Gradient Methods with Adaptive Step-sizes

BIN ZHOU* LI GAO†
School of Mathematical Sciences and LMAM, Peking University, Beijing, 100871, People's Republic of China.
AND
YU-HONG DAI‡
State Key Laboratory of Scientific and Engineering Computing, Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and System Sciences, Chinese Academy of Sciences, P. O. Box 2719, Beijing, 100080, People's Republic of China.

Abstract

Motivated by the superlinear behavior of the Barzilai-Borwein (BB) method for two-dimensional quadratics, we propose two gradient methods which adaptively choose a small step-size or a large step-size at each iteration. The small step-size is primarily used to induce a favorable descent direction for the next iteration, while the large step-size is primarily used to produce a sufficient reduction. Although the new algorithms are still linearly convergent in the quadratic case, numerical experiments on some typical test problems indicate that they compare favorably with the BB method and some other efficient gradient methods.

Keywords: linear system; gradient method; adaptive step-size; Barzilai-Borwein method; superlinear behavior; trust-region approach.

1 Introduction

We consider the minimization problem of a quadratic function

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

where $A \in R^{n\times n}$ is symmetric and positive definite (SPD), and $b, x \in R^n$. It is well-known that this problem is equivalent to solving the linear system $Ax = b$.

*Email: bzhou@math.pku.edu.cn
†Email: gaol@pku.edu.cn
‡Email: dyh@lsec.cc.ac.cn

The gradient method for (1) can be defined by the iteration

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

where $g_k = Ax_k - b$ and α_k is a step-size depending on the line search applied.

For instance, the classical steepest descent (SD) method [8] determines α_k by

$$\alpha_k^{SD} = \frac{g_k^t g_k}{g_k^t A g_k}, \tag{3}$$

which minimizes the function value $f(x)$ along the ray $\{x_k - \alpha g_k : \alpha > 0\}$.

Another straightforward line search is to minimize the gradient norm $\|g(x)\|_2$ along the ray $\{x_k - \alpha g_k : \alpha > 0\}$—we name the associated algorithm "minimal gradient (MG) method" for convenience. Trivial deductions (see [11]) yield

$$\alpha_k^{MG} = \frac{g_k^t A g_k}{g_k^t A^2 g_k}. \tag{4}$$

Moreover, by the Cauchy inequality, we can prove that $\alpha_k^{MG} \leqslant \alpha_k^{SD}$.

Despite the optimal properties, the SD and MG methods behave poorly in most cases. Specifically, Akaike [1] showed that the sequence $\{x_k\}$ generated by the SD method tends to zigzag in two orthogonal directions, which usually implies deteriorations in convergence. Early efforts to improve this method gave rise to the development of the conjugate gradient (CG) method [16].

In 1988, Barzilai and Borwein [3] developed another two formulae for α_k ($k > 0$):

$$\alpha_k^{BB1} = \frac{s_{k-1}^t s_{k-1}}{s_{k-1}^t y_{k-1}} = \frac{g_{k-1}^t g_{k-1}}{g_{k-1}^t A g_{k-1}} = \alpha_{k-1}^{SD}, \tag{5}$$

$$\alpha_k^{BB2} = \frac{s_{k-1}^t y_{k-1}}{y_{k-1}^t y_{k-1}} = \frac{g_{k-1}^t A g_{k-1}}{g_{k-1}^t A^2 g_{k-1}} = \alpha_{k-1}^{MG}, \tag{6}$$

where $s_{k-1} = x_k - x_{k-1}$ and $y_{k-1} = g_k - g_{k-1}$. These two step-sizes minimize $\|\alpha^{-1} s_{k-1} - y_{k-1}\|_2$ and $\|s_{k-1} - \alpha y_{k-1}\|_2$ respectively, providing a scalar approximation to each of the secant equations $B_k s_{k-1} = y_{k-1}$ and $H_k y_{k-1} = s_{k-1}$.

The BB method ((2)∼(5) or (2)∼(6)) performs much better than the SD method in practice (see also [12]). Especially when $n = 2$, it converges R-superlinearly to the global minimizer [3]. In any dimension, it is still globally convergent [21] but the convergence is R-linear [10]. Raydan [22] adapted the method to unconstrained optimization by incorporating the non-monotone line search of Grippo *et al.* [15]. Since this work, the BB method has been successfully extended to many fields such as convex constrained optimization [2, 4, 5, 6, 7], nonlinear systems [17, 18], support vector machines [24], *etc.*

Inspired by the BB method, Friedlander *et al.* [14] introduced a family of gradient methods with retards (GMR). Given positive integers m and $q_i (i = 1, ..., m)$, the GMR sets the step-size as

$$\alpha_k^{GMR} = \frac{g_{\nu(k)}^t A^{\rho(k)-1} g_{\nu(k)}}{g_{\nu(k)}^t A^{\rho(k)} g_{\nu(k)}}, \tag{7}$$

where

$$\begin{aligned} &\nu(k) \in \{k, k-1, ..., \max\{0, k-m\}\}, \\ &\rho(k) \in \{q_1, q_2, ..., q_m\}. \end{aligned} \tag{8}$$

It is clear that the step-sizes (3)–(6) are all special cases of (7). [23] and [9] investigated a particular member of the GMR, in which the SD and BB1 step-sizes are used by turns. This algorithm, known as the alternate step (AS) gradient method in [9], has proved to be a promising alternative to the BB method. Another remarkable member of the GMR is the alternate minimization (AM) gradient method [11], which takes alternately the SD and MG step-sizes, and shows to be efficient for computing a low-precision solution. For further information on the GMR, see [19] and [20].

In this paper, we propose two gradient methods which adaptively choose a small step-size or a large one at every iteration. Here the small step-size serves to induce a favorable descent direction for the next iteration, while the large one serves to produce a sufficient reduction. To be more exact, in the first method, we combine the SD and MG step-sizes in a dynamical way by which the worst-case behaviors of the SD and MG methods are prevented at the same time; and in the second method, we derive a trust-region-like strategy by which we are able to choose step-size between the two alternatives of the original BB method. For strictly convex quadratics, we prove that the new algorithms converge $Q-$linearly and $R-$linearly, respectively. Numerical results on some typical test problems are presented, which suggest that our methods are competitive with the CG method and generally preferable to the BB, AS, and AM methods.

The rest of the paper is organized as follows. In the next section, we describe our motivating ideas and derive the new methods. Two simple numerical examples are presented to illustrate their behaviors. In Section 3 we provide convergence analyses. In Section 4, we present additional numerical results to compare our methods with the CG method as well as with the BB, AS, and AM methods. Finally, in Section 5, we present discussions and concluding remarks.

