The Cyclic Barzilai-Borwein Method for Unconstrained Optimization*

Yu-Hong Dai[†] William W. Hager [‡]
Klaus Schittkowski [§] Hongchao Zhang [¶]

May 31, 2005

Abstract

In the cyclic Barzilai-Borwein (CBB) method, the same BB stepsize is reused for m consecutive iterations. It is proved that CBB is locally linearly convergent at a local minimizer with positive definite Hessian. Numerical evidence indicates that when $m > n/2 \geq 3$, CBB is locally superlinearly convergent. In the special case $m = 3$ and $n = 2$, it is proved that the convergence rate is no better than linear, in general. An implementation of the CBB method, called adaptive CBB (ACBB), combines a nonmonotone line search and an adaptive choice for the cycle length m. In numerical experiments using the CUTEr [8] test problem library, ACBB performs better than an existing BB gradient algorithm, while it is competitive with the well-known PRP+ conjugate gradient algorithm.

Key words: unconstrained optimization, gradient method, convex quadratic programming, nonmonotone line search.

AMS(MOS) subject classifications. 90C30.

*First author supported by the Alexander von Humboldt Foundation under grant CHN/1112740 STP and Chinese National Science Foundation grants 10171104 and 40233029. Second and fourth authors supported by U. S. National Science Foundation Grant No. 0203270.

[†]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, PO Box 2719, Beijing 100080, PR China. Email: dyh@lsec.cc.ac.cn

[‡]Department of Mathematics, University of Florida, Gainesville, FL 32611, USA. Email: hager@math.ufl.edu

[§]Department of Computer Science, University of Bayreuth, 95440 Bayreuth, Germany. Email: klaus.schittkowski@uni-bayreuth.de

[¶]Department of Mathematics, University of Florida, Gainesville, FL 32611, USA. Email: hzhang@math.ufl.edu

1 Introduction

In this paper, we develop a cyclic Barzilai-Borwein (BB) gradient type method for solving an unconstrained optimization problem

$$\min f(x), \quad x \in \Re^n, \tag{1.1}$$

where f is continuously differentiable. Gradient methods start from an initial point x_0 and generate new iterates by the rule

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

$k \geq 0$, where $g_k = \nabla f(x_k)$ is the gradient, viewed as a column vector, and α_k is a stepsize computed by some line search algorithm.

In the steepest descent (SD) method, which can be traced back to Cauchy [7], the "exact steplength" is given by

$$\alpha_k \in \arg\min_{\alpha \in \Re} f(x_k - \alpha g_k). \tag{1.3}$$

It is well-known that steepest descent can be very slow when the Hessian of f is ill-conditioned at a local minimum (see Akaike [1] and Forsythe [22]). In this case, the iterates slowly approach the minimum in a zigzag fashion. On the other hand, it has been shown that if the exact steepest descent step is reused in a cyclic fashion, then the convergence is accelerated. Given an integer $m \geq 1$, which we call the cycle length, cyclic steepest descent can be expressed:

$$\alpha_{mk+i} = \alpha_{mk+1}^{SD} \quad \text{for } i = 1, \ldots, m, \tag{1.4}$$

$k = 0, 1, \ldots$, where α_k^{SD} is the exact steplength given by (1.3). Formula (1.4) is first proposed in [23], while the particular choice $m = 2$ is also investigated in [10] and [36]. The analysis in [11] shows that if $m > \frac{n}{2}$, cyclic steepest descent is likely R-superlinearly convergent. Hence, steepest descent is accelerated when the stepsize is repeated.

In this paper, we develop a cyclic BB method. The basic idea of Barzilai and Borwein [2] is to regard the matrix $D(\alpha_k) = \frac{1}{\alpha_k} I$ as an approximation of the Hessian $\nabla^2 f(x_k)$ and impose a quasi-Newton property on $D(\alpha_k)$:

$$\alpha_k = \arg\min_{\alpha \in \Re} \|D(\alpha) s_{k-1} - y_{k-1}\|_2, \tag{1.5}$$

where $s_{k-1} = x_k - x_{k-1}$, $y_{k-1} = g_k - g_{k-1}$, and $k \geq 2$. The proposed stepsize, obtained from (1.5), is

$$\alpha_k^{BB} = \frac{s_{k-1}^\mathsf{T} s_{k-1}}{s_{k-1}^\mathsf{T} y_{k-1}}. \tag{1.6}$$

Other possible choices for the stepsize α_k include [10, 13, 16, 17, 23, 27, 36, 37]. In this paper, we refer to (1.6) as the Barzilai-Borwein (BB) formula. The gradient method (1.2) corresponding to the BB stepsize (1.6) is called the BB method.

Due to their simplicity, efficiency, and low memory requirements, BB-like methods have been used in many applications. Glunt, Hayden, and Raydan [25] present a direct application of the BB method in chemistry. Birgin *et al.* [3] use a globalized BB method to estimate the optical constants and the thickness of thin films, while in Birgin *et al.* [5] further extensions are given, leading to more efficient projected gradient methods. Liu and Dai [31] provide a powerful scheme for solving noisy unconstrained optimization problems by combining the BB method and a stochastic approximation method. The projected BB-like method turns out to be very useful in machine learning for training support vector machines (see Serafini *et al* [37] and Dai and Fletcher [13]). Empirically, good performance is observed on a wide variety of classification problems.

The superior performance of cyclic steepest descent, compared to the ordinary steepest descent, as shown in [11], leads us to consider the cyclic BB method (CBB):

$$\alpha_{mk+i} = \alpha_{mk+1}^{BB} \quad \text{for } i = 1, \ldots, m, \tag{1.7}$$

where $m \geq 1$ is again the cycle length. An advantage of the CBB method is that for general nonlinear functions, the stepsize is given by the simple formula (1.5) in contrast to the nontrivial optimization problem associated with the steepest descent step (1.3).

