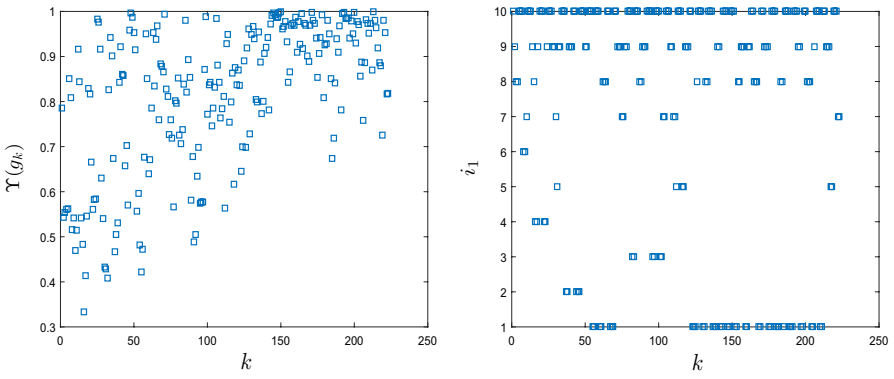

Fig. 1 Problem (10) with $n = 10$: history of $\Upsilon(g_k)$ (left) and the index i_1 (right) generated by the BB1 method

two-dimensional subspaces. The history of index i_1 against the iteration number is also plotted in Fig. 1 (right), where we can see that $|g_k^{(i_1)}|$ corresponds more frequently to the largest eigenvalues λ_{10} or λ_9. Since

$$g_{k+1}^{(j)} = (1 - \alpha_k \lambda_j) g_k^{(j)}, \; j = 1, \ldots, n. \tag{11}$$

and $1/\lambda_n \leq \alpha_k \leq 1/\lambda_1$, the history of i_1 in Fig. 1 (right) in fact indicates that, most stepsizes generated by the BB1 method are often much larger than $1/\lambda_{10}$ or $1/\lambda_9$. As a result, the BB1 method may need many iterations to reduce the corresponding components of the gradients $g_k^{(9)}$ or $g_k^{(10)}$.

In [23], we showed that a family of gradient methods including SD and MG will asymptotically reduce their searches in a two-dimensional subspace and could be accelerated by exploiting certain orthogonal properties in this two-dimensional subspace. In a similar spirit, we could also accelerate the convergence of the class of gradient methods (8) in a lower dimensional subspace if certain orthogonal properties hold.

Suppose that, for a given $k > 0$, there exists a q_k satisfying

$$(I - \alpha_{k-1} A) q_k = g_{k-1}. \tag{12}$$

Since this q_k is also invariant under translations and rotations, for later analysis we may still assume A in (12) is diagonal as in (9). The following lemma shows a generalized orthogonal property for q_k and g_{k+1}, which is a key property for deriving our new stepsize in the next subsection.

Lemma 1 (Orthogonal property) *Suppose that the sequence* $\{g_k\}$ *is obtained by applying the gradient method* (2) *with the stepsize* (8) *to minimize the quadratic function* (4) *and* q_k *satisfies* (12). *Then*,

$$q_k^T \Psi(A) g_{k+1} = 0. \tag{13}$$

Proof By (2), (8) and (12) we get

$$\begin{aligned} q_k^T \Psi(A) g_{k+1} &= q_k^T \Psi(A)(I - \alpha_k A)(I - \alpha_{k-1} A) g_{k-1} \\ &= g_{k-1}^T \Psi(A)(I - \alpha_k A) g_{k-1} \\ &= g_{k-1}^T \Psi(A) g_{k-1} - \alpha_k g_{k-1}^T \Psi(A) A g_{k-1} = 0. \end{aligned}$$

This completes the proof. □

2.2 A new stepsize

In this subsection, we derive a new stepsize based on the iterations of the class of gradient methods (8). We show that combining the new stepsize with the class (8), we can achieve finite termination for minimizing two-dimensional strongly convex quadratic functions.

By Lemma 1, we have that $g_k^T \Psi(A) q_{k-1} = 0$ for $k > 0$. Now, suppose both $\Psi^r(A) q_{k-1}$ and $\Psi^{1-r}(A) g_k$ are nonzero vectors, where $r \in \mathbb{R}$. Let us consider to minimize the function f in a two-dimensional subspace spanned by $\frac{\Psi^r(A) q_{k-1}}{\|\Psi^r(A) q_{k-1}\|}$ and $\frac{\Psi^{1-r}(A) g_k}{\|\Psi^{1-r}(A) g_k\|}$, and let

$$\begin{aligned} \varphi(t,l) &:= f\left(x_k + t \frac{\Psi^r(A) q_{k-1}}{\|\Psi^r(A) q_{k-1}\|} + l \frac{\Psi^{1-r}(A) g_k}{\|\Psi^{1-r}(A) g_k\|} \right) \\ &= f(x_k) + \vartheta_k^T \begin{pmatrix} t \\ l \end{pmatrix} + \frac{1}{2} \begin{pmatrix} t \\ l \end{pmatrix}^T H_k \begin{pmatrix} t \\ l \end{pmatrix}, \end{aligned} \tag{14}$$

where

$$\vartheta_k = B_k g_k = \begin{pmatrix} \frac{g_k^T \Psi^r(A) q_{k-1}}{\|\Psi^r(A) q_{k-1}\|} \\ \frac{g_k^T \Psi^{1-r}(A) g_k}{\|\Psi^{1-r}(A) g_k\|} \end{pmatrix} \text{ with } B_k = \left(\frac{\Psi^r(A) q_{k-1}}{\|\Psi^r(A) q_{k-1}\|}, \frac{\Psi^{1-r}(A) g_k}{\|\Psi^{1-r}(A) g_k\|} \right)^T \tag{15}$$

and

$$H_k = B_k A B_k^T = \begin{pmatrix} \frac{q_{k-1}^T \Psi^{2r}(A) A q_{k-1}}{\|\Psi^r(A) q_{k-1}\|^2} & \frac{q_{k-1}^T \Psi(A) A g_k}{\|\Psi^r(A) q_{k-1}\| \|\Psi^{1-r}(A) g_k\|} \\ \frac{q_{k-1}^T \Psi(A) A g_k}{\|\Psi^r(A) q_{k-1}\| \|\Psi^{1-r}(A) g_k\|} & \frac{g_k^T \Psi^{2(1-r)}(A) A g_k}{\|\Psi^{1-r}(A) g_k\|^2} \end{pmatrix}. \tag{16}$$

Denote the components of H_k by $H_k^{(ij)}$, $i,j = 1,2$ and notice that $B_k B_k^T = I$ by $g_k^T \Psi(A) q_{k-1} = 0$. Then, we have the following finite termination theorem.

Theorem 1 (Finite termination) *Suppose that the gradient method* (2) *is applied to minimize the two-dimensional quadratic function* (4) *with* α_k *given by* (8) *for all* $k \neq k_0$ *and uses the stepsize*

$$\tilde{\alpha}_{k_0} = \frac{2}{(H_{k_0}^{(11)} + H_{k_0}^{(22)}) + \sqrt{(H_{k_0}^{(11)} - H_{k_0}^{(22)})^2 + 4(H_{k_0}^{(12)})^2}} \tag{17}$$

at the k_0*-th iteration where* $k_0 \geq 2$. *Then, the method will find the minimizer in at most* $k_0 + 3$ *iterations*.

Proof Let us suppose x_k is not the minimizer for all $k = 1, \ldots, k_0 + 2$. We then show $k_0 + 3$ must be the minimizer, i.e. $g_{k_0+3} = 0$. For notation convenience, in the following proof of this theorem, let us simply use k to denote k_0.