2 Derivation of the new methods

Barzilai and Borwein [3] proved that the BB method converges R-superlinearly for two-dimensional quadratics, which can be illustrated by the following numerical

phenomenon: As k increases, the gradient g_k generated by the BB method tends to be an eigenvector of the matrix A. Unfortunately, this phenomenon does not evidently hold for any-dimensional case where only R-linear convergence is obtained [10]. We now get a heuristic idea as below: *Assume the eigenvalues of A can be simply classified as the large ones and the small ones, as if there were only two different eigenvalues, and suppose we can frequently make g_k approach some large or small eigenvectors (by which we mean eigenvectors associated with the large or small eigenvalues) of A, then the resulting algorithm might exhibit certain superlinear behavior and outperform the BB method when n is very large.*

In order to realize this idea, we choose to modify the SD method. Thus let us go back to this old approach for more inspirations. Without loss of generality, we assume that A is a diagonal matrix in our derivation. The following theorem illustrates the worst-case behavior of the SD method in two dimensions. (In any dimension, the worst-case behaviors of the SD and MG methods are only related to the smallest and largest eigenvalues of A. Hence we just need to consider the two-dimensional case.)

Theorem 2.1 *Consider the minimization problem* (1)*, where $A = diag(\lambda_1, \lambda_2)$ with $\lambda_2 \geqslant \lambda_1 > 0$. Let $x^* = A^{-1}b$, and let $\{x_k\}$ be the sequence generated by the SD method* (2)$\sim$(3)*. If $g_0 \parallel (1, \pm 1)^t$, then*

$$\frac{f(x_{k+1}) - f(x^*)}{f(x_k) - f(x^*)} = \left(\frac{\lambda_2 - \lambda_1}{\lambda_2 + \lambda_1}\right)^2 \quad \textit{for all } k \geqslant 0,$$

which means the SD method reaches its slowest convergence rate.

Proof: To prove this theorem, it suffices to show that if $g_0 \parallel (1, \pm 1)$, then we must have

$$g_1 \parallel (1, \mp 1)^t \quad \text{and} \quad \frac{f(x_1) - f(x^*)}{f(x_0) - f(x^*)} = \left(\frac{\lambda_2 - \lambda_1}{\lambda_2 + \lambda_1}\right)^2. \tag{9}$$

Assume $g_0 = (u, \pm u)^t$ with $u \neq 0$. Substituting it into (3), we have

$$\alpha_0^{SD} = \frac{2u^2}{(\lambda_2 + \lambda_1)u^2} = \frac{2}{\lambda_2 + \lambda_1}. \tag{10}$$

Moreover, by (2), we have

$$g_1 = Ax_1 - b = A(x_0 - \alpha_0^{SD} g_0) - b = g_0 - \alpha_0^{SD} A g_0,$$

which, together with (10) and $g_0 = (u, \pm u)^t$, yields

$$g_1 = \frac{\lambda_2 - \lambda_1}{\lambda_2 + \lambda_1} u\,(1, \mp 1)^t. \tag{11}$$

For the second part of (9), we have by Taylor's theorem and $\nabla f(x^*) = 0$ that

$$\begin{aligned} f(x_k) - f(x^*) &= (x_k - x^*)^t \nabla f(x^*) + \frac{(x_k - x^*)^t A (x_k - x^*)}{2} \\ &= \frac{(Ax_k - Ax^*)^t A^{-1} (Ax_k - Ax^*)}{2} \\ &= \frac{(Ax_k - b)^t A^{-1} (Ax_k - b)}{2} = \frac{g_k^t A^{-1} g_k}{2}. \end{aligned}$$

Then by (11) and $g_0 = (u, \pm u)^t$, we have

$$\begin{aligned} \frac{f(x_1) - f(x^*)}{f(x_0) - f(x^*)} &= \frac{g_1^t A^{-1} g_1}{g_0^t A^{-1} g_0} = \left(\frac{\lambda_2 - \lambda_1}{\lambda_2 + \lambda_1}\right)^2 \frac{(1, \mp 1)^t A^{-1} (1, \mp 1)}{(1, \pm 1)^t A^{-1} (1, \pm 1)} \\ &= \left(\frac{\lambda_2 - \lambda_1}{\lambda_2 + \lambda_1}\right)^2 \frac{\lambda_1^{-1} + \lambda_2^{-1}}{\lambda_1^{-1} + \lambda_2^{-1}} = \left(\frac{\lambda_2 - \lambda_1}{\lambda_2 + \lambda_1}\right)^2. \end{aligned}$$

Thus we have by induction that $\dfrac{f(x_{k+1}) - f(x^*)}{f(x_k) - f(x^*)} = \left(\dfrac{\lambda_2 - \lambda_1}{\lambda_2 + \lambda_1}\right)^2$ for all $k \geqslant 0$. □

We can see from Theorem 2.1 that: While g_k is about 45° away from any eigenvector of A, the convergence of the SD method deteriorates. Naturally we hope that the next gradient g_{k+1} will approach some eigenvector of A. But how to reset the k-th step-size? The theorem below shows that the MG step-size is a desirable option when the SD step-size is unfavorable.

Theorem 2.2 *Consider the minimization problem* (1)*, where* $A = diag(\lambda_1, \lambda_2)$ *with* $\lambda_2 \geqslant \lambda_1 > 0$*. Let* $\{x_k\}$ *be the sequence generated by a gradient method. If* $g_k \parallel (1, \pm 1)^t$*, i.e.* $g_k = (u, \pm u)^t$ *with* $u \neq 0$*, then it holds that*

$$\alpha_k^{MG} / \alpha_k^{SD} > 0.5$$

and

$$g_{k+1} = \begin{cases} \dfrac{\lambda_2 - \lambda_1}{\lambda_2 + \lambda_1} u(1, \mp 1)^t, & \text{if } \alpha_k = \alpha_k^{SD}; \\ \dfrac{\lambda_2 - \lambda_1}{\lambda_2^2 + \lambda_1^2} u(\lambda_2, \mp \lambda_1)^t, & \text{if } \alpha_k = \alpha_k^{MG}. \end{cases} \tag{12}$$

Proof: Substituting $g_k = (u, \pm u)^t$ into (3) and (4), we have

$$\alpha_k^{SD} = \frac{2}{\lambda_1 + \lambda_2} \quad \text{and} \quad \alpha_k^{MG} = \frac{\lambda_1 + \lambda_2}{\lambda_1^2 + \lambda_2^2}. \tag{13}$$

Therefore,

$$\frac{\alpha_k^{MG}}{\alpha_k^{SD}} = \frac{(\lambda_1 + \lambda_2)^2}{2(\lambda_1^2 + \lambda_2^2)} > 0.5.$$