In [23] it is shown that when f is a strongly convex quadratic, CBB is at least linearly convergent. In Section 2 we prove local R-linear convergence for the CBB method at a local minimizer of a general nonlinear function. In Section 3 numerical evidence for strongly convex quadratic functions indicates that the convergence is superlinear if $m > n/2 \geq 3$. In the special case $m = 3$ and $n = 2$, we prove that the convergence is at best linear, in general.

In Section 4 we propose an adaptive method for computing an appropriate cycle length, and we obtain a globally convergent nonmonotone scheme by using a modified version of the line search developed in [18]. This new line search, an adaptive analogue of Toint's scheme [38] for trust region methods, accepts the original BB stepsize more often than does Raydan's [34] strategy for globalizing the BB method. We refer to Raydan's globalized BB implementation as the GBB method. Numerical comparisons with the PRP+ algorithm and with the SPG2 algorithm [5] (one version of the GBB method) are given in Section 4 using the CUTEr test problem library [8].

2 Local linear convergence

In this section we prove R-linear convergence for the CBB method. In [31], it is proposed that R-linear convergence for the BB method applied to a general nonlinear function could be obtained from the R-linear convergence results for a quadratic by comparing the iterates associated with a quadratic approximation to the general nonlinear iterates. In our R-linear convergence result for the CBB method, we make such a comparison.

The CBB iteration can be expressed as

$$x_{k+1} = x_k - \alpha_k g_k, \tag{2.8}$$

where

$$\alpha_k = \frac{s_i^{\mathsf T} s_i}{s_i^{\mathsf T} y_i}, \quad i = \nu(k), \quad \text{and} \quad \nu(k) = m\lfloor (k-1)/m \rfloor, \tag{2.9}$$

$k \ge 1$. For $r \in \Re$, $\lfloor r \rfloor$ denotes the largest integer j such that $j \le r$. We assume that f is two times Lipschitz continuously differentiable in a neighborhood of a local minimizer x^* where the Hessian $H = \nabla^2 f(x^*)$ is positive definite. The second-order Taylor approximation $\hat{f}$ to f around x^* is given by

$$\hat{f}(x) = f(x^*) + \frac{1}{2}(x - x^*)^{\mathsf T} H (x - x^*). \tag{2.10}$$

We will compare an iterate x_{k+j} generated by (2.8) to a CBB iterate $\hat{x}_{k,j}$ associated with $\hat{f}$ and the starting point $\hat{x}_{k,0} = x_k$. More precisely, we define:

$$\begin{aligned} \hat{x}_{k,0} &= x_k \\ \hat{x}_{k,j+1} &= \hat{x}_{k,j} - \hat{\alpha}_{k,j} \hat{g}_{k,j}, \quad j \ge 0, \end{aligned} \tag{2.11}$$

where

$$\hat{\alpha}_{k,j} = \begin{cases} \alpha_k & \text{if} \quad \nu(k+j) = \nu(k) \\ \dfrac{\hat{s}_i^{\mathsf T} \hat{s}_i}{\hat{s}_i^{\mathsf T} \hat{y}_i}, & i = \nu(k+j), \quad \text{otherwise.} \end{cases}$$

Here $\hat{s}_{k+j} = \hat{x}_{k,j+1} - \hat{x}_{k,j}$, $\hat{g}_{k,j} = H(\hat{x}_{k,j} - x^*)$, and $\hat{y}_{k+j} = \hat{g}_{k,j+1} - \hat{g}_{k,j}$.

We exploit the following result established in [10, Thm. 3.2]:

Lemma 1. *Let $\{\hat{x}_{k,j} : j \ge 0\}$ be the CBB iterates associated with the starting point $\hat{x}_{k,0} = x_k$ and the quadratic $\hat{f}$ in* (2.10)*, where H is positive definite. Given two arbitrary constants $C_2 > C_1$, there exists a positive integer M with the following property: For any $k \ge 1$ and*

$$\hat{\alpha}_{k,0} \in [C_1, C_2], \tag{2.12}$$

$$\|\hat{x}_{k,M} - x^*\| \le \frac{1}{2} \|\hat{x}_{k,0} - x^*\|.$$

In our next lemma, we estimate the distance between $\hat{x}_{k,j}$ and x_{k+j}. Let $B_\rho(x)$ denote the ball with center x and radius ρ. Since f is two times Lipschitz continuously differentiable and $\nabla^2 f(x^*)$ is positive definite, there exists positive constants ρ, λ, and $\Lambda_2 > \Lambda_1$ such that

$$\|\nabla f(x) - H(x - x^*)\| \le \lambda \|x - x^*\|^2 \quad \text{for all } x \in B_\rho(x^*) \tag{2.13}$$

and

$$\Lambda_1 \le \frac{y^{\mathsf T} \nabla^2 f(x) y}{y^{\mathsf T} y} \le \Lambda_2 \quad \text{for all } y \in \Re^n \text{ and } x \in B_\rho(x^*). \tag{2.14}$$

Notice that if x_i and $x_{i+1} \in B_\rho(x^*)$, then the fundamental theorem of calculus applied to $y_i = g_{i+1} - g_i$ yields

$$\frac{1}{\Lambda_2} \le \frac{s_i^\mathsf{T} s_i}{s_i^\mathsf{T} y_i} \le \frac{1}{\Lambda_1}. \tag{2.15}$$

Hence, when the CBB iterates lie in $B_\rho(x^*)$, the condition (2.12) of Lemma 1 is satisfied with $C_1 = 1/\Lambda_2$ and $C_2 = 1/\Lambda_1$. If we define $g(x) = \nabla f(x)$, then the fundamental theorem of calculus can also be used to deduce that

$$\|g(x)\| = \|g(x) - g(x^*)\| \le \Lambda_2 \|x - x^*\| \tag{2.16}$$

for all $x \in B_\rho(x^*)$.