First, we show that using the stepsize (17) at the k-th iteration implies

$$g_{k+1} \quad \text{is parallel to} \quad -B_k^T H_k^{-1} \vartheta_k + \tilde{\alpha}_k g_k, \tag{18}$$

where ϑ_k, B_k and H_k are given by (15) and (16). In fact, $\tilde{\alpha}_k$ given by (17) satisfies the following quadratic equation:

$$\tilde{\alpha}_k^2 \Delta - \tilde{\alpha}_k (H_k^{(11)} + H_k^{(22)}) + 1 = 0, \tag{19}$$

where $\Delta = \det(H_k) = \det(A) > 0$. Let

$$\Theta = (H_k^{(12)} \vartheta_k^{(1)} + H_k^{(22)} \vartheta_k^{(2)}) \vartheta_k^{(1)} - (H_k^{(11)} \vartheta_k^{(1)} + H_k^{(12)} \vartheta_k^{(2)}) \vartheta_k^{(2)},$$

where $\vartheta_k^{(i)}$ are components of ϑ_k, $i = 1, 2$. Then, multiplying Θ to (19), we have

$$\tilde{\alpha}_k^2 \Delta \Theta - \tilde{\alpha}_k (H_k^{(11)} + H_k^{(22)}) \Theta + \Theta = 0, \tag{20}$$

which is exactly

$$\begin{aligned} &(H_k^{(22)} \vartheta_k^{(1)} - H_k^{(12)} \vartheta_k^{(2)} - \tilde{\alpha}_k \Delta \vartheta_k^{(1)})[\vartheta_k^{(2)} - \tilde{\alpha}_k (H_k^{(12)} \vartheta_k^{(1)} + H_k^{(22)} \vartheta_k^{(2)})] \\ &\quad = (H_k^{(11)} \vartheta_k^{(2)} - H_k^{(12)} \vartheta_k^{(1)} - \tilde{\alpha}_k \Delta \vartheta_k^{(2)})[\vartheta_k^{(1)} - \tilde{\alpha}_k (H_k^{(11)} \vartheta_k^{(1)} + H_k^{(12)} \vartheta_k^{(2)})]. \end{aligned} \tag{21}$$

The above identity (21) implies the vector

$$\begin{pmatrix} \vartheta_k^{(1)} - \tilde{\alpha}_k (H_k^{(11)} \vartheta_k^{(1)} + H_k^{(12)} \vartheta_k^{(2)}) \\ \vartheta_k^{(2)} - \tilde{\alpha}_k (H_k^{(12)} \vartheta_k^{(1)} + H_k^{(22)} \vartheta_k^{(2)}) \end{pmatrix}$$

is parallel to

$$\begin{pmatrix} H_k^{(22)} \vartheta_k^{(1)} - H_k^{(12)} \vartheta_k^{(2)} - \tilde{\alpha}_k \Delta \vartheta_k^{(1)} \\ H_k^{(11)} \vartheta_k^{(2)} - H_k^{(12)} \vartheta_k^{(1)} - \tilde{\alpha}_k \Delta \vartheta_k^{(2)} \end{pmatrix},$$

which written in a matrix format just means

$$\vartheta_k + H_k(-\tilde{\alpha}_k \vartheta_k) \quad \text{is parallel to} \quad H_k^{-1} \vartheta_k - \tilde{\alpha}_k \vartheta_k. \tag{22}$$

Since $n = 2$, we have $B_k B_k^T = B_k^T B_k = I$. Then, it follows from $g_k = B_k^T \vartheta_k$, $\vartheta_k = B_k g_k$, $H_k = B_k A B_k^T$ and $g_{k+1} = g_k - \tilde{\alpha}_k A g_k$ that $g_{k+1} = B_k^T \vartheta_k + B_k^T H_k(-\tilde{\alpha}_k \vartheta_k)$. So, we have from (22) that (18) holds. Therefore, (17) implies (18) holds.

Now, it follows from (15) and $H_k^{-1} = B_k A^{-1} B_k^T$ that

$$-B_k^T H_k^{-1} \vartheta_k + \tilde{\alpha}_k g_k = -A^{-1} g_k + \tilde{\alpha}_k g_k = -A^{-1}(g_k - \tilde{\alpha}_k A g_k) = -A^{-1} g_{k+1}.$$

Hence, (18) implies g_{k+1} is parallel to $A^{-1} g_{k+1}$. So, if $\tilde{\alpha}_k$ given by (17) is used at the k-th iteration, then g_{k+1} is parallel to $A^{-1} g_{k+1}$. Since x_{k+1} is not the minimizer, we have $g_{k+1} \neq 0$. Then, g_{k+1} is an eigenvector of A, i.e. $A g_{k+1} = \lambda g_{k+1}$ for some $\lambda > 0$. Since x_{k+2} is not the minimizer, we have $g_{k+2} \neq 0$ and the algorithm will not stop at the $k+2$-th iteration. By (8), we have $\alpha_{k+2} = \frac{g_{k+1}^T \Psi(A) g_{k+1}}{g_{k+1}^T \Psi(A) A g_{k+1}} = 1/\lambda$. Thus, we get $g_{k+3} = (1 - \alpha_{k+2}\lambda) g_{k+2} = 0$, which implies x_{k+3} must be the minimizer. We complete the proof. □

Notice that by setting $k_0 = 2$ in the above Theorem 1, the new gradient method in Theorem 1 will find the exact minimizer in at most 5 iterations when minimizing a two-dimensional strongly convex quadratic function. In fact, since $\Delta = \lambda_1 \lambda_2$ and

$H_k^{(11)} + H_k^{(22)} = \lambda_1 + \lambda_2$, the equation (19) has two positive roots $1/\lambda_1$ and $1/\lambda_2$, and $\tilde{\alpha}_{k_0} = 1/\lambda_2$. This observation allows us to use the stepsize $\tilde{\alpha}_{k_0}$ with some retards as stated in the following corollary.

Corollary 1 *Suppose that a gradient method is applied to the two-dimensional quadratic function* (4) *with* $\alpha_{k_0+m} = \tilde{\alpha}_{k_0}$ *for* $k_0 \geq 2$ *and some positive integer* m, *and* α_k *given by* (8) *for all* $k \neq k_0 + m$. *Then, the method stops in at most* $k_0 + m + 3$ *iterations*.

By setting $\Psi(A) = I$, $\Psi(A) = A$ and $r = 1/2$ in (16), and setting $k_0 = k$ in (17), we can derive the following two stepsizes:

$$\tilde{\alpha}_k^{BB1} = \frac{2}{\frac{q_{k-1}^T A q_{k-1}}{\|q_{k-1}\|^2} + \frac{1}{\alpha_k^{SD}} + \sqrt{\left(\frac{q_{k-1}^T A q_{k-1}}{\|q_{k-1}\|^2} - \frac{1}{\alpha_k^{SD}}\right)^2 + \frac{4(q_{k-1}^T A g_k)^2}{\|q_{k-1}\|^2 \|g_k\|^2}}} \tag{23}$$

and

$$\tilde{\alpha}_k^{BB2} = \frac{2}{\frac{1}{\hat{\alpha}_{k-1}} + \frac{1}{\alpha_k^{MG}} + \sqrt{\left(\frac{1}{\hat{\alpha}_{k-1}} - \frac{1}{\alpha_k^{MG}}\right)^2 + \Gamma_k}}, \tag{24}$$

respectively, where

$$\hat{\alpha}_k = \frac{q_k^T A q_k}{q_k^T A^2 q_k} \quad \text{and} \quad \Gamma_k = \frac{4(q_{k-1}^T A^2 g_k)^2}{q_{k-1}^T A q_{k-1} \cdot g_k^T A g_k}. \tag{25}$$

From (23) and (24), we have

$$\tilde{\alpha}_k^{BB1} \leq \min\left\{\alpha_k^{SD}, \frac{\|q_{k-1}\|^2}{q_{k-1}^T A q_{k-1}}\right\} \quad \text{and} \quad \tilde{\alpha}_k^{BB2} \leq \min\{\alpha_k^{MG}, \hat{\alpha}_{k-1}\}. \tag{26}$$

Hence, both $\tilde{\alpha}_k^{BB1}$ and $\tilde{\alpha}_k^{BB2}$ are short monotone stepsizes for reducing the value and gradient norm of the objective function, respectively. It follows from Theorem 1 that by inserting the monotone steps with $\tilde{\alpha}_k^{BB1}$ and $\tilde{\alpha}_k^{BB2}$ into the BB1 and BB2 methods, respectively, the class of gradient methods (8) will have finite termination for minimizing two-dimensional strongly convex quadratic functions.