And by (2) we have

$$g_{k+1} = Ax_{k+1} - b = Ax_k - \alpha_k Ag_k - b = g_k - \alpha_k Ag_k. \tag{14}$$

Then we obtain (12) from (13) and (14) by direct calculations. □

Since $\lambda_2 \geqslant \lambda_1 > 0$, it is easy to see that the choice of α_k^{MG} makes g_{k+1} more inclined to the eigenvector direction $(1,0)^t$ or $(-1,0)^t$ than the choice of α_k^{SD}. Moreover, it holds that $\alpha_k^{MG}/\alpha_k^{SD} > 0.5$ when $g_k \parallel (1,\pm 1)^t$. Thus we can set $\alpha_k = \alpha_k^{MG}$ to avoid the worst-case behavior of the SD method, and let the inequality $\alpha_k^{MG}/\alpha_k^{SD} > 0.5$ be the corresponding switch criterion. More precisely, we have the following scheme for choosing the step-size:

$$\alpha_k = \begin{cases} \alpha_k^{MG}, & \text{if } \alpha_k^{MG}/\alpha_k^{SD} > \kappa; \\ \alpha_k^{SD}, & \text{otherwise,} \end{cases} \tag{15}$$

where $\kappa \in (0,1)$ is a parameter close to 0.5.

One might ask this question: Does the scheme (15) also avoid the worst-case behavior of the MG method? The answer is yes when the condition number of A is big enough. To make it comprehensible, we first present a theorem illustrating the worst-case behavior of the MG method in two dimensions.

Theorem 2.3 *Consider the minimization problem* (1)*, where* $A = diag(\lambda_1, \lambda_2)$ *with* $\lambda_2 \geqslant \lambda_1 > 0$*. Let* $\{x_k\}$ *be the sequence generated by the MG method* (2)∼(4)*. If* $g_0 \parallel (\sqrt{\lambda_2}, \pm\sqrt{\lambda_1})^t$*, then it holds for all* $k \geqslant 0$
(a) *The MG method reaches its slowest convergence rate:*

$$\frac{\|g_{k+1}\|_2^2}{\|g_k\|_2^2} = \left(\frac{\lambda_2 - \lambda_1}{\lambda_2 + \lambda_1}\right)^2.$$

(b) *The ratio* $\alpha_k^{MG}/\alpha_k^{SD}$ *reaches its smallest value:*

$$\frac{\alpha_k^{MG}}{\alpha_k^{SD}} = \frac{4\lambda_1\lambda_2}{(\lambda_1+\lambda_2)^2} = \min_{g\neq 0} \frac{(g^tAg)^2}{(g^tg)(g^tA^2g)}.$$

The proof of (a) is quite similar to that of Theorem 2.1, while (b) can be obtained inductively by direct calculations (the second equality herein is an equivalent form of the Kantorovich inequality). Supposing $\lambda_2 \gg \lambda_1$, we can see that $\alpha_k^{MG}/\alpha_k^{SD} \approx 0$ and g_0 is almost parallel to the eigenvector $(1,0)^t$. That is to say, the worst-case behavior of the MG method is usually due to the excessively small step-size and not to the descent direction. Scheme (15) sets $\alpha_k = \alpha_k^{SD}$ when $\alpha_k^{MG}/\alpha_k^{SD} \leqslant \kappa$; therefore it is able to avoid the worst-case behavior of the MG method for very ill-conditioned problems.

In a recent paper, Raydan and Svaiter [23] indicated that the SD method with (over and under) relaxation usually improves upon the original method. This idea was also considered by Dai and Yuan [11], in whose paper two shortened SD step gradient methods have been well compared with the AM and SD methods. Motivated by these works, we modify (15) as follows:

$$\alpha_k = \begin{cases} \alpha_k^{MG}, & \text{if } \alpha_k^{MG}/\alpha_k^{SD} > \kappa; \\ \alpha_k^{SD} - \delta\, \alpha_k^{MG}, & \text{otherwise,} \end{cases} \tag{16}$$

where $\kappa, \delta \in (0, 1)$. Notice that when $\alpha_k^{MG}/\alpha_k^{SD}$ reaches its smallest value, α_k^{SD} is the least shortened. As α_k can be viewed as an adaptively under-relaxed SD step-size (recall that $\alpha_k^{MG} \leqslant \alpha_k^{SD}$), we call the method (2)$\sim$(16) "adaptive steepest descent (ASD) method". Compared to the SD method, this method requires one extra inner product per iteration.

How the ASD method might overcome the drawbacks of the SD method and simulate the superlinear behavior of the BB method in two dimensions is depicted in Figure 1, where x_k^{SD} and x_k^{ASD} signify the iterates of the SD and ASD methods, respectively. We can observe that the ASD method ($\kappa, \delta = 0.5$) refines x_0 less than the SD method at the first iteration, but it induces a gradient inclined to the eigenvector direction $(-1, 0)^t$, thereby obtaining a considerable descent at the next iteration.

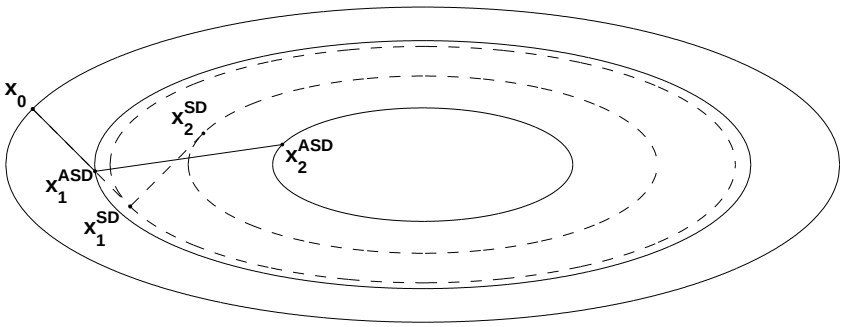

Figure 1: ASD vs. SD when $A = diag(1, 7)$, $g_0 \parallel (1, -1)^t$.

Before we show by a simple example that the ASD method is still efficient for solving any-dimensional problems, we first give a condition under which we should expect that the method still would exhibit certain superlinear behavior. Note that our derivation of the ASD method is totally based on the assumption that the matrix A has two different eigenvalues. We assume here that A has two clusters of eigenvalues. Then, in view of the above figure and theorems, we should expect that some of the descent directions generated by the ASD method would incline towards the eigenvectors that relate to the eigenvalues with small magnitude.