Lemma 2. *Let $\{x_j : j \ge k\}$ be a sequence generated by the CBB method applied to a function f with a local minimizer x^*, and assume that the Hessian $H = \nabla^2 f(x^*)$ is positive definite with (2.14) satisfied. Then for any fixed positive integer M, there exist positive constants δ and γ with the following property: For any $x_k \in B_\delta(x^*)$, $\alpha_k \in [\Lambda_2^{-1}, \Lambda_1^{-1}]$, $l \in [0, M]$ with*

$$\|\hat{x}_{k,j} - x^*\| \ge \frac{1}{2}\|\hat{x}_{k,0} - x^*\| \quad \text{for all } j \in [0, l-1], \tag{2.17}$$

we have

$$x_{k+j} \in B_\rho(x^*) \text{ and } \|x_{k+j} - \hat{x}_{k,j}\| \le \gamma \|x_k - x^*\|^2 \tag{2.18}$$

for all $j \in [0, l]$.

Proof. Throughout the proof, we let c denote a generic constant, which depends on fixed constants such as M or Λ_1 or Λ_2 or λ, but not on either k or the choice of $x_k \in B_\delta(x^*)$ or the choice of $\alpha_k \in [\Lambda_2^{-1}, \Lambda_1^{-1}]$. To facilitate the proof, we also show that

$$\begin{aligned} \|g(x_{k+j}) - \hat{g}(\hat{x}_{k,j})\| &\le c\|x_k - x^*\|^2, && (2.19)\\ \|s_{k+j}\| &\le c\|x_k - x^*\|, && (2.20)\\ |\alpha_{k+j} - \hat{\alpha}_{k,j}| &\le c\|x_k - x^*\|, && (2.21) \end{aligned}$$

for all $j \in [0, l]$, where $\hat{g}(x) = \nabla \hat{f}(x) = H(x - x^*)$.

The proof of (2.18)–(2.21) is by induction on l. For $l = 0$, we take $\delta = \rho$. The relation (2.18) is trivial since $\hat{x}_{k,0} = x_k$. By (2.13), we have

$$\|g(x_k) - \hat{g}(\hat{x}_{k,0})\| = \|g(x_k) - \hat{g}(x_k)\| \le \lambda \|x_k - x^*\|^2,$$

which gives (2.19). Since $\delta = \rho$ and $x_k \in B_\delta(x^*)$, it follows from (2.16) that

$$\|s_k\| = \|\alpha_k g_k\| \le \frac{\Lambda_2}{\Lambda_1}\|x_k - x^*\|,$$

which gives (2.20). The relation (2.21) is trivial since $\hat{\alpha}_{k,0} = \alpha_k$.

Now, proceeding by induction, suppose that there exist $L \in [1, M)$ and $\delta > 0$ with the property that if (2.17) holds for any $l \in [0, L-1]$, then (2.18)–(2.21) are satisfied for all $j \in [0, l]$. We wish to show that for a smaller choice of $\delta > 0$, we can replace L by $L+1$. Hence, we suppose that (2.17) holds for all $j \in [0, L]$. Since (2.17) holds for all $j \in [0, L-1]$, it follows from the induction hypothesis and (2.20) that

$$\begin{aligned} \|x_{k+L+1} - x^*\| &\leq \|x_k - x^*\| + \sum_{i=0}^{L} \|s_{k+i}\| \\ &\leq c\|x_k - x^*\|. \end{aligned} \tag{2.22}$$

Consequently, by choosing δ smaller if necessary, we have $x_{k+L+1} \in B_\rho(x^*)$ when $x_k \in B_\delta(x^*)$.

By the triangle inequality,

$$\begin{aligned} &\|x_{k+L+1} - \hat{x}_{k,L+1}\| \\ &\quad = \|x_{k+L} - \alpha_{k+L} g(x_{k+L}) - [\hat{x}_{k,L} - \hat{\alpha}_{k,L}\hat{g}(\hat{x}_{k,L})]\| \\ &\quad \leq \|x_{k+L} - \hat{x}_{k,L}\| + |\hat{\alpha}_{k,L}|\|g(x_{k+L}) - \hat{g}(\hat{x}_{k,L})\| \\ &\qquad\quad + |\alpha_{k+L} - \hat{\alpha}_{k,L}|\|g(x_{k+L})\|. \end{aligned} \tag{2.23}$$

We now analyze each of the terms in (2.23). By the induction hypothesis, the bound (2.18) with $j = L$ holds, which gives

$$\|x_{k+L} - \hat{x}_{k,L}\| \leq c\|x_k - x^*\|^2. \tag{2.24}$$

By the definition of $\hat{\alpha}$, either $\hat{\alpha}_{k,L} = \alpha_k \in [\Lambda_2^{-1}, \Lambda_1^{-1}]$, or

$$\hat{\alpha}_{k,L} = \frac{\hat{s}_i^{\mathsf{T}} \hat{s}_i}{\hat{s}_i^{\mathsf{T}} \hat{y}_i}, \quad i = \nu(k+L).$$

In this latter case,

$$\frac{1}{\Lambda_2} \leq \frac{\hat{s}_i^{\mathsf{T}} \hat{s}_i}{\hat{s}_i^{\mathsf{T}} H \hat{s}_i} = \frac{\hat{s}_i^{\mathsf{T}} \hat{s}_i}{\hat{s}_i^{\mathsf{T}} \hat{y}_i} \leq \frac{1}{\Lambda_1}.$$