To numerically verify this finite termination property, we applied the class (8) with $\Psi(A) = I$ (i.e. the BB1 method) and $\tilde{\alpha}_2^{BB1}$ given by (23) to minimize the two-dimensional quadratic function (4) with

$$A = \text{diag}\{1, \lambda\} \quad \text{and} \quad b = 0. \tag{27}$$

We run the algorithm for five iterations using ten random starting points. The average values of $\|g_5\|$ and $f(x_5)$ are presented in Table 1. We also run the BB1 method for a comparison purpose. We can observe that for different values of λ, the values of $\|g_5\|$ and $f(x_5)$ obtained by the BB1 method with $\tilde{\alpha}_2^{BB1}$ given by (23) are numerically

Table 1 Average results on problem (27) with different condition numbers

λ	BB1		BB1 with $\tilde{\alpha}_2^{BB1}$ given by (23)	
	$\|g_5\|$	$f(x_5)$	$\|g_5\|$	$f(x_5)$
10	6.6873e+00	4.8701e+00	1.1457e−16	5.8735e−31
100	8.6772e+01	1.9969e+02	2.1916e−16	2.8047e−30
1000	2.0925e+02	9.7075e+01	3.4053e−19	2.3730e−29
10000	1.7943e+02	9.6935e+00	5.1688e−19	6.7870e−28

very close to zero. However, even for the case $\lambda = 10$, $\|g_5\|$ and $f(x_5)$ obtained by the pure BB1 method are far away from zero. These numerical results coincide with our analysis and show that the nonmonotone class (8) can be significantly accelerated by incorporating proper monotone steps.

3 New method

In this section, based on the above analysis, we propose an adaptive nonmonotone gradient method (ANGM) and its two variants, ANGR1 and ANGR2.

As mentioned in Sect. 2.1, the stepsizes α_k^{BB1} generated by the BB1 method may be far away from the reciprocals of the largest eigenvalues of the Hessian A. In other words, the stepsize α_k^{BB1} may be too large to effectively decrease the components of g_k corresponding to the first several largest eigenvalues, which, by (11), can be greatly reduced when small stepsizes are employed. In addition, it has been observed by many works in the recent literature that gradient methods using long and short stepsizes adaptively generally perform much better than those using only one type of stepsizes, for example see [9, 11, 12, 21–23]. So, we would like to develop gradient methods that combines the two nonmonotone BB stepsizes with the short monotone stepsize given by (17).

We first extend the orthogonal property developed in Lemma 1 and the finite termination result given in Theorem 1.

Lemma 2 (Generalized orthogonal property) *Suppose that the gradient method* (2) *with stepsizes in the form of* (8) *is applied to minimize the quadratic function* (4). *In particular, at the* $k-1$*-th and* k*-th iteration, two stepsizes* $\alpha_{k-1}(\Psi(A))$ *and* $\alpha_k(\Psi_1(A))$ *are used, respectively, where* Ψ *and* Ψ_1 *may be two different analytic functions used in* (8). *If* $q_k \in \mathbb{R}^n$ *satisfies*

$$(I - \alpha_{k-1}(\Psi(A))A)q_k = g_{k-1}, \tag{28}$$

then we have

$$q_k^T \Psi_1(A) g_{k+1} = 0. \tag{29}$$

Proof Notice that by (2), we have

$$g_k = g_{k-1} - \alpha_{k-1}(\Psi(A))Ag_{k-1} \quad \text{and} \quad g_{k+1} = g_k - \alpha_k(\Psi_1(A))Ag_k.$$

Then, the proof is essential the same as those in the proof of Lemma 1. □

Based on Lemma 2 and using the same arguments as those in the proof of Theorem 1, we can obtain the following finite termination result even different function Ψ's are used in (8) to obtain the stepsizes.

Theorem 2 (Generalized finite termination) *Suppose that the gradient method* (2) *is applied to minimize the two-dimensional quadratic function* (4) *with* α_k *given by* (8) *for all* $k \neq k_0$ *and* $k \neq k_0 - 1$, *and uses the stepsizes* $\alpha_{k-1}(\Psi_1(A))$ *and* $\alpha_k(\Psi_1(A))$ *at the* $k-1$*-th and* k*-th iteration, respectively, where* $k_0 \geq 2$. *Then, the method will find the minimizer in at most* $k_0 + 3$ *iterations*.

Theorem 2 allows us to incorporate the nonmonotone BB stepsizes α_k^{BB1} and α_k^{BB2}, and the short monotone stepsize $\tilde{\alpha}_k^{BB2}$ in one gradient method. Alternate or adaptive scheme has been employed for choosing long and short stepsizes in BB-type methods [5, 35]. Recent studies show that adaptive strategies are more preferred than the alternate scheme [6, 35]. Hence, we would like to develop adaptive strategies to choose proper stepsizes for our new gradient methods. In particular, our adaptive nonmonotone gradient method (ANGM) takes the long BB stepsize α_k^{BB1} when $\alpha_k^{BB2}/\alpha_k^{BB1} \geq \tau_1$ for some $\tau_1 \in (0, 1)$. Otherwise, a short stepsize α_k^{BB2} or $\tilde{\alpha}_k^{BB2}$ will be taken depending on the ratio $\|g_{k-1}\|/\|g_k\|$. Notice that α_k^{BB2} minimizes the gradient in the sense that

$$\alpha_k^{BB2} = \alpha_{k-1}^{MG} = \arg\min_{\alpha \in \mathbb{R}} \|g_{k-1} - \alpha A g_{k-1}\|.$$

When $\|g_{k-1}\|/\|g_k\| > \tau_2$ for some $\tau_2 > 1$, i.e. the gradient norm decreases, the previous stepsize α_{k-1} is often a reasonable approximation of $\alpha_k^{BB2} = \alpha_{k-1}^{MG}$. By our numerical experiments, when the BB method is applied the searches are often dominated in some two-dimensional subspaces. Theorem 2 indicates that the gradient method would have finite termination for minimizing two-dimensional convex quadratic functions when the new stepsize $\tilde{\alpha}_k^{BB2}$ is applied after some BB2 steps. Hence, our ANGM would employ the new monotone stepsize $\tilde{\alpha}_k^{BB2}$ when $\|g_{k-1}\| \geq \tau_2\|g_k\|$; otherwise, certain BB2 steps should be taken. In practice, we find that when $\|g_{k-1}\| < \tau_2\|g_k\|$, ANGM often has good performance by taking the stepsize $\min\{\alpha_k^{BB2}, \alpha_{k-1}^{BB2}\}$. To summarize, our ANGM applies the following adaptive strategies for choosing stepsizes:

$$\alpha_k = \begin{cases} \min\{\alpha_k^{BB2}, \alpha_{k-1}^{BB2}\}, & \text{if} \quad \alpha_k^{BB2} < \tau_1 \alpha_k^{BB1} \text{ and } \|g_{k-1}\| < \tau_2\|g_k\|; \\ \tilde{\alpha}_k^{BB2}, & \text{if} \quad \alpha_k^{BB2} < \tau_1 \alpha_k^{BB1} \text{ and } \quad \|g_{k-1}\| \geq \tau_2\|g_k\|; \\ \alpha_k^{BB1}, & \text{otherwise.} \end{cases} \tag{30}$$

Notice that the calculation of $\tilde{\alpha}_k^{BB2}$ needs to compute α_k^{MG} which is not easy to obtain when the objective function is non-quadratic. Instead, the calculation of $\tilde{\alpha}_{k-1}^{BB2}$ will just require α_k^{BB2}, which is readily available even for general objective functions.

Moreover, it is found in recent research that gradient methods using retard stepsizes can often lead better performances [19]. Hence, in the first variant of ANGM, we simply replace $\tilde{\alpha}_k^{BB2}$ in (30) by $\tilde{\alpha}_{k-1}^{BB2}$, i.e. the stepsizes are chosen as

$$\alpha_k = \begin{cases} \min\{\alpha_k^{BB2}, \alpha_{k-1}^{BB2}\}, & \text{if} \quad \alpha_k^{BB2} < \tau_1 \alpha_k^{BB1} \text{ and } \|g_{k-1}\| < \tau_2 \|g_k\|; \\ \tilde{\alpha}_{k-1}^{BB2}, & \text{if} \quad \alpha_k^{BB2} < \tau_1 \alpha_k^{BB1} \text{ and } \|g_{k-1}\| \geq \tau_2 \|g_k\|; \\ \alpha_k^{BB1}, & \text{otherwise.} \end{cases} \tag{31}$$

We call the gradient method using stepsize (31) ANGR1. On the other hand, since the calculation of $\tilde{\alpha}_{k-1}^{BB2}$ also needs $\hat{\alpha}_{k-2}$ and Γ_{k-1} and by (26),

$$\tilde{\alpha}_{k-1}^{BB2} \leq \min\{\alpha_k^{BB2}, \hat{\alpha}_{k-2}\}, \tag{32}$$

to simplify ANGR1, we may further replace $\tilde{\alpha}_{k-1}^{BB2}$ in (31) by its upper bound in (32). As a result, we have the second variant of ANGM, which chooses stepsizes as

$$\alpha_k = \begin{cases} \min\{\alpha_k^{BB2}, \alpha_{k-1}^{BB2}\}, & \text{if} \quad \alpha_k^{BB2} < \tau_1 \alpha_k^{BB1} \text{ and } \|g_{k-1}\| < \tau_2 \|g_k\|; \\ \min\{\alpha_k^{BB2}, \hat{\alpha}_{k-2}\}, & \text{if} \quad \alpha_k^{BB2} < \tau_1 \alpha_k^{BB1} \text{ and } \|g_{k-1}\| \geq \tau_2 \|g_k\|; \\ \alpha_k^{BB1}, & \text{otherwise.} \end{cases} \tag{33}$$

We call the gradient method using stepsize (33) ANGR2.