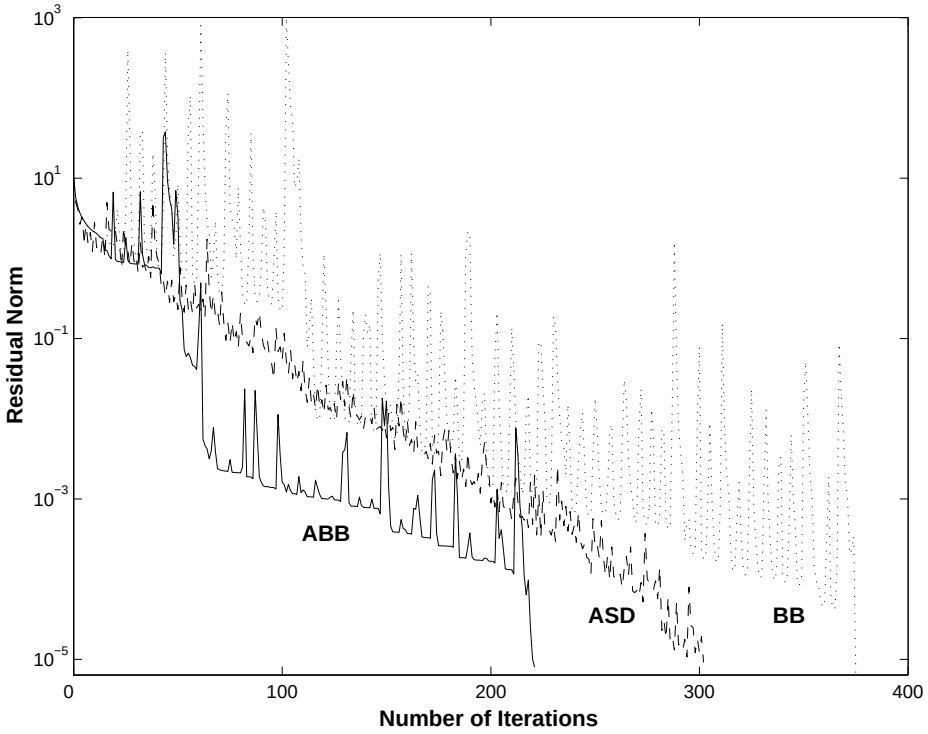

Figure 2: Performances of BB, ASD and ABB for a 100-dimensional problem.

To illustrate the performance of the ASD method in any dimension, a simple test problem is devised (see Section 4 for more experiments). We let $A = diag(\lambda_1, ..., \lambda_{100})$ with $\lambda_1 = 0.1$ and $\lambda_i = i$ for $i = 2, ..., 100$. The right-hand-side b is set as $(1, ..., 1)^t$ and the initial guess is chosen as $x_0 = 0 \in \Re^{100}$. In order to get $\|g_k\|_2 \leqslant 10^{-6}\|g_0\|_2$, the BB method ($\alpha_0 = \alpha_0^{SD}$) needs 375 iterations, while the ASD ($\kappa, \delta = 0.5$) method requires only 302 iterations. Figure 2 reports the gradient norms for the BB and ASD methods at each iteration (the figure also reports the gradient norms for a third method, ABB, which will be defined later). From this figure, we can observe that the ASD method gives the better solution at most iterations. Besides, this method seems to produce smaller oscillations than the BB method. As will be proved in Section 3, the ASD method is in fact a monotone algorithm.

Now let us extend (16) and derive an ASD-like variant of the BB method. Looking into the iterations of the ASD method for the above test problem, we find that the sign of $\alpha_k^{MG}/\alpha_k^{SD} - \kappa$ is opposite to that of $\alpha_{k-1}^{MG}/\alpha_{k-1}^{SD} - \kappa$ at most iterations (to be exact, 238 out of 302 iterations). Similar phenomenon is also observed in many other experiments (where $\kappa = 0.5$). For this reason, we replace $\alpha_k^{MG}/\alpha_k^{SD} > \kappa$ in

(16) with $\alpha_{k-1}^{MG}/\alpha_{k-1}^{SD} < \kappa$:

$$\alpha_k = \begin{cases} \alpha_k^{MG}, & \text{if } \alpha_{k-1}^{MG}/\alpha_{k-1}^{SD} < \kappa; \\ \alpha_k^{SD} - \delta\,\alpha_k^{MG}, & \text{otherwise.} \end{cases} \tag{17}$$

Since $\alpha_{k-1}^{SD} = \alpha_k^{BB1}$ and $\alpha_{k-1}^{MG} = \alpha_k^{BB2}$, we rewrite (17) as

$$\alpha_k = \begin{cases} \alpha_k^{MG}, & \text{if } \alpha_k^{BB2}/\alpha_k^{BB1} < \kappa; \\ \alpha_k^{SD} - \delta\,\alpha_k^{MG}, & \text{otherwise.} \end{cases}$$

In order to get a simple scheme, we replace $\alpha_k^{SD} - \delta\,\alpha_k^{MG}$ and α_k^{MG} with α_k^{BB1} and α_k^{BB2} respectively. So we finally have

$$\alpha_k = \begin{cases} \alpha_k^{BB2}, & \text{if } \alpha_k^{BB2}/\alpha_k^{BB1} < \kappa; \\ \alpha_k^{BB1}, & \text{otherwise,} \end{cases} \tag{18}$$

where $\kappa \in (0, 1)$. Here we do not shorten α_k^{BB1} because the BB method itself overcomes the drawbacks of the SD method. Analogously, we call the method (2)$\sim$(18) "adaptive Barzilai-Borwein (ABB) method". For the above test problem, this method ($\kappa = 0.5$, and $\alpha_0 = \alpha_0^{SD}$) requires only 221 iterations (see also Figure 2).

At first glance, the ABB method seems to require one more inner product per iteration than the BB method, but this extra computational work can be eliminated by using the following formulae:

$$\alpha_k^{BB1} = \frac{s_{k-1}^t s_{k-1}}{s_{k-1}^t y_{k-1}} = \alpha_{k-1} \frac{g_{k-1}^t g_{k-1}}{g_{k-1}^t g_{k-1} - g_{k-1}^t g_k},$$
$$\alpha_k^{BB2} = \frac{s_{k-1}^t y_{k-1}}{y_{k-1}^t y_{k-1}} = \alpha_{k-1} \frac{g_{k-1}^t g_{k-1} - g_{k-1}^t g_k}{g_{k-1}^t g_{k-1} - 2g_{k-1}^t g_k + g_k^t g_k}.$$

Here we have used $s_{k-1} = x_k - x_{k-1} = -\alpha_{k-1} g_{k-1}$ and $y_{k-1} = g_k - g_{k-1}$. Notice that at every iteration, we only need to compute two inner products, i.e. $g_{k-1}^t g_k$ and $g_k^t g_k$. Our extensive numerical experiments show that these formulae are reliable in the quadratic case. For non-quadratic functions, there might be negative or zero denominators, which could also happen while using the original formulae. Raydan [22] proposed a simple approach to deal with this situation.