Hence, in either case $\hat{\alpha}_{k,L} \in [\Lambda_2^{-1}, \Lambda_1^{-1}]$. It follows from (2.19) with $j = L$ that

$$\begin{aligned} |\hat{\alpha}_{k,L}|\|g(x_{k+L}) - \hat{g}(\hat{x}_{k,L})\| &\leq \frac{1}{\Lambda_1}\|g(x_{k+L}) - \hat{g}(\hat{x}_{k,L})\| \\ &\leq c\|x_k - x^*\|^2. \end{aligned} \tag{2.25}$$

Also, by (2.21) with $j = L$ and (2.16), we have

$$|\alpha_{k+L} - \hat{\alpha}_{k,L}|\|g(x_{k+L})\| \leq c\|x_k - x^*\|\|x_{k+L} - x^*\|.$$

Utilizing (2.22) (with L replaced by $L-1$) gives

$$|\alpha_{k+L} - \hat{\alpha}_{k,L}|\|g(x_{k+L})\| \leq c\|x_k - x^*\|^2. \tag{2.26}$$

We combine (2.23)–(2.26) to obtain (2.18) for $j = L+1$. Notice that in establishing (2.18), we exploited (2.19)–(2.21). Consequently, to complete the induction step, each of these estimates should be proved for $j = L+1$.

Focusing on (2.19) for $j = L+1$, we have

$$\begin{aligned}
&\|g(x_{k+L+1}) - \hat{g}(\hat{x}_{k,L+1})\| \\
&\quad \leq \|g(x_{k+L+1}) - \hat{g}(x_{k+L+1})\| + \|\hat{g}(x_{k+L+1}) - \hat{g}(\hat{x}_{k,L+1})\| \\
&\quad = \|g(x_{k+L+1}) - \hat{g}(x_{k+L+1})\| + \|H(x_{k+L+1} - \hat{x}_{k,L+1})\| \\
&\quad \leq \|g(x_{k+L+1}) - H(x_{k+L+1} - x^*)\| + \Lambda_2\|x_{k+L+1} - \hat{x}_{k,L+1}\| \\
&\quad \leq \|g(x_{k+L+1}) - H(x_{k+L+1} - x^*)\| + c\|x_k - x^*\|^2,
\end{aligned}$$

since $\|H\| \leq \Lambda_2$ by (2.14). The last inequality is due to (2.18) for $j = L+1$, which was just established. Since we chose δ small enough that $x_{k+L+1} \in B_\rho(x^*)$ (see (2.22), (2.13) implies that

$$\|g(x_{k+L+1}) - H(x_{k+L+1} - x^*)\| \leq \lambda\|x_{k+L+1} - x^*\|^2 \leq c\|x_k - x^*\|^2.$$

Hence, $\|g(x_{k+L+1}) - \hat{g}(\hat{x}_{k,L+1})\| \leq c\|x_k - x^*\|^2$, which establishes (2.19) for $j = L+1$.

Observe that α_{k+L+1} either equals $\alpha_k \in [\Lambda_2^{-1}, \Lambda_1^{-1}]$, or $(s_i^\mathsf{T} s_i)/(s_i^\mathsf{T} y_i)$, where $k + L \geq i = \nu(k+L+1) > k$. In this latter case, since $x_{k+j} \in B_\rho(x^*)$ for $0 \leq j \leq L+1$, it follows from (2.15) that

$$\alpha_{k+L+1} \leq \frac{1}{\Lambda_1}.$$

Combining this with (2.16) and the bound (2.20) for $j \leq L$, we obtain

$$\begin{aligned}
\|s_{k+L+1}\| &= \|\alpha_{k+L+1} g(x_{k+L+1})\| \leq \frac{\Lambda_2}{\Lambda_1}\|x_{k+L+1} - x^*\| \\
&\leq \frac{\Lambda_2}{\Lambda_1}\left(\|x_k - x^*\| + \sum_{j=0}^{L} \|s_{k+j}\|\right) \\
&\leq c\|x_k - x^*\|.
\end{aligned}$$

Hence, (2.20) is established for $j = L+1$.

Finally, we focus on (2.21) for $j = L+1$. If $\nu(k+L+1) = \nu(k)$, then $\hat{\alpha}_{k,L+1} = \alpha_{k+L+1} = \alpha_k$, so we are done. Otherwise, $\nu(k+L+1) > \nu(k)$, and there exists an index $i \in (0, L]$ such that

$$\alpha_{k+L+1} = \frac{s_{k+i}^\mathsf{T} s_{k+i}}{s_{k+i}^\mathsf{T} y_{k+i}} \quad \text{and} \quad \hat{\alpha}_{k,L+1} = \frac{\hat{s}_{k+i}^\mathsf{T} \hat{s}_{k+i}}{\hat{s}_{k+i}^\mathsf{T} \hat{y}_{k+i}}.$$

By (2.18) and the fact that $i \leq L$, we have

$$\|s_{k+i} - \hat{s}_{k+i}\| \leq c\|x_k - x^*\|^2.$$

Combining this with (2.20) gives

$$|s_{k+i}^{\mathsf{T}} s_{k+i} - \hat{s}_{k+i}^{\mathsf{T}} \hat{s}_{k+i}| = \left| 2 s_{k+i}^{\mathsf{T}} (s_{k+i} - \hat{s}_{k+i}) - \|\hat{s}_{k+i} - s_{k+i}\|^2 \right| \leq c \|x_k - x^*\|^3. \quad (2.27)$$

Since $\hat{\alpha}_{k,i} \in [\Lambda_2^{-1}, \Lambda_1^{-1}]$, we have

$$\begin{aligned} \|\hat{s}_{k+i}\| &= \|\hat{\alpha}_{k,i} \hat{g}_{k,i}\| \geq \frac{1}{\Lambda_2} \|H(\hat{x}_{k,i} - x^*)\| \\ &\geq \frac{\Lambda_1}{\Lambda_2} \|\hat{x}_{k,i} - x^*\|. \end{aligned}$$