Before presenting our algorithms, we briefly describe how to compute the new stepsize $\tilde{\alpha}_k^{BB2}$ without additional matrix-vector products. Since $\tilde{\alpha}_k^{BB2}$ involves $\hat{\alpha}_{k-1}$, Γ_k, and α_k^{MG}, we investigate their computation cost one by one. By (12) we have that

$$Aq_k = \frac{1}{\alpha_{k-1}}(q_k - g_{k-1}),$$

which together with (25) gives

$$\hat{\alpha}_{k-1} = \frac{q_{k-1}^T A q_{k-1}}{q_{k-1}^T A^2 q_{k-1}} = \frac{\alpha_{k-2} q_{k-1}^T (q_{k-1} - g_{k-2})}{(q_{k-1} - g_{k-2})^T (q_{k-1} - g_{k-2})} \tag{34}$$

and

$$\Gamma_k = \frac{4(q_{k-1}^T A^2 g_k)^2}{q_{k-1}^T A q_{k-1} \cdot g_k^T A g_k} = \frac{4((q_{k-1} - g_{k-2})^T A g_k)^2}{\alpha_{k-2} q_{k-1}^T (q_{k-1} - g_{k-2}) \cdot g_k^T A g_k}. \tag{35}$$

Recall that Ag_k is necessary for updating g_k. If Ag_k is kept in memory, by (34), (35) and the definition of α_k^{MG}, we know that $\tilde{\alpha}_k^{BB2}$ can be computed without additional matrix-vector products when q_{k-1} is available.

However, computing q_k exactly from (12) may be as difficult as minimizing the quadratic function. Notice that the q_k satisfying (12) also satisfies the secant equation

$$q_k^T g_k = \|g_{k-1}\|^2. \tag{36}$$

Hence, we may find an approximation of q_k by requiring the above secant condition holds. One efficient way to find such a q_k satisfying the secant equation (36) is to simply treat the Hessian A as the diagonal matrix (9) and derive q_k from (12), that is when $g_k^{(i)} \neq 0$,

Algorithm 1: ANGM method for quadratic optimization

Initialize $x_0 \in \mathbb{R}^n$ and $\alpha_0 > 0$. Choose $\tau_1 \in (0,1)$ and $\tau_2 \geq 1$. $x = x_0$, $g_0 = Ax - b$, $w = Ag_0$, $u = g_k^T w$, $v = w^T w$, $\rho_c = g_0^T g_0$. Set $k := 0$.
while $\sqrt{\rho_c} > 0$ **do**
  $x = x - \alpha_k g_k$
  $\alpha_{k+1}^{BB1} = \rho_c / u, \quad \alpha_{k+1}^{BB2} = u/v$
  $g_{k+1} = g_k - \alpha_k w$
  $\rho_n = g_{k+1}^T g_{k+1}$
  $w = Ag_{k+1}$, $u = g_{k+1}^T w$, $v = w^T w$
  if $k > 1$ *and* $\alpha_{k+1}^{BB1} < \tau_1 \alpha_{k+1}^{BB2}$ **then**
    if $\sqrt{\rho_c} < \tau_2 \sqrt{\rho_n}$ **then**
      $\alpha_{k+1} = \min\{\alpha_{k+1}^{BB2}, \alpha_k^{BB2}\}$
    else
      $$q^{(i)} = \begin{cases} \frac{(g_{k-1}^{(i)})^2}{g_k^{(i)}}, & \text{if } g_k^{(i)} \neq 0; \\ 0, & \text{if } g_k^{(i)} = 0. \end{cases}$$
      $\beta = (q - g_{k-1})^T q$, $\gamma = (q - g_{k-1})^T (q - g_{k-1})$
      $\hat{\alpha} = \alpha_{k-1} \beta / \gamma$, $\Gamma = 4((q - g_{k-1})^T w)^2 / (\alpha_{k-1} \beta u)$
      $$\alpha_{k+1} = 2 \left(1/\hat{\alpha} + v/u + \sqrt{(1/\hat{\alpha} - v/u)^2 + \Gamma} \right)^{-1}$$
    end
  else
    $\alpha_{k+1} = \alpha_{k+1}^{BB1}$
  end
  $\rho_c = \rho_n$
  $k = k + 1$
end

$$q_k^{(i)} = \frac{g_{k-1}^{(i)}}{1 - \alpha_{k-1}\lambda_i} = \frac{(g_{k-1}^{(i)})^2}{g_k^{(i)}}, \quad i = 1, \ldots, n. \tag{37}$$

We can just let $q_k^{(i)} = 0$, if $g_k^{(i)} = 0$. To summarize, the approximated q_k can be computed by

$$q_k^{(i)} = \begin{cases} \frac{(g_{k-1}^{(i)})^2}{g_k^{(i)}}, & \text{if} \quad g_k^{(i)} \neq 0; \\ 0, & \text{if} \quad g_k^{(i)} = 0. \end{cases} \tag{38}$$

As we will see in Sect. 4, this simple way of calculating q_k leads to a very efficient algorithm.

Now we formally present our ANGM method in Algorithm 1 and ANGR1 method in Algorithm 2, respectively. The ANGR2 method can be obtained by replacing lines 14 and 15 in Algorithm 2 as

$$\hat{\alpha} = \alpha_{k-2}\beta/\gamma, \quad \alpha_{k+1} = \min\{\alpha_{k+1}^{BB2}, \hat{\alpha}\}.$$

Clearly, a step of ANGM requires one matrix-vector product and three inner products when $\sqrt{\rho_c} < \tau_2\sqrt{\rho_n}$; otherwise, three more inner products are required. Five length-n vectors and the vector q are required. The computational cost and memory cost for a step of ANGR1 is the same as ANGM since it requires g_{k-2} while g_{k+1} is not necessary. Compared to ANGR1, a step of ANGR2 requires one fewer inner product when $\sqrt{\rho_c} < \tau_2\sqrt{\rho_n}$ due to the absence of Γ.

Algorithm 2: ANGR1 method for quadratic optimization

1 Initialize $x_0 \in \mathbb{R}^n$ and $\alpha_0 > 0$. Choose $\tau_1 \in (0,1)$ and $\tau_2 \geq 1$. $x = x_0$, $g_0 = Ax - b$, $\rho_c = g_0^T g_0$. Set $k := 0$.
2 **while** $\sqrt{\rho_c} > 0$ **do**
3 $x = x - \alpha_k g_k$
4 $w = Ag_k$, $u = g_k^T w$, $v = w^T w$
5 $\alpha_{k+1}^{BB1} = \rho_c/u$, $\alpha_{k+1}^{BB2} = u/v$
6 $g_{k+1} = g_k - \alpha_k w$
7 $\rho_n = g_{k+1}^T g_{k+1}$
8 **if** $k > 1$ *and* $\alpha_{k+1}^{BB1} < \tau_1 \alpha_{k+1}^{BB2}$ **then**
9 **if** $\sqrt{\rho_c} < \tau_2\sqrt{\rho_n}$ **then**
10 $\alpha_{k+1} = \min\{\alpha_{k+1}^{BB2}, \alpha_k^{BB2}\}$
11 **else**
12 $q^{(i)} = \begin{cases} \frac{(g_{k-2}^{(i)})^2}{g_{k-1}^{(i)}}, & \text{if } g_{k-1}^{(i)} \neq 0; \\ 0, & \text{if } g_{k-1}^{(i)} = 0. \end{cases}$
13 $\beta = (q - g_{k-2})^T q$, $\gamma = (q - g_{k-2})^T (q - g_{k-2})$
14 $\hat{\alpha} = \alpha_{k-2}\beta/\gamma$, $\Gamma = 4((q - g_{k-2})^T w)^2/(\alpha_{k-2}\beta u)$
15 $\alpha_{k+1} = 2\left(1/\hat{\alpha}_{k-1} + 1/\alpha_{k+1}^{BB2} + \sqrt{\left(1/\hat{\alpha} - 1/\alpha_{k+1}^{BB2}\right)^2 + \Gamma}\right)^{-1}$
16 **end**
17 **else**
18 $\alpha_{k+1} = \alpha_{k+1}^{BB1}$
19 **end**
20 $\rho_c = \rho_n$
21 $k = k + 1$
22 **end**

Remark 1 We notice that a BB step requires one matrix-vector product, two inner products, and three length-n vectors while a CG step requires one more length-n vector. Although the memory cost and computational cost of our methods are slightly higher than those for BB and CG, we will see in next section that the cost is deserved especially when the condition number of the Hessian is large.