Before concluding this section, we would like to give an intuitive explanation to (18). Recall that when $\alpha_k^{BB2}/\alpha_k^{BB1} = \alpha_{k-1}^{MG}/\alpha_{k-1}^{SD} \approx 0$, the MG method performs poorly at the point x_{k-1} (see Theorem 2.3); although we generally have $\alpha_{k-1} \neq \alpha_{k-1}^{MG}$ in the ABB method, there must be little reduction in $\|g(x)\|_2$ at the previous iteration since it is the MG step-size that minimizes $\|g(x)\|_2$ along the gradient descent direction. Thus we can interpret (18) as follows: *If the previous iterate x_{k-1} is a bad point for the MG method, and so that there is little reduction in $\|g(x)\|_2$,*

choose the smaller step-size α_k^{BB2}; otherwise, choose the larger step-size α_k^{BB1}. This explanation reminds us of the trust-region approach, which uses a somewhat similar strategy while choosing the trust-region radius. In addition, considering the good performance of the ABB method, it seems reasonable to conjecture that the smaller step-size used in ABB might be good at inducing a favorable descent direction, while the larger step-size might be good at producing a sufficient reduction.

3 Convergence rate analyses

We first analyze the convergence rate of the ASD method. Let x^* be the unique minimizer of the quadratic $f(x)$ in (1) and define $\|x\|_A = \sqrt{x^t A x}$. Then we have the following Q-linear convergence result for the ASD method.

Theorem 3.1 *Consider the minimization problem* (1)*. Let* $\{x_k\}$ *be the sequence generated by the ASD method* (2)$\sim$(16)*. Define*

$$s := min\{\kappa, 1-\kappa\} \quad and \quad c := \frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}, \tag{19}$$

where λ_1 and λ_n are the smallest and largest eigenvalues of A, respectively. Then it holds for all $k \geqslant 0$ that

$$\|x_{k+1} - x^*\|_A < \sqrt{c^2 + (1-c^2)(1-s)^2}\, \|x_k - x^*\|_A. \tag{20}$$

Thus the ASD method is Q-linearly convergent.

Proof: If $\alpha_k^{MG}/\alpha_k^{SD} \leqslant \kappa$, we have by $\delta \in (0,1)$ that

$$\begin{aligned} \alpha_k^{ASD} &= \alpha_k^{SD} - \delta \alpha_k^{MG} \\ &> \alpha_k^{SD} - \alpha_k^{MG} \\ &\geqslant (1-\kappa)\alpha_k^{SD}. \end{aligned} \tag{21}$$

Else if $\alpha_k^{MG}/\alpha_k^{SD} > \kappa$, we have that

$$\alpha_k^{ASD} = \alpha_k^{MG} > \kappa \alpha_k^{SD}. \tag{22}$$

On the other hand, it always holds that

$$\alpha_k^{ASD} \leqslant \alpha_k^{SD}. \tag{23}$$

Let $\theta_k = \alpha_k^{ASD}/\alpha_k^{SD}$. Then we have the following relation by (21)–(23):

$$s < \theta_k \leqslant 1. \tag{24}$$

To simplify notation, define $e_k = x_k - x^*$ and $E_k = \|e_k\|_A^2$. It is easy to see that

$$g_k = Ae_k \tag{25}$$

and

$$e_{k+1} = e_k - \theta_k \alpha_k^{SD} g_k. \tag{26}$$

Then we have by (25), (26) and (3) that

$$\begin{aligned} \frac{E_k - E_{k+1}}{E_k} &= \frac{e_k^t A e_k - e_{k+1}^t A e_{k+1}}{e_k^t A e_k} \\ &= \frac{2\theta_k \alpha_k^{SD} g_k^t A e_k - \theta_k^2 (\alpha_k^{SD})^2 g_k^t A g_k}{g_k^t A^{-1} g_k} \\ &= \frac{(2\theta_k - \theta_k^2)(g_k^t g_k)^2}{(g_k^t A g_k)(g_k^t A^{-1} g_k)}. \end{aligned} \tag{27}$$

Noting that $2\theta_k - \theta_k^2 = 1 - (1-\theta_k)^2$, we have by (24) that

$$2\theta_k - \theta_k^2 > 1 - (1-s)^2. \tag{28}$$

And by the Kantorovich inequality and (19), we have that

$$\frac{(g_k^t g_k)^2}{(g_k^t A g_k)(g_k^t A^{-1} g_k)} \geqslant \frac{4\lambda_1 \lambda_n}{(\lambda_1 + \lambda_n)^2} = 1 - c^2. \tag{29}$$

Applying (28) and (29) to (27), we obtain that

$$\frac{E_k - E_{k+1}}{E_k} > [1 - (1-s)^2](1 - c^2),$$

or equivalently,

$$E_{k+1} < \{1 - [1 - (1-s)^2](1-c^2)\} E_k = [c^2 + (1-c^2)(1-s)^2] E_k. \tag{30}$$

As $E_k = \|x_k - x^*\|_A^2$, the relation (30) is (20) in disguise. Notice also that

$$c^2 + (1-c^2)(1-s)^2 < c^2 + (1-c^2) = 1.$$

Then it follows from (20) that the ASD method is Q-linearly convergent. □

Observe that $\|x_k - x^*\|_A = 2(f(x_k) - f(x^*))$. Thus we have also proved that the ASD method is a monotone algorithm.

In addition, we can observe that (20) is worse than the convergence relation of the SD method, which is $\|x_{k+1} - x^*\|_A \leqslant c \|x_k - x^*\|_A$. Nonetheless, extensive numerical experiments show that the ASD method significantly outperforms the SD method. We think the explanation for this is as follows. As shown by (23), the ASD method

always takes a shortened SD step-size. Therefore, it may improve x_k quite slowly at some steps. On the other hand, it is the small improvements at some steps that bring out suitable descent directions and hence speed up the overall convergence of the method. (See Figure 1 for a geometric interpretation.)

Now we consider the convergence of the ABB method. Note that this method can be regarded as a particular member of the GMR (7)$\sim$(8) for which $\nu(k) = k-1$ and $\rho(k) \in \{1, 2\}$. Recently, the third author of this paper has proved that the GMR is R-linear convergent for any-dimensional quadratics (see [9], Corollary 4.2). Therefore, we have the following R-linear convergence result for the ABB method.