Furthermore, by (2.17) it follows that

$$\|\hat{s}_{k+i}\| \geq \frac{\Lambda_1}{2\Lambda_2} \|\hat{x}_{k,0} - x^*\| = \frac{\Lambda_1}{2\Lambda_2} \|x_k - x^*\|. \quad (2.28)$$

Hence, combining (2.27) and (2.28) gives

$$\left| 1 - \frac{s_{k+i}^{\mathsf{T}} s_{k+i}}{\hat{s}_{k+i}^{\mathsf{T}} \hat{s}_{k+i}} \right| = \frac{|s_{k+i}^{\mathsf{T}} s_{k+i} - \hat{s}_{k+i}^{\mathsf{T}} \hat{s}_{k+i}|}{\hat{s}_{k+i}^{\mathsf{T}} \hat{s}_{k+i}} \leq c \|x_k - x^*\|. \quad (2.29)$$

Now let us consider the denominators of α_{k+i} and $\hat{\alpha}_{k,i}$. Observe that

$$\begin{aligned} s_{k+i}^{\mathsf{T}} y_{k+i} - \hat{s}_{k+i}^{\mathsf{T}} \hat{y}_{k+i} &= s_{k+i}^{\mathsf{T}} (y_{k+i} - \hat{y}_{k+i}) + (s_{k+i} - \hat{s}_{k+i})^{\mathsf{T}} \hat{y}_{k+i} \\ &= s_{k+i}^{\mathsf{T}} (y_{k+i} - \hat{y}_{k+i}) + (s_{k+i} - \hat{s}_{k+i})^{\mathsf{T}} H \hat{s}_{k+i}. \quad (2.30) \end{aligned}$$

By (2.18) and (2.20), we have

$$\begin{aligned} |(s_{k+i} - \hat{s}_{k+i})^{\mathsf{T}} H \hat{s}_{k+i}| &= |(s_{k+i} - \hat{s}_{k+i})^{\mathsf{T}} H s_{k+i} - (s_{k+i} - \hat{s}_{k+i})^{\mathsf{T}} H (s_{k+i} - \hat{s}_{k+i})| \\ &\leq c \|x_k - x^*\|^3. \quad (2.31) \end{aligned}$$

Also, by (2.19) and (2.20), we have

$$|s_{k+i}^{\mathsf{T}} (y_{k+i} - \hat{y}_{k+i})| \leq \|s_{k+i}\| (\|g_{k+i+1} - \hat{g}_{k,i+1}\| + \|g_{k+i} - \hat{g}_{k,i}\|) \leq c \|x_k - x^*\|^3. \quad (2.32)$$

Combining (2.30)–(2.32) yields

$$s_{k+i}^{\mathsf{T}} y_{k+i} - \hat{s}_{k+i}^{\mathsf{T}} \hat{y}_{k+i} \leq c \|x_k - x^*\|^3. \quad (2.33)$$

Since x_{k+i} and $x_{k+i+1} \in B_\rho(x^*)$, it follows from (2.14) that

$$s_{k+i}^{\mathsf{T}} y_{k+i} = s_{k+i}^{\mathsf{T}} (g_{k+i+1} - g_{k+i}) \geq \Lambda_1 \|s_{k+i}\|^2 = \Lambda_1 |\alpha_{k+i}|^2 \|g_{k+i}\|^2. \quad (2.34)$$

By (2.15) and (2.14), we have

$$|\alpha_{k+i}|^2 \|g_{k+i}\|^2 \geq \frac{1}{\Lambda_2^2} \|g_{k+i}\|^2 = \frac{1}{\Lambda_2^2} \|g(x_{k+i}) - g(x^*)\|^2 \geq \frac{\Lambda_1^2}{\Lambda_2^2} \|x_{k+i} - x^*\|^2. \quad (2.35)$$

Finally, (2.17) gives

$$\|x_{k+i} - x^*\|^2 \geq \frac{1}{4}\|x_k - x^*\|^2. \tag{2.36}$$

Combining (2.34)–(2.36) yields

$$s_{k+i}^\mathsf{T} y_{k+i} \geq \frac{\Lambda_1^3}{4\Lambda_2^2}\|x_k - x^*\|^2. \tag{2.37}$$

Combining (2.33) and (2.37) gives

$$\left|1 - \frac{\hat{s}_{k+i}^\mathsf{T}\hat{y}_{k+i}}{s_{k+i}^\mathsf{T} y_{k+i}}\right| = \frac{|s_{k+i}^\mathsf{T} y_{k+i} - \hat{s}_{k+i}^\mathsf{T}\hat{y}_{k+i}|}{s_{k+i}^\mathsf{T} y_{k+i}} \leq c\|x_k - x^*\|. \tag{2.38}$$

Together, (2.29) and (2.38) yield

$$\begin{aligned}
|\alpha_{k+L+1} - \hat{\alpha}_{k,L+1}| &= \left|\frac{s_{k+i}^\mathsf{T} s_{k+i}}{s_{k+i}^\mathsf{T} y_{k+i}} - \frac{\hat{s}_{k+i}^\mathsf{T}\hat{s}_{k+i}}{\hat{s}_{k+i}^\mathsf{T}\hat{y}_{k+i}}\right| \\
&= \hat{\alpha}_{k,L+1}\left|1 - \left(\frac{s_{k+i}^\mathsf{T} s_{k+i}}{\hat{s}_{k+i}^\mathsf{T}\hat{s}_{k+i}}\right)\left(\frac{\hat{s}_{k+i}^\mathsf{T}\hat{y}_{k+i}}{s_{k+i}^\mathsf{T} y_{k+i}}\right)\right| \\
&\leq \frac{1}{\Lambda_1}\left|1 - \left(\frac{s_{k+i}^\mathsf{T} s_{k+i}}{\hat{s}_{k+i}^\mathsf{T}\hat{s}_{k+i}}\right)\left(\frac{\hat{s}_{k+i}^\mathsf{T}\hat{y}_{k+i}}{s_{k+i}^\mathsf{T} y_{k+i}}\right)\right| \\
&\leq c\|x_k - x^*\|.
\end{aligned}$$