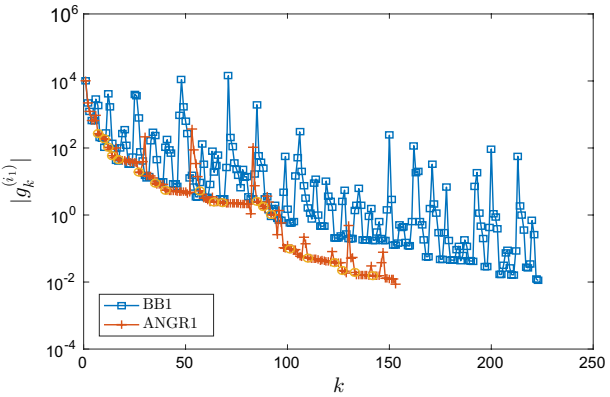

Fig. 2 Problem (10) with $n = 10$: history of $|g_k^{(i_1)}|$ generated by the BB1 and ANGR1 methods

Table 2 The correspondence of i_1 and λ_j by the BB1 and ANGR1 methods

n_j	1	2	3	4	5	6	7	8	9	10
BB1	40	5	6	4	6	2	10	21	42	87
ANGR1	51	4	22	18	16	4	1	8	12	17

In terms of global convergence for minimizing the quadratic function (4), by (26), we can easily show the R-linear global convergence of ANGM since it satisfies the property in [4]. Similarly, R-linear convergence of ANGR1 and ANGR2 can be also established. See the proof of Theorem 3 in [6] for example.

For a simple illustration of numerical behavior of ANGR1, we again applied ANGR1 with $\tau_1 = 0.85$ and $\tau_2 = 1.3$ to solve problem (10) with $n = 10$. Figure 2 shows the largest component $|g_k^{(i_1)}|$ of the gradient generated by the BB1 and ANGR1 methods against the iteration number, where circle means the ANGR1 method takes $\tilde{\alpha}_{k-1}^{BB2}$ at that iteration. It can be seen that, at those circle iterations, $|g_k^{(i_1)}|$ generated by the BB1 method often increases significantly with a much larger value. On the other hand, $|g_k^{(i_1)}|$ generated by the ANGR1 method is often reduced and kept small after a circle iteration (i.e. after $\tilde{\alpha}_{k-1}^{BB2}$ is applied). A detail correspondence of i_1 and λ_j is presented in Table 2, where n_j is the total number of $i_1's$ for which $i_1 = j$, $j = 1, \ldots, 10$. We can see from the last three columns in Table 2 that ANGR1 is more efficient than BB1 for decreasing those components of g_k corresponding to large eigenvalues. Hence, the undesired behavior of BB1 discussed in the motivation Sect. 2.1 is greatly eliminated by ANGR1.

4 Numerical results

In this section, we present numerical comparisons of ANGM, ANGR1 and ANGR2 with the BB1 [2], DY [10], ABBmin2 [18], SDC [11] and CG (Algorithm 11.3.3 in [20]) methods on quadratic optimization. All the methods were

implemented in MATLAB (v.9.0-R2016a) and run on a laptop with an Intel Core i7, 2.9 GHz processor and 8 GB of RAM running Windows 10 system.

We first solve some randomly generated quadratic problems in the form (see [33])

$$\min_{x\in\mathbb{R}^n} f(x) = (x - x^*)^T V(x - x^*), \tag{39}$$

where x^* is randomly generated with components between -10 and 10, $V = \text{diag}\{v_1, \ldots, v_n\}$ is a diagonal matrix with $v_1 = 1$ and $v_n = \kappa$, and v_j, $j = 2, \ldots, n-1$, are generated by the *rand* function between 1 and κ.

We have tested five sets of problems (39) with $n = 1000$ using different spectral distributions of the Hessian listed in Table 3. The algorithm was stopped once the number of iteration exceeds 20,000 or the gradient norm was reduced by a factor of ϵ, which was set to $10^{-6}, 10^{-9}$ and 10^{-12}, respectively. Five different condition numbers $\kappa = 10^2, 10^3, 10^4, 10^5, 10^6$ were tested. For each value of κ or ϵ, 10 instances of the problem were randomly generated and the average results obtained by the starting point $x_0 = (0, \ldots, 0)^T$ are presented.

For the ABBmin2 method, τ was set to 0.9 as suggested in [18]. The parameter pair (h, s) of the SDC method was set to (8, 6) which is more efficient than other choices for this test. For ANGM, ANGR1 and ANGR2, we chose τ_1 from $\{0.1, 0.2, \ldots, 0.9\}$ such that the methods achieve the best performance for $\tau_2 = \{1, 1.02, 1.1, 1.2\}$. In particular, the pair (τ_1, τ_2) was to (0.1, 1), (0.1, 1) and (0.3, 1) for ANGM, ANGR1 and ANGR2, respectively. The q_k was calculated by (38) for our methods.

The average number of iterations for the compared methods are presented in Table 4. We see that, our three methods are much better than BB1, DY and SDC especially for high accuracy or a large condition number. As compared with ABBmin2, similar picture emerges regard to the second to last problem sets though ABBmin2 wins for the first problem set. Moreover, for a small condition number, our three methods are competitive with CG when low accuracy is required. For the case $\epsilon = 10^{-6}$, our ANGR1 and ANGR2 methods are

Table 3 Distributions of v_j

Problem	Spectrum
1	$\{v_2, \ldots, v_{n-1}\} \subset (1, \kappa)$
2	$\{v_2, \ldots, v_{n/5}\} \subset (1, 100)$
	$\{v_{n/5+1}, \ldots, v_{n-1}\} \subset (\frac{\kappa}{2}, \kappa)$
3	$\{v_2, \ldots, v_{n/2}\} \subset (1, 100)$
	$\{v_{n/2+1}, \ldots, v_{n-1}\} \subset (\frac{\kappa}{2}, \kappa)$
4	$\{v_2, \ldots, v_{4n/5}\} \subset (1, 100)$
	$\{v_{4n/5+1}, \ldots, v_{n-1}\} \subset (\frac{\kappa}{2}, \kappa)$
5	$\{v_2, \ldots, v_{n/5}\} \subset (1, 100)$
	$\{v_{n/5+1}, \ldots, v_{4n/5}\} \subset (100, \frac{\kappa}{2})$
	$\{v_{4n/5+1}, \ldots, v_{n-1}\} \subset (\frac{\kappa}{2}, \kappa)$

Table 4 The average number of iterations for the ANGM, ANGR1, ANGR2, BB1, DY, SDC, CG and ABBmin2 methods on problems in Table 3