Theorem 3.2 *Consider the minimization problem* (1)*. Let* $\{x_k\}$ *be the sequence generated by the ABB method* (2)$\sim$(18)*. Then either* $g_k = 0$ *for some finite* k*, or the sequence* $\{\|g_k\|_2\}$ *converges to zero R-linearly.*

4 Numerical experiments

In this section, we compare the ASD and ABB methods with the CG, BB, AS and AM methods on some typical test problems. Here BB just denotes the method (2)$\sim$(5), which appears preferable to the version (2)$\sim$(6) in practice. Unless otherwise noted, the parameters in ASD and ABB are set to be $\kappa, \delta = 0.5$. All experiments were run on a 2.6GHz Pentium IV with 512MB of RAM, using double precision Fortran 90. The initial guess was always chosen as $x_0 = 0 \in \Re^n$, and the stopping criterion was chosen as

$$\|g_k\|_2 \leqslant \theta \|g_0\|_2$$

with different values of θ.

We now describe three different test problems and report their corresponding numerical results.

Example 1 (Random problems [14]): Consider the matrix $A = QDQ^t$, where

$$Q = (I - 2w_3w_3^t)(I - 2w_2w_2^t)(I - 2w_1w_1^t),$$

and w_1, w_2 and w_3 are random unitary vectors. $D = diag(\sigma_1, ..., \sigma_n)$ is a diagonal matrix where $\sigma_1 = 1$, $\sigma_n = cond$, and σ_i is randomly generated between 1 and $cond$ for $i = 2, ..., n-1$. The entries of the right-hand-side b are randomly generated between -10 and 10. In this experiment, we used $n = 5000$ and allowed a maximum 10000 of iterations.

In Table 1 we report the numbers of iterations required by different methods. The linear CG method is undoubtedly the most efficient approach. However, it seems not so appealing when a low precision is required. The ASD and ABB methods are both less efficient than the CG method. But they generally need fewer iterations than the BB, AS and AM methods. Particularly, when the condition number is very

Table 1: Numbers of iterations required by different methods for solving random problems. CG: conjugate gradient method; BB: Barzilai-Borwein method; AS: alternate step gradient method; AM: alternate minimization gradient method; ASD: adaptive steepest descent method; ABB: adaptive Barzilai-Borwein method.

cond	θ	CG	BB	AS	AM	ASD	ABB
10^2	10^{-1}	10	17	18	10	9	15
10^2	10^{-2}	22	28	30	24	23	34
10^2	10^{-3}	33	47	50	40	41	42
10^2	10^{-4}	45	82	54	59	61	54
10^2	10^{-5}	57	93	84	72	88	62
10^3	10^{-1}	26	17	18	12	13	15
10^3	10^{-2}	63	94	78	98	73	64
10^3	10^{-3}	101	187	136	196	144	119
10^3	10^{-4}	138	294	210	332	194	186
10^3	10^{-5}	169	318	262	470	213	219
10^4	10^{-1}	63	19	18	16	16	15
10^4	10^{-2}	174	236	216	298	201	175
10^4	10^{-3}	240	408	466	1206	315	305
10^4	10^{-4}	310	667	564	1688	470	494
10^4	10^{-5}	360	835	720	2049	648	629
10^5	10^{-1}	213	28	18	16	16	15
10^5	10^{-2}	289	772	640	1423	409	540
10^5	10^{-3}	350	1766	1376	1426	1324	1491
10^5	10^{-4}	388	2625	1576	6458	2553	1586
10^5	10^{-5}	461	3609	3376	9412	2826	1721
10^6	10^{-1}	291	28	18	16	16	15
10^6	10^{-2}	356	2552	3550	8686	1783	986
10^6	10^{-3}	391	4304	4428	>10000	3353	989
10^6	10^{-4}	465	8636	4428	>10000	3530	1016
10^6	10^{-5}	497	>10000	4474	>10000	5351	1042

large and a high precision is required, the ABB method clearly outperforms other gradient approaches.

It is very interesting to observe that, when $cond = 10^5$ and $cond = 10^6$, as the precision requirement increases, the ABB method requires only a few extra iterations. Similar phenomena, although not so obvious, can be observed in the results of the ASD and AS methods. These phenomena confirm our original expectation that the new methods might exhibit certain superlinear behavior when n is very large.

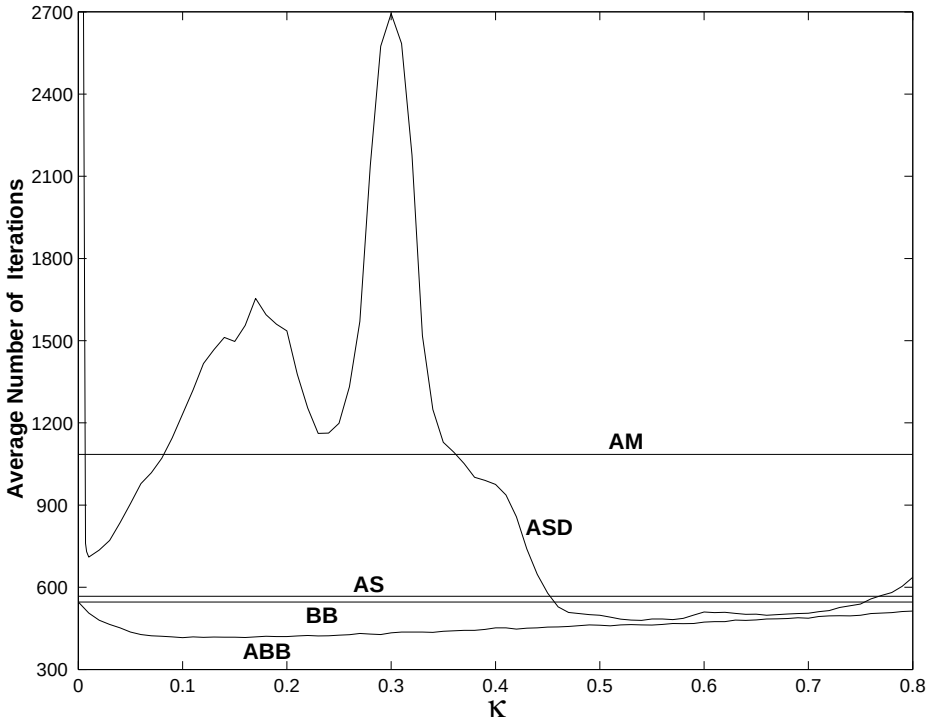

Figure 3: Average numbers of iterations for solving 1000 random problems.

To investigate the influence of the parameter κ on the performances of the new methods, we selected 80 different κ's, which range from 0.01 to 0.80. For each κ and each method, we tested 1000 random problems with $cond = 5000$. Depicted in Figure 3 are average numbers of iterations used by different methods to obtain $\|g_k\|_2 \leqslant 10^{-5}\|g_0\|_2$. We observe that the ABB method outperforms other methods for all κ's, and the best choice of κ seems to be $0.1 \leqslant \kappa \leqslant 0.2$. For the ASD method, different choice of κ makes great difference. However, it seems better to set κ slightly bigger than 0.5.