This completes the proof of (2.18)–(2.21). □

Theorem 1. *Let x^* be a local minimizer of f, and assume that the Hessian $\nabla^2 f(x^*)$ is positive definite. Then there exist positive constants δ and γ, and a constant $c < 1$ with the property that for all starting points $x_0, x_1 \in B_\delta(x^*)$, $x_0 \neq x_1$, the CBB iterates generated by* (2.8)–(2.9) *satisfy*

$$\|x_k - x^*\| \leq \gamma c^k \|x_1 - x^*\|.$$

Proof. Let $M > 0$ be the integer given in Lemma 1, corresponding to $C_1 = \Lambda_1^{-1}$ and $C_2 = \Lambda_2^{-1}$, and let δ_1 and γ_1 denote the constants δ and γ given in Lemma 2. Let γ_2 denote the constant c in (2.20). In other words, these constant δ_1, γ_1, and and γ_2 have the property that whenever $\|x_k - x^*\| \leq \delta_1$, $\alpha_k \in [\Lambda_2^{-1}, \Lambda_1^{-1}]$, and

$$\|\hat{x}_{k,j} - x^*\| \geq \frac{1}{2}\|\hat{x}_{k,0} - x^*\| \quad \text{for } 0 \leq j \leq l - 1 < M,$$

we have

$$\begin{aligned}
\|x_{k+j} - \hat{x}_{k,j}\| &\leq \gamma_1\|x_k - x^*\|^2, & (2.39)\\
\|s_{k+j}\| &\leq \gamma_2\|x_k - x^*\|, & (2.40)\\
x_{k+j} &\in B_\rho(x^*), & (2.41)
\end{aligned}$$

for all $j \in [0, l]$. Moreover, by the triangle inequality and (2.40), it follows that

$$\begin{aligned} \|x_{k+j} - x^*\| &\leq (M\gamma_2 + 1)\|x_k - x^*\| \\ &= \gamma_3\|x_k - x^*\|, \quad \gamma_3 = (M\gamma_2 + 1), \end{aligned} \tag{2.42}$$

for all $j \in [0, l]$. We define

$$\delta = \min\{\delta_1, \rho, (4\gamma_1)^{-1}\}. \tag{2.43}$$

For any x_0 and $x_1 \in B_\delta(x^*)$, we define a sequence $1 = k_1 < k_2 < \ldots$ in the following way: Starting with the index $k_1 = 1$, let $j_1 > 0$ be the smallest integer with the property that

$$\|\hat{x}_{k_1,j_1} - x^*\| \leq \frac{1}{2}\|\hat{x}_{k_1,0} - x^*\| = \frac{1}{2}\|x_1 - x^*\|.$$

Since x_0 and $x_1 \in B_\delta(x^*) \subset B_\rho(x^*)$, it follows from (2.15) that

$$\hat{\alpha}_{1,0} = \alpha_1 = \frac{s_0^\mathsf{T} s_0}{s_0^\mathsf{T} y_0} \in [\Lambda_2^{-1}, \Lambda_1^{-1}].$$

By Lemma 1, $j_1 \leq M$. Define $k_2 = k_1 + j_1 > k_1$. By (2.39) and (2.43), we have

$$\begin{aligned} \|x_{k_2} - x^*\| &= \|x_{k_1+j_1} - x^*\| \leq \|x_{k_1+j_1} - \hat{x}_{k_1,j_1}\| + \|\hat{x}_{k_1,j_1} - x^*\| \\ &\leq \gamma_1\|x_{k_1} - x^*\|^2 + \frac{1}{2}\|\hat{x}_{k_1,0} - x^*\| \\ &= \gamma_1\|x_{k_1} - x^*\|^2 + \frac{1}{2}\|x_{k_1} - x^*\| \\ &\leq \frac{3}{4}\|x_{k_1} - x^*\|. \end{aligned} \tag{2.44}$$

Since $\|x_1 - x^*\| \leq \delta$, it follows that $x_{k_2} \in B_\delta(x^*)$. By (2.41), $x_j \in B_\rho(x^*)$ for $1 \leq j \leq k_1$.

Now, proceed by induction. Assume that k_i has been determined with $x_{k_i} \in B_\delta(x^*)$ and $x_j \in B_\rho(x^*)$ for $1 \leq j \leq k_i$. Let $j_i > 0$ be the smallest integer with the property that

$$\|\hat{x}_{k_i,j_i} - x^*\| \leq \frac{1}{2}\|\hat{x}_{k_i,0} - x^*\| = \frac{1}{2}\|x_{k_i} - x^*\|.$$

Set $k_{i+1} = k_i + j_i > k_i$. Exactly as in (2.44), we have

$$\|x_{k_{i+1}} - x^*\| \leq \frac{3}{4}\|x_{k_i} - x^*\|.$$

Again, $x_{k_{i+1}} \in B_\delta(x^*)$ and $x_j \in \mathcal{B}_\rho(x^*)$ for $j \in [1, k_{i+1}]$.