κ	ϵ	BB1	DY	SDC	CG	ABBmin2	ANGM	ANGR1	ANGR2
Problem set 1									
10^2	10^{-6}	70.4	69.8	64.7	53.6	62.8	64.3	65.5	67.5
	10^{-9}	111.3	112.0	103.6	86.9	99.8	102.2	104.1	101.1
	10^{-12}	149.0	147.2	144.2	118.2	135.9	143.1	142.4	141.8
10^3	10^{-6}	182.7	154.8	147.5	109.9	139.4	156.9	158.9	145.6
	10^{-9}	333.7	277.3	286.8	163.4	237.4	292.3	305.0	259.3
	10^{-12}	460.8	406.6	400.8	203.8	328.2	428.5	434.2	375.3
10^4	10^{-6}	329.1	295.1	295.9	160.3	276.5	303.5	303.7	298.9
	10^{-9}	912.6	741.4	779.0	203.5	430.2	584.6	619.8	525.5
	10^{-12}	1420.2	1168.2	1178.0	237.2	604.7	885.1	889.2	723.9
10^5	10^{-6}	214.7	192.8	174.6	162.2	195.9	197.7	203.8	182.8
	10^{-9}	2196.7	1961.1	2024.0	211.7	489.3	641.1	632.0	681.3
	10^{-12}	4039.0	3393.8	3625.8	243.5	651.2	949.4	916.3	864.8
10^6	10^{-6}	204.9	174.8	170.5	111.5	143.8	152.2	175.2	167.4
	10^{-9}	6390.7	5805.9	3286.2	220.5	501.6	711.5	765.1	696.8
	10^{-12}	11861.8	14022.9	6550.7	250.9	581.8	962.6	886.3	945.6
Problem set 2									
10^2	10^{-6}	67.6	62.6	64.6	44.7	57.1	65.9	64.3	59.4
	10^{-9}	111.0	103.2	101.2	65.5	87.5	97.4	97.9	98.6
	10^{-12}	151.8	142.1	135.6	82.5	125.1	136.0	136.7	139.0
10^3	10^{-6}	156.3	163.3	144.8	96.1	137.0	152.7	144.0	135.0
	10^{-9}	305.1	325.2	265.9	155.0	240.4	279.4	273.3	257.0
	10^{-12}	422.4	461.7	370.2	197.4	338.2	397.9	406.2	359.9
10^4	10^{-6}	256.1	239.7	226.4	113.9	235.6	238.6	192.3	173.8
	10^{-9}	625.8	573.7	524.9	223.5	506.7	473.7	442.5	386.6
	10^{-12}	915.7	871.5	788.5	302.9	719.8	739.5	664.4	563.4
10^5	10^{-6}	359.1	316.2	193.1	89.8	340.3	171.4	128.2	111.3
	10^{-9}	1419.7	1246.3	840.0	253.5	983.4	691.5	469.6	432.8
	10^{-12}	2266.9	2244.5	1395.1	383.4	1636.2	1114.8	778.9	749.2
10^6	10^{-6}	319.4	317.8	94.0	47.3	269.2	106.2	70.4	62.7
	10^{-9}	2777.4	2352.3	946.0	255.7	2014.5	971.6	450.7	566.2
	10^{-12}	5110.1	4762.1	1770.1	415.1	3976.3	1744.5	907.8	1064.2
Problem set 3									
10^2	10^{-6}	69.1	62.5	67.6	50.8	61.1	62.5	64.6	64.0
	10^{-9}	113.4	101.4	104.0	79.9	95.1	100.6	101.3	99.2
	10^{-12}	149.7	141.8	141.4	105.7	134.0	138.8	138.0	138.7
10^3	10^{-6}	162.3	169.5	160.9	109.2	148.9	159.9	161.2	149.6
	10^{-9}	302.9	323.4	286.7	188.9	261.9	282.4	293.7	266.0
	10^{-12}	427.1	472.1	401.2	257.8	372.6	396.8	410.3	379.6

Table 4 (continued)

κ	ϵ	BB1	DY	SDC	CG	ABBmin2	ANGM	ANGR1	ANGR2
10^4	10^{-6}	284.4	266.0	252.6	128.2	264.8	248.5	240.6	193.7
	10^{-9}	650.1	619.8	526.1	259.3	540.6	507.5	460.5	396.9
	10^{-12}	961.9	937.0	796.9	367.6	823.5	767.0	704.2	584.5
10^5	10^{-6}	416.5	412.7	268.1	103.9	471.8	225.1	168.4	153.6
	10^{-9}	1350.5	1451.3	970.8	283.7	1178.7	767.8	508.2	475.3
	10^{-12}	2269.4	2300.2	1523.1	437.0	1824.8	1211.0	784.2	783.6
10^6	10^{-6}	491.7	339.9	125.7	61.1	460.6	159.3	91.2	87.0
	10^{-9}	3134.5	2695.6	1036.5	279.1	2607.6	927.5	527.3	624.0
	10^{-12}	5677.6	4968.0	2045.3	476.7	4294.6	1842.8	963.8	1041.8
Problem set 4									
10^2	10^{-6}	70.0	62.6	63.4	52.7	62.4	63.7	65.1	66.0
	10^{-9}	113.4	107.7	105.8	84.8	98.4	101.8	104.2	102.1
	10^{-12}	159.3	141.7	144.1	115.1	138.0	139.5	143.4	141.4
10^3	10^{-6}	177.0	190.9	161.4	116.4	157.8	171.1	165.1	166.9
	10^{-9}	319.7	344.2	292.9	195.3	271.8	303.9	294.6	277.1
	10^{-12}	450.2	484.3	425.9	266.6	377.9	418.3	416.5	394.1
10^4	10^{-6}	330.5	317.9	265.1	141.8	290.9	269.2	254.0	229.4
	10^{-9}	707.4	626.2	546.9	267.5	567.9	571.8	513.3	416.1
	10^{-12}	1068.8	931.4	853.3	380.1	842.2	828.1	739.0	605.6
10^5	10^{-6}	542.8	504.9	326.7	123.2	517.7	252.5	187.1	187.7
	10^{-9}	1585.8	1429.6	966.8	286.5	1168.7	800.7	529.5	512.8
	10^{-12}	2321.1	2372.0	1632.1	444.9	1849.5	1254.8	837.0	817.0
10^6	10^{-6}	649.2	490.3	173.9	76.3	694.5	191.7	110.0	105.3
	10^{-9}	3683.3	2610.5	1084.0	282.8	2806.0	1015.3	529.4	648.0
	10^{-12}	5992.1	4551.9	1973.8	484.5	4599.9	1873.4	916.3	1165.5
Problem set 5									
10^2	10^{-6}	69.8	65.6	63.7	44.9	55.8	61.9	61.9	60.4
	10^{-9}	106.4	102.5	101.1	65.6	93.7	100.7	99.1	97.0
	10^{-12}	147.2	144.3	137.8	82.1	127.1	139.4	136.4	131.1
10^3	10^{-6}	196.8	176.7	165.2	129.0	158.3	173.4	173.9	163.8
	10^{-9}	326.4	299.4	283.6	207.6	263.5	297.1	311.2	272.6
	10^{-12}	456.9	424.5	408.3	270.2	379.4	439.5	437.4	397.0
10^4	10^{-6}	494.4	388.8	388.5	327.8	397.8	403.3	394.2	388.1
	10^{-9}	1015.1	869.8	897.3	577.5	762.8	845.7	831.6	810.6
	10^{-12}	1517.6	1356.0	1391.1	752.2	1153.0	1208.4	1260.5	1184.8
10^5	10^{-6}	713.2	660.4	664.1	508.8	881.7	677.5	690.8	646.7
	10^{-9}	2786.6	2302.7	2459.6	1081.4	2149.0	2101.1	2171.8	1990.0
	10^{-12}	5020.2	4050.1	4051.9	1486.1	3324.6	3312.7	3272.4	3128.1

Table 4 (continued)

κ	ϵ	BB1	DY	SDC	CG	ABBmin2	ANGM	ANGR1	ANGR2
10^6	10^{-6}	1369.7	901.6	910.1	456.7	1409.0	820.7	891.7	781.2
	10^{-9}	8078.3	7186.9	6280.7	1570.7	6521.2	4501.7	4365.2	4491.8
	10^{-12}	16607.0	14221.9	11014.9	2465.7	10518.4	8257.9	8243.3	8213.9

comparable to CG even if the condition number is 10^6. However, CG clearly outperforms other methods for $\epsilon \leq 10^{-9}$.

Next we compared the above methods on a two-point boundary value problem [9, 19] which can be transferred as a linear system $Ax = b$ by the finite difference method. In particular, the matrix $A = (a_{ij})$ is given by

$$a_{ij} = \begin{cases} \frac{2}{h^2}, & \text{if} \quad i = j; \\ -\frac{1}{h^2}, & \text{if} \quad i = j \pm 1; \\ 0, & \text{otherwise}, \end{cases} \tag{40}$$

where $h = 11/n$. Clearly, the condition number κ increases as n becomes large. The vector b is defined as $b = Ax^*$ where x^* is a randomly generated as the former test.

We set $(h, s) = (30, 2)$ for the SDC method in this test. For ANGM, ANGR1 and ANGR2, the pair (τ_1, τ_2) was to (0.2, 1.02). Other setting are the same as above. Table 5 presents the number of iterations for the compared methods to meet different accuracy requirements for $n = 500, 1000, 2000, 3000, 5000$ using the starting point $x_0 = e$ where e is the vector of all ones. It can be seen that, the compared methods are often better than CG when $\epsilon = 10^{-6}$ though CG is the winner for the case $\epsilon = 10^{-9}$. In addition, for each n, our three methods outperform BB1, DY, SDC and ABBmin2 especially when the accuracy is high.