Example 2 (3D Laplace's equation [13]): In the second experiment, we tested a large-scale real problem, which is based on a 7-point finite difference ap-

Table 2: Numbers of iterations required by different methods for solving problem **L1**. CG: conjugate gradient method; BB: Barzilai-Borwein method; AS: alternate step gradient method; AM: alternate minimization gradient method; ASD: adaptive steepest descent method; ABB: adaptive Barzilai-Borwein method.

n	Problem	CG	BB	AS	AM	ASD	ABB
100^3	L1(a)	189	505	690	1282	413	392
	L1(b)	273	569	406	946	542	329
110^3	L1(a)	208	709	672	1434	561	558
	L1(b)	301	631	564	962	484	435
120^3	L1(a)	227	676	872	1540	597	543
	L1(b)	328	463	560	1432	494	452
130^3	L1(a)	246	670	696	1376	758	612
	L1(b)	356	532	546	978	587	411
140^3	L1(a)	265	930	796	1638	958	684
	L1(b)	384	770	612	1242	491	635
150^3	L1(a)	284	677	562	2236	934	438
	L1(b)	412	690	646	1490	638	646
160^3	L1(a)	303	1210	908	2144	789	816
	L1(b)	440	790	612	1584	715	605
170^3	L1(a)	322	1093	1000	2638	740	819
	L1(b)	468	864	844	2010	696	681
180^3	L1(a)	341	1159	868	2011	903	590
	L1(b)	496	945	946	2458	836	847

proximation to 3D Laplace's equation on a unitary cube. Define the matrices

$$T = \begin{bmatrix} 6 & -1 & & & \\ -1 & 6 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 6 & -1 \\ & & & -1 & 6 \end{bmatrix}, \qquad W = \begin{bmatrix} T & -I & & & \\ -I & T & -I & & \\ & \ddots & \ddots & \ddots & \\ & & -I & T & -I \\ & & & -I & T \end{bmatrix}$$

and

$$A = \begin{bmatrix} W & -I & & & \\ -I & W & -I & & \\ & \ddots & \ddots & \ddots & \\ & & -I & W & -I \\ & & & -I & W \end{bmatrix}, \tag{31}$$

where T is $m \times m$, W and A are block $m \times m$. Here m is the number of interior nodes in each coordinate direction. Obviously, the problem has m^3 variables. We

denote the solution by u^* and fix it to be function

$$u(x,y,z) = x(x-1)y(y-1)z(z-1)\exp\left(-\frac{\sigma^2((x-\alpha)^2+(y-\beta)^2+(z-\gamma)^2)}{2}\right) \tag{32}$$

evaluated at the nodal points. This function is a Gaussian centered at (α, β, γ) and multiplied by $x(x-1)y(y-1)z(z-1)$, which gives $u = 0$ on the boundary. The parameter σ controls the decay rate of the Gaussian. We calculate the right-hand-side $b = Au^*$ and denote the resulting problem to be **L1**. We stop the iteration when $\|g_k\|_2 \leqslant 10^{-6}\|g_0\|_2$ and choose the parameters in two different ways, i.e.

(a) $\sigma = 20,\ \alpha = \beta = \gamma = 0.5$; (b) $\sigma = 50,\ \alpha = 0.4,\ \beta = 0.7,\ \gamma = 0.5$.

The results for solving **L1** are listed in Table 2. We observe that the linear CG method is still the outright winner, and the ASD and ABB methods still outperform the BB, AS and AM methods in most cases, although the difference between the number of iterations is not very great. The ABB method seems to be the best gradient method. It requires the fewest iterations in 13 tests.

Example 3 (A non-quadratic problem [13]): To obtain some information about the ABB method for general functions, a non-quadratic test problem is derived from Example 2, in which the objective function is

$$\frac{1}{2}u^tAu - b^tu + \frac{1}{4}h^2\sum_{ijk}u_{ijk}^4. \tag{33}$$

This problem is referred to as **L2**. The matrix A is defined as in (31), and the vector b is chosen so that the minimizer of (33) is function (32) evaluated at the nodal points. We stop the iteration when $\|g_k\|_2 \leqslant 10^{-5}\|g_0\|_2$.

In Table 3, we report the time and numbers of evaluations required by the Polak-Ribière CG (PR-CG), unmodified BB and ABB methods. The columns headed #ls, #f and #g give the numbers of line searches, function evaluations and gradient evaluations to solve the problem. Note that the unmodified BB and ABB methods require no line searches and function evaluations. In the PR-CG method, we require the step-size to satisfy the standard strong Wolfe conditions with a relative slope tolerance of 0.1. From Table 3, we can observe that the unmodified BB and ABB methods greatly improve upon the PR-CG method, and the ABB method usually gives the best performance. These results suggest that the ABB method might surpass the PR-CG and BB methods while solving unconstrained optimization problems.

5 Concluding remarks and discussions

In this paper, by simulating the numerical behavior of the BB method for two-dimensional quadratics, we have introduced two gradient methods with adaptive

Table 3: Time (seconds) and numbers of evaluations required by different methods for solving problem **L2**. PR-CG: Polak-Ribière conjugate gradient method; BB: Barzilai-Borwein method; ABB: adaptive Barzilai-Borwein method.

		PR-CG		BB		ABB	
n	Problem	Time	#ls-#f-#g	Time	#g	Time	#g
100^3	L2(a)	104.7	262-530-280	41.2	601	25.8	380
	L2(b)	97.7	227-450-229	32.8	412	29.3	358
110^3	L2(a)	195.1	365-740-384	35.2	389	32.2	345
	L2(b)	145.2	250-496-257	46.3	439	34.8	303
120^3	L2(a)	244.1	350-712-372	57.1	473	48.3	399
	L2(b)	204.9	273-539-277	69.8	508	44.5	317
130^3	L2(a)	291.9	328-666-350	81.2	536	66.8	439
	L2(b)	281.1	295-583-300	88.7	502	75.2	421
140^3	L2(a)	505.5	449-913-475	132.8	695	127.9	675
	L2(b)	378.4	319-630-323	104.3	468	89.9	400
150^3	L2(a)	531.0	390-793-413	128.4	556	106.0	441
	L2(b)	500.9	341-677-348	122.3	444	118.3	428
160^3	L2(a)	667.6	404-821-430	167.2	593	143.9	512
	L2(b)	650.2	365-724-372	180.6	550	197.1	592
170^3	L2(a)	845.4	431-876-457	311.1	921	308.1	911
	L2(b)	825.8	387-772-398	287.5	734	252.0	631
180^3	L2(a)	771.2	323-675-362	418.4	1031	334.0	833
	L2(b)	1069.1	422-843-434	316.2	677	240.9	509

step-sizes, namely, ASD and ABB. The ASD method combines the SD and MG methods by avoiding their drawbacks at the same time, while the ABB method uses a trust-region-like strategy to choose its step-size from the two alternatives of the original BB method.