For any $j \in [k_i, k_{i+1})$, we have $j \leq k_i + M - 1 \leq Mi$, since $k_i \leq M(i-1) + 1$. Hence, $i \geq j/M$. Also, (2.42) gives

$$\|x_j - x^*\| \leq \gamma_3\|x_{k_i} - x^*\|$$

$$
\begin{aligned}
&\leq \gamma_3 \left(\frac{3}{4}\right)^{i-1} \|x_{k_1} - x^*\| \\
&\leq \gamma_3 \left(\frac{3}{4}\right)^{(j/M)-1} \|x_1 - x^*\| \\
&= \gamma c^j \|x_1 - x^*\|,
\end{aligned}
$$

where

$$\gamma = \left(\frac{4}{3}\right)\gamma_3 \quad \text{and} \quad c = \left(\frac{3}{4}\right)^{1/M} < 1.$$

This completes the proof. □

3 The CBB method for convex quadratic programming

In this section, we give numerical evidence which indicates that when m is sufficiently large, the CBB method is superlinearly convergent for a quadratic function

$$f(x) = \frac{1}{2}x^{\mathsf{T}}Ax - b^{\mathsf{T}}x, \tag{3.45}$$

where $A \in \Re^{n\times n}$ is symmetric, positive definite and $b \in \Re^n$. Since CBB is invariant under an orthogonal transformation and since gradient components corresponding to identical eigenvalues can be combined (see for example Dai and Fletcher [11]), we assume without loss of generality that A is diagonal:

$$A = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n) \quad \text{with } 0 < \lambda_1 < \lambda_2 < \cdots < \lambda_n. \tag{3.46}$$

In the following subsections, we give an overview of the experimental convergence results; we then show in the special case $m = 2$ and $n = 3$ that the convergence rate is no better than linear, in general. Finally, we show that the convergence rate for CBB is strictly faster than that of steepest descent. We obtain some further sights by applying our techniques to cyclic steepest descent.

3.1. Asymptotic behavior and cycle number

In the quadratic case, it follows from (1.2) and (3.45) that

$$g_{k+1} = (I - \alpha_k A)g_k. \tag{3.47}$$

If $g_k^{(i)}$ denotes the i-th component of the gradient g_k, then by (3.47) and (3.46), we have

$$g_{k+1}^{(i)} = (1 - \alpha_k\lambda_i)g_k^{(i)} \qquad i = 1, 2, \ldots, n. \tag{3.48}$$

We assume that $g_k^{(i)} \neq 0$ for all sufficiently large k. If $g_k^{(i)} = 0$, then by (3.48) component i remains zero during all subsequent iterations; hence it can be discarded. In the BB method, starting values are needed for x_0 and x_1 in order to compute α_1.

In our study of CBB, we treat α_1 as a free parameter. In our numerical experiments, α_1 is the exact stepsize (1.3).

For different choices of the diagonal matrix (3.46) and the starting point x_1, we have evaluated the convergence rate of CBB. By the analysis given in [23] for positive definite quadratics, or by the result given in Theorem 1 for general nonlinear functions, the convergence rate of the iterates is at least linear. On the other hand, for m sufficiently large, we observe experimentally, that the convergence rate is superlinear. The largest value of m for which the convergence rate is linear is shown in Table 1. For $m >$ the integer given in the first row of the Table 1, the convergence rate is superlinear, while for $m \leq$ the value appearing in the table, the convergence is linear.

The limiting integers appearing in Table 1 are computed in the following way: For each dimension, we randomly generate 30 problems, with eigenvalues uniformly distributed on $[0, n]$, and 50 starting points – a total of 1500 problems. For each test problem, we perform $1000n$ CBB iterations, and we plot $\log(\log(\|g_k\|_\infty))$ versus the iteration number. We fit the data with a least squares line, and we compute the correlation coefficient to determine how well the linear regression model fits the data. If the correlation coefficient is 1 (or -1), then the linear fit is perfect, while a correlation coefficient of 0 means that the data is uncorrelated. A linear fit in a plot of $\log(\log(\|g_k\|_\infty))$ versus the iteration number indicates superlinear convergence. For m large enough, the correlation coefficients are between -1.0 and $-.98$, indicating superlinear convergence. As we decrease m, the correlation coefficient abruptly jumps to the order of $-.8$. The integers shown in Table 1 are the values of m where the correlation coefficient jumps. For strictly larger m, the convergence is superlinear.

m	1	2	3	4	5	6	7
n	4	3	5	8	10	12	14

Table 1. Transition to superlinear convergence

Referring to the $m = 1$ column, which corresponds to the BB method, the convergence rate is superlinear for $n = 1, 2$, and 3; the convergence rate is linear for $n = 4, 5, \ldots$. For $m = 2$, the convergence is linear for $n \geq 3$, while the convergence is superlinear for $n = 1$ or $n = 2$. Based on Table 1, the convergence rate is conjectured to be superlinear for $m > n/2 \geq 3$. For $n < 8$, the relationship between m and n at the transition between linear and superlinear convergence is more complicated. Graphs illustrating the convergence appear in Figure 1. The horizontal axis in these figures is the iteration number, while the vertical axis gives $\log(\log(\|g_k\|_\infty))$. Here $\|\cdot\|_\infty$ represents the sup-norm. In this case, straight lines correspond to superlinear convergence – the slope of the line reflects the convergence order. In Figure 1, the bottom two graphs correspond to superlinear convergence, while the top two graphs correspond to linear convergence – for these top two examples, a plot of $\log(\|g_k\|_\infty)$ versus the iteration number is linear.

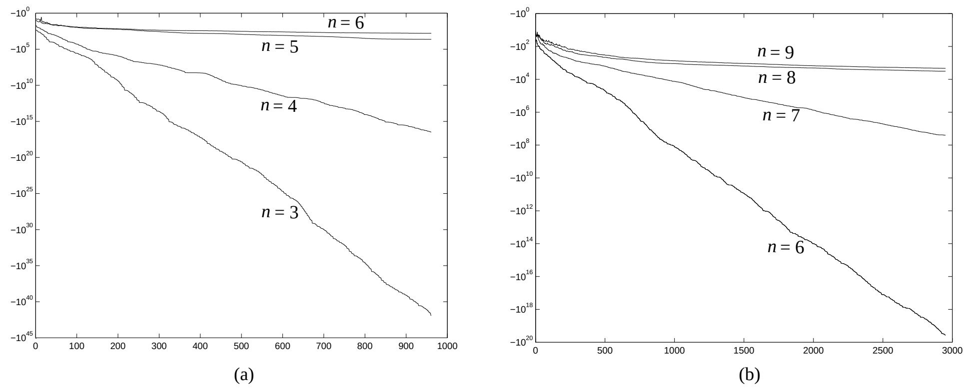

Figure 1: Graphs of $\log(\log(\|g_k\|_\infty))$ versus k, (a) $3 \le n \le 6$ and $m = 3$, (b) $6 \le n \le 9$ and $m = 4$.