To confirm the previous results obtained for our new methods on quadratics with large condition numbers under relative low accuracy, we tested the above methods on some quadratics with the matrix A chosen from the University of Florida Sparse Matrix Collection [30] and $b = Ae$. Table 6 lists the names, dimensions, numbers of nonzero entries and condition numbers of 20 tested matrices, where "–" means the condition number for the matrix is not given by the collection.

The null vector was employed as the starting point. We stopped the iteration of the compared methods once $\|g_k\| \leq 10^{-6}\|g_0\|$. We chose parameters for our methods in the same way as the above random problems, and set the pair (τ_1, τ_2) to (0.1, 1.1), (0.1, 1.02) and (0.1, 1.02) for ANGM, ANGR1 and ANGR2, respectively. Other settings are the same as the above two-point boundary value problems.

We present the number of iterations required by each method in Table 7. It is observed that for most of the test problems our three methods perform better than the BB1, DY, SDC and ABBmin2 methods in terms of number of iterations. As compared with the CG method, our three methods provide competitive results. For LFAT5000, "–" is filled for the CG method since it takes more than 50000 iterations to satisfy the stopping condition. A possible reason for this phenomenon is that the

Table 5 The number of iterations for the ANGM, ANGR1, ANGR2, BB1, DY, SDC, CG and ABBmin2 methods on the two-point boundary value problem

n	ϵ	BB1	DY	SDC	CG	ABBmin2	ANGM	ANGR1	ANGR2
500	10^{-3}	29	25	36	28	33	48	32	28
	10^{-6}	758	559	799	500	1033	713	609	603
	10^{-9}	3410	2484	4033	501	1672	1763	2158	1682
1000	10^{-3}	30	25	36	29	26	30	30	30
	10^{-6}	663	555	761	819	1049	606	508	534
	10^{-9}	6346	4423	6887	1001	3278	2942	2622	3092
2000	10^{-3}	36	27	38	31	29	50	28	28
	10^{-6}	578	607	635	1500	558	526	424	503
	10^{-9}	14102	19640	7359	2001	6136	4567	4599	4050
3000	10^{-3}	30	25	36	30	26	42	30	30
	10^{-6}	671	579	543	2160	665	518	522	434
	10^{-9}	12104	17109	14977	3001	8125	9153	9768	8210
5000	10^{-3}	29	25	36	30	26	32	30	30
	10^{-6}	580	503	613	1370	663	429	443	552
	10^{-9}	29211	13667	15217	5001	14674	10107	9912	10158

Table 6 Matrices from the University of Florida sparse matrix collection

Matrices	n	Nonzeros	Condition number
LFAT5000	19994	79966	5.132558×10^{17}
bcsstk14	1806	63454	1.192324×10^{10}
bcsstk15	3948	117816	6.538185×10^{9}
bcsstk16	4884	290378	4.943183×10^{9}
bcsstk17	10974	428650	1.296064×10^{10}
bcsstk18	11948	149090	3.459995×10^{11}
bundle1	10581	770811	1.004238×10^{3}
cbuckle	13681	676515	3.299134×10^{7}
cvxbqp1	50000	349968	–
ex15	6867	98671	8.612330×10^{12}
gyro	17361	1021159	1.095832×10^{9}
m_t1	97578	9753570	–
s3dkq4m2	90449	4427725	1.896133×10^{11}
s3dkt3m2	90449	3686223	3.625322×10^{11}
s3rmq4m1	5489	262943	1.765559×10^{10}
s3rmt3m1	5489	217669	2.481977×10^{10}
CurlCurl_3	1219574	13544618	–
atmosmodl	1489752	10319760	–
atmosmodm	1489752	10319760	–
Offshore	259789	4242673	–

Table 7 The number of iterations for the ANGM, ANGR1, ANGR2, BB1, DY, SDC, CG and ABBmin2 methods on quadratics with A given in Table 6

problem	BB1	DY	SDC	CG	ABBmin2	ANGM	ANGR1	ANGR2
LFAT5000	16665	27443	14559	–	19854	9032	11288	**8575**
bcsstk14	2429	3183	3135	3096	3440	**2251**	2325	2924
bcsstk15	6429	4664	4095	4531	6911	3892	4649	**3315**
bcsstk16	536	350	417	**229**	397	479	369	402
bcsstk17	11494	12688	10879	9437	20365	9536	9554	**9339**
bcsstk18	3826	5180	5247	9593	5964	**3498**	3584	3822
bundle1	392	297	323	**129**	201	210	266	222
cbuckle	6894	7364	7905	**1820**	5254	4651	4392	4276
cvxbqp1	**250**	259	304	369	312	317	290	266
ex15	4100	4222	2911	**972**	2296	2459	3000	2037
gyro	10377	10175	11495	10554	11695	9427	11030	**8119**
m_t1	1925	1648	1535	9265	1785	**1327**	1689	1642
s3dkq4m2	9464	14136	7519	10747	11385	**5839**	6771	7826
s3dkt3m2	10397	30195	24040	15461	14188	9741	**8235**	9120
s3rmq4m1	10950	8603	7201	**3125**	6531	4742	7076	5630
s3rmt3m1	14894	21703	11041	**4483**	5644	4513	6561	5988
CurlCurl_3	5240	4008	3937	3922	6315	3724	**2949**	3326
atmosmodl	359	**262**	353	531	352	**262**	304	272
atmosmodm	220	219	257	3323	244	**168**	176	**168**
offshore	4775	4520	3871	18724	5881	**2825**	4205	2932

For each problem, the best results among the compared methods are marked in bold

conjugacy property of directions generated by CG is lost during the iteration process because rounding error may lead to inexact line search which is key to the conjugacy property, see [7] for example.

To further investigate the performance of the compared methods, we present the CPU time in seconds costed by each method to meet the given accuracy in Table 8. We again observe that our ANGM method is comparable to CG and faster than other methods while ANGR1 and ANGR2 are faster than BB1, DY, SDC, and ABBmin2. This can clearly be seen in the performance profiles (see [13]) on the CPU time metric plotted in Fig. 3 where the vertical axis shows the percentage of problems the method solves within factor ρ of the metric used by the most effective method in this comparison.

5 Conclusions

We have developed techniques to accelerate the Barzilai–Borwein (BB) method motivated from finite termination for minimizing two-dimensional strongly convex quadratic functions. More precisely, by exploiting certain orthogonal properties of the gradients, we derive a new monotone stepsize that can be combined with BB

Table 8 The CPU time in seconds for the ANGM, ANGR1, ANGR2, BB1, DY, SDC, CG and ABBmin2 methods on quadratics with A given in Table 6

Problem	BB1	DY	SDC	CG	ABBmin2	ANGM	ANGR1	ANGR2
LFAT5000	3.10	5.39	2.74	–	7.11	1.70	2.28	**1.70**
bcsstk14	**0.17**	0.21	0.22	0.22	0.47	0.22	0.18	0.22
bcsstk15	1.22	0.88	0.74	0.95	3.05	**0.54**	0.90	0.87
bcsstk16	0.24	0.15	0.18	**0.11**	0.55	0.22	0.25	0.23
bcsstk17	8.65	9.61	8.73	**6.76**	25.70	7.81	7.54	15.50
bcsstk18	1.35	1.69	1.94	3.45	4.93	**1.29**	1.30	1.37
bundle1	0.62	0.42	0.54	**0.20**	0.58	0.31	0.39	0.42
cbuckle	8.45	8.91	9.31	**2.02**	10.63	7.81	6.07	5.80
cvxbqp1	**0.20**	0.20	0.28	0.32	0.48	0.24	0.26	0.22
ex15	0.62	0.53	0.46	**0.15**	0.71	0.38	0.43	0.29
gyro	16.61	16.20	17.91	17.80	40.56	14.51	16.94	**13.29**
m_t1	29.87	27.45	24.92	129.62	46.87	**17.66**	22.72	22.72
s3dkq4m2	58.21	88.86	46.48	68.61	145.63	**36.04**	42.36	48.73
s3dkt3m2	55.50	159.72	126.64	83.41	144.76	52.06	**44.09**	48.79
s3rmq4m1	3.73	2.97	2.43	**1.28**	4.57	1.58	2.53	2.00
s3rmt3m1	3.91	5.74	2.88	**1.21**	2.95	1.23	1.77	1.68
CurlCurl_3	148.35	112.16	111.82	116.79	348.05	118.67	**94.97**	107.43
atmosmodl	9.58	**6.84**	9.36	14.91	13.07	7.92	9.38	8.37
atmosmodm	5.74	5.70	6.72	91.76	9.04	**5.01**	5.28	5.13
offshore	49.38	45.52	39.30	194.48	115.36	**30.10**	45.72	33.21