Based on our numerical experiments, we conclude that the ASD and ABB methods are comparable and in general preferable to the BB, AS and AM methods. Particularly, the ABB method seems to be a good option if the coefficient matrix is very ill-conditioned and a high precision is required. We also report that the new algorithms outperform the linear CG method when a low precision is required; and for a specific non-quadratic function, the ABB method clearly surpasses the PR-CG method as well as the unmodified BB method.

In view of the remarkable performance of the ASD method and the fact that it is a monotone algorithm, we should expect that this method would be useful in some circumstances. For example, if the dimension of the problem is very large and hence only a few iterations can be carried out, then the ASD method might be a very good option. The ABB method is not a monotone algorithm but it also

has an obvious advantage. This method itself requires no line searches for general functions and therefore might be able to save a lot of computation work while solving unconstrained optimization problems. To ensure the global convergence, one could combine the ABB method with the non-monotone line search of Grippo *et al.* [15] or a new type of non-monotone line search recently proposed by Zhang and Hager [25].

Finally, note that in our experiments, the parameter κ is usually chosen as 0.5. However, extensive numerical experiments show that taking a different κ sometimes may lead to better results. According to our experience, it is better to set κ slightly bigger than 0.5 in the ASD method, and smaller than 0.5 in the ABB method. In fact, the choice of κ is a tradeoff between a large step-size and a small one. We would like to choose the large step-size to give a substantial reduction, but at the same time, we might also need the small one to induce a favorable descent direction for the next iteration. A suitable κ, therefore, should ensure that both the large and small step-sizes will be frequently adopted.

Acknowledgements

The authors wish to thank two anonymous referees for their valuable comments and suggestions. This work was supported by the Chinese NSF grant 10171104. The third author was also supported by the Alexander-von-Humboldt Foundation.

References

[1] H. Akaike, "On a successive transformation of probability distribution and its application to the analysis of the optimum gradient method," Ann. Inst. Statist. Math. Tokyo, vol. 11, pp. 1–17, 1959.

[2] R. Andreani, E.G. Birgin, J.M. Martínez, and J. Yuan, "Spectral projected gradient and variable metric methods for optimization with linear inequalities," IMA J. Numer. Anal., vol. 25, pp. 221–252, 2005.

[3] J. Barzilai and J.M. Borwein, "Two-point step size gradient methods," IMA J. Numer. Anal., vol. 8, pp. 141–148, 1988.

[4] L. Bello and M. Raydan, "Preconditioned spectral projected gradient methods on convex sets," Journal of Computational Mathematics, vol. 23, pp. 225–232, 2005.

[5] E.G. Birgin, J.M. Martínez, and M. Raydan, "Nonmonotone spectral projected gradient methods on convex sets," SIAM J. Optim., vol. 10, pp. 1196–1211, 2000.

[6] E.G. Birgin, J.M. Martínez, and M. Raydan, "Algorithm 813: SPG—software for convex-constrained optimization," ACM Trans. Math. Software, vol. 27, pp. 340–349, 2001.

[7] E.G. Birgin, J.M. Martínez, and M. Raydan, "Inexact spectral projected gradient methods on convex sets," IMA J. Numer. Anal., vol. 23, pp. 539–559, 2003.

[8] A. Cauchy, "Méthode générale pour la résolution des systèms d'équations simultanées," Comp. Rend. Sci. Paris, vol. 25, pp. 536–538, 1847.

[9] Y.H. Dai, "Alternate step gradient method," Optimization, vol. 52, pp. 395–415, 2003.

[10] Y.H. Dai and L.Z. Liao, "R-linear convergence of the Barzilai and Borwein gradient method," IMA J. Numer. Anal., vol. 22, pp. 1–10, 2002.

[11] Y.H. Dai and Y. Yuan, "Alternate minimization gradient method," IMA J. Numer. Anal., vol. 23, pp. 377–393, 2003.

[12] R. Fletcher, "Low storage methods for unconstrained optimization," Lectures in Applied Mathematics (AMS), vol. 26, pp. 165–179, 1990.

[13] R. Fletcher, "On the Barzilai-Borwein method," Numerical Analysis Report NA/207, Department of Mathematics, University of Dundee, 2001.

[14] A. Friedlander, J.M. Martínez, B. Molina, and M. Raydan, "Gradient method with retards and generalizations," SIAM J. Numer. Anal., vol. 36, pp. 275–289, 1999.

[15] L. Grippo, F. Lampariello, and S. Lucidi, "A nonmonotone line search technique for Newton's method," SIAM J. Numer. Anal., vol. 23, pp. 707–716, 1986.

[16] M.R. Hestenes and E.L. Stiefel, "Methods of conjugate gradients for solving linear systems," J. Research National Bureau of Standards, vol. B49, pp. 409–436, 1952.

[17] W. La Cruz, J.M. Martínez, and M. Raydan, "Spectral residual method without gradient information for solving large-scale nonlinear systems of equations," Mathematics of Computation, to appear.

[18] W. La Cruz and M. Raydan, "Nonmonotone spectral methods for large-scale nonlinear systems," Optimization Methods and Software, vol. 18, pp. 583–599, 2003.

[19] J.-L. Lamotte, B. Molina, and M. Raydan, "Smooth and adaptive gradient method with retards," Mathematical and Computer Modelling, vol. 36, pp. 1161–1168, 2002.

[20] F. Luengo and M. Raydan, "Gradient method with dynamical retards for large-scale optimization problems," Electronic Transactions on Numerical Analysis, vol. 16, pp. 186–193, 2003.

[21] M. Raydan, "On the Barzilai and Borwein choice of steplength for the gradient method," IMA J. Numer. Anal., vol. 13, pp. 321–326, 1993.

[22] M. Raydan, "The Barzilai and Borwein method for the large scale unconstrained minimization problem," SIAM J. Optim., vol. 7, pp. 26–33, 1997.

[23] M. Raydan and B.F. Svaiter, "Relaxed steepest descent and Cauchy-Barzilai-Borwein method," Computational Optimization and Applications, vol. 21, pp. 155–167, 2002.

[24] T. Serafini, G. Zanghirati, and L. Zanni, "Gradient projection methods for quadratic programs and applications in training support vector machines," Tech. Rep. 48, University of Modena and Reggio Emilia, Italy, 2003.

[25] H. Zhang and W.W. Hager, "A nonmonotone line search technique and its application to unconstrained optimization," SIAM J. Optim., vol. 14, pp. 1043–1056, 2004.