3.2. Analysis for the case $m = 2$ and $n = 3$

The theoretical verification of the experimental results given in Table 1 is not easy. We have the following partial result in connection with the column $m = 2$.

Theorem 2. *For $n = 3$, there exists a choice for the diagonal matrix* (3.46) *and a starting guess x_1 with the property that $\alpha_{k+8} = \alpha_k$ for each k, and the convergence rate of CBB with $m = 2$ is at most linear.*

Proof. To begin, we treat the initial stepsize α_1 as a variable. For each k, we define the vector u_k by

$$u_k^{(i)} = \frac{(g_k^{(i)})^2}{\|g_k\|^2}, \qquad i = 1, \ldots, n. \tag{3.49}$$

The above definition is important and is used for some other gradient methods, see [22, 16]. For the case $m = 2$, we can obtain by (3.48), (2.8), (2.9) and the definition of u_k that

$$u_{2k+1}^{(i)} = \frac{(1 - \alpha_{2k-1}\lambda_i)^4 u_{2k-1}^{(i)}}{\sum_{l=1}^{n} (1 - \alpha_{2k-1}\lambda_l)^4 u_{2k-1}^{(l)}} \tag{3.50}$$

for all $k \ge 1$ and $i = 1, \ldots, n$. In the same fashion, we have

$$\alpha_{2k+1} = \frac{\sum_{l=1}^{n} (1 - \alpha_{2k-1}\lambda_l)^2 u_{2k-1}^{(l)}}{\sum_{l=1}^{n} \lambda_l (1 - \alpha_{2k-1}\lambda_l)^2 u_{2k-1}^{(l)}}. \tag{3.51}$$

We want to force our examples to satisfy

$$u_9 = u_1 \quad \text{and} \quad \alpha_9 = \alpha_1. \tag{3.52}$$

For $k \geq 1$, a subsequent iteration of the method is uniquely determined by u_{2k-1} and α_{2k-1}. It follows from (3.52) that $u_{8k+1} = u_1$ and $\alpha_{8k+1} = \alpha_1$ for all $k \geq 1$, and hence a cycle occurs.

For any i and j, let b_{ij} be defined by

$$b_{ij} = 1 - \alpha_{2i-1}\lambda_j. \tag{3.53}$$

Henceforth, we focus on the case $n = 3$ specified in the statement of the Theorem 2. To satisfy the relation (3.52), we impose the following condition on the stepsizes $\{\alpha_1, \alpha_3, \alpha_5, \alpha_7\}$,

$$\left| \prod_{i=1}^{4} b_{ij} \right| = \tau, \quad j = 1, 2, 3, \tag{3.54}$$

where $\tau > 0$ is a positive number. By (3.54) and (3.50), we know that the first equation of (3.52) is satisfied. On the other hand, (3.50), (3.51), $\alpha_9 = \alpha_1$, and the definition of (3.53) imply the following system of linear equations for u_1,

$$T\,u_1 = \begin{bmatrix} b_{11}^2 b_{21} & b_{12}^2 b_{22} & b_{13}^2 b_{23} \\ b_{11}^4 b_{21}^2 b_{31} & b_{12}^4 b_{22}^2 b_{32} & b_{13}^4 b_{23}^2 b_{33} \\ b_{11}^4 b_{21}^4 b_{31}^2 b_{41} & b_{12}^4 b_{22}^4 b_{32}^2 b_{42} & b_{13}^4 b_{23}^4 b_{33}^2 b_{43} \\ b_{11}^5 b_{21}^4 b_{31}^4 b_{41}^2 & b_{12}^5 b_{22}^4 b_{32}^4 b_{42}^2 & b_{13}^5 b_{23}^4 b_{33}^4 b_{43}^2 \end{bmatrix} \begin{bmatrix} u_1^{(1)} \\ u_1^{(2)} \\ u_1^{(3)} \end{bmatrix} = 0. \tag{3.55}$$

The above system has 3 variables and 4 equations. Multiplying the j-th column by $b_{1j}^{-2} b_{2j}^{-1} b_{4j}$ for $j = 1, 2, 3$ and using the condition (3.54), it follows that the rank of the coefficient matrix T is the same as the rank of the 4 by 3 matrix B with entries b_{ij}. By the definition of b_{ij}, the rank of T is at most 2; hence, the linear system (3.55) has a nonzero solution u_1.

To complete the construction, u_1 should satisfy the constraints

$$u_1^{(i)} > 0 \qquad i = 1, 2, 3 \tag{3.56}$$

and

$$u_1^{(1)} + u_1^{(2)} + u_1^{(3)} = 1. \tag{3.57}$$

The above conditions are fulfilled if we look for a solution $\{\alpha_1, \alpha_3, \alpha_5, \alpha_7\}$ of (3.54) such that

$$\alpha_1^{-1},\ \alpha_3^{-1} \in (\lambda_1, \lambda_2) \qquad \text{and} \qquad \alpha_5^{-1},\ \alpha_7^{-1} \in (\lambda_2, \lambda_3). \tag{3.58}$$

In this case, we may choose

$$u_1 = t \left[b_{11}^{-2} b_{21}^{-1} \left(\frac{