For each problem, the best results among the compared methods are marked in bold

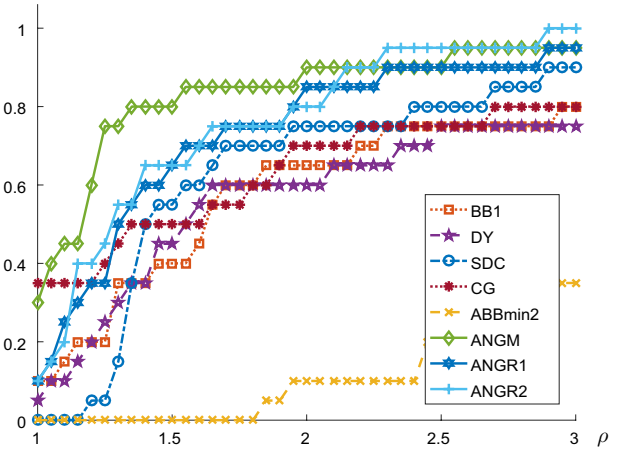

Fig. 3 Performance profiles of the compared methods on quadratics with A given in Table 6, CPU time metric

stepsizes to significantly improve their performance on quadratic optimization. By adaptively using this new stepsize and the two BB stepsizes, we develop a new gradient method called ANGM and its two variants ANGR1 and ANGR2. Our numerical experiments show that the newly developed methods are competitive with some very successful gradient methods developed in the recent literature and are comparable to the conjugate gradient method for solving quadratic optimization under low accuracy requirement especially when the condition number of the Hessian is large.

Acknowledgements The authors would like to thank the associate editor and the anonymous referees for their valuable comments and suggestions. This work was supported by the National Natural Science Foundation of China (Grant Nos. 11701137, 11631013, 12071108, 11671116, 11991021, 12021001), the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No. XDA27000000), Beijing Academy of Artificial Intelligence (BAAI), China Scholarship Council (No. 201806705007), Natural Science Foundation of Hebei Province (Grant No. A2021202010), and USA National Science Foundation (Grant Nos. 2110722, 1819161).

Declarations

Conflict of interest All data generated or analysed during this study are available from the corresponding author on reasonable request.

References

1. Akaike, H.: On a successive transformation of probability distribution and its application to the analysis of the optimum gradient method. Ann. Inst. Stat. Math. **11**(1), 1–16 (1959)
2. Barzilai, J., Borwein, J.M.: Two-point step size gradient methods. IMA J. Numer. Anal. **8**(1), 141–148 (1988)
3. Cauchy, A.: Méthode générale pour la résolution des systemes déquations simultanées. Comp. Rend. Sci. Paris **25**, 536–538 (1847)
4. Dai, Y.H.: Alternate step gradient method. Optimization **52**(4–5), 395–415 (2003)
5. Dai, Y.H., Fletcher, R.: Projected Barzilai–Borwein methods for large-scale box-constrained quadratic programming. Numer. Math. **100**(1), 21–47 (2005)
6. Dai, Y.H., Huang, Y., Liu, X.W.: A family of spectral gradient methods for optimization. Comp. Optim. Appl. **74**(1), 43–65 (2019)
7. Dai, Y.H., Kou, C.X.: A Barzilai–Borwein conjugate gradient method. Sci. China Math. **59**(8), 1511–1524 (2016)
8. Dai, Y.H., Liao, L.Z.: *R*-linear convergence of the Barzilai and Borwein gradient method. IMA J. Numer. Anal. **22**(1), 1–10 (2002)
9. Dai, Y.H., Yuan, Y.X.: Alternate minimization gradient method. IMA J. Numer. Anal. **23**(3), 377–393 (2003)
10. Dai, Y.H., Yuan, Y.X.: Analysis of monotone gradient methods. J. Ind. Manag. Optim. **1**(2), 181–192 (2005)
11. De Asmundis, R., Di Serafino, D., Hager, W.W., Toraldo, G., Zhang, H.: An efficient gradient method using the yuan steplength. Comp. Optim. Appl. **59**(3), 541–563 (2014)
12. Di Serafino, D., Ruggiero, V., Toraldo, G., Zanni, L.: On the steplength selection in gradient methods for unconstrained optimization. Appl. Math. Comput. **318**, 176–195 (2018)
13. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**(2), 201–213 (2002)
14. Fletcher, R.: On the Barzilai–Borwein method. In: Qi, L., Teo, K., Yang, X. (eds.) Optimization and Control with Applications, pp. 235–256. Springer, Boston (2005)
15. Fletcher, R.: A limited memory steepest descent method. Math. Program. **135**(1–2), 413–436 (2012)

16. Hestenes, M.R., Stiefel, E.: Method of conjugate gradient for solving linear system. J. Res. Nat. Bur. Stand. **49**, 409–436 (1952)
17. Forsythe, G.E.: On the asymptotic directions of the s-dimensional optimum gradient method. Numer. Math. **11**(1), 57–76 (1968)
18. Frassoldati, G., Zanni, L., Zanghirati, G.: New adaptive stepsize selections in gradient methods. J. Ind. Manag. Optim. **4**(2), 299–312 (2008)
19. Friedlander, A., Martínez, J.M., Molina, B., Raydan, M.: Gradient method with retards and generalizations. SIAM J. Numer. Anal. **36**(1), 275–289 (1998)
20. Golub, G.H., Van Loan, C.F.: Matrix Computations, 4th edn. Johns Hopkins University Press, Baltimore, Maryland (2013)
21. Gonzaga, C.C., Schneider, R.M.: On the steepest descent algorithm for quadratic functions. Comp. Optim. Appl. **63**(2), 523–542 (2016)
22. Huang, Y., Dai, Y.H., Liu, X.W., Zhang, H.: Gradient methods exploiting spectral properties. Optim. Method Softw. **35**(4), 681–705 (2020)
23. Huang, Y., Dai, Y.H., Liu, X.W., Zhang, H.: On the asymptotic convergence and acceleration of gradient methods. J. Sci. Comput. **90**, 7 (2022)
24. Huang, Y., Liu, H., Zhou, S.: Quadratic regularization projected Barzilai–Borwein method for nonnegative matrix factorization. Data Min. Knowl. Disc. **29**(6), 1665–1684 (2015)
25. Jiang, B., Dai, Y.H.: Feasible Barzilai–Borwein-like methods for extreme symmetric eigenvalue problems. Optim. Method Softw. **28**(4), 756–784 (2013)
26. Liu, Y.F., Dai, Y.H., Luo, Z.Q.: Coordinated beamforming for miso interference channel: complexity analysis and efficient algorithms. IEEE Trans. Signal Process. **59**(3), 1142–1157 (2011)
27. Raydan, M.: On the Barzilai and Borwein choice of steplength for the gradient method. IMA J. Numer. Anal. **13**(3), 321–326 (1993)
28. Raydan, M.: The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. SIAM J. Optim. **7**(1), 26–33 (1997)
29. Tan, C., Ma, S., Dai, Y.H., Qian, Y.: Barzilai–Borwein step size for stochastic gradient descent. In: Advances in Neural Information Processing Systems, pp. 685–693 (2016)
30. Davis, T.A., Hu, Y.: The university of Florida sparse matrix collection. ACM Trans. Math. Softw. **38**(1), 1–25 (2011)
31. Wang, Y., Ma, S.: Projected Barzilai–Borwein method for large-scale nonnegative image restoration. Inverse Probl. Sci. En. **15**(6), 559–583 (2007)
32. Wright, S.J., Nowak, R.D., Figueiredo, M.A.: Sparse reconstruction by separable approximation. IEEE Trans. Signal Process. **57**(7), 2479–2493 (2009)
33. Yuan, Y.X.: A new stepsize for the steepest descent method. J. Comput. Math. **24**(2), 149–156 (2006)
34. Yuan, Y.X.: Step-sizes for the gradient method. AMS IP Stud. Adv. Math. **42**(2), 785–796 (2008)
35. Zhou, B., Gao, L., Dai, Y.H.: Gradient methods with adaptive step-sizes. Comp. Optim. Appl. **35**(1), 69–86 (2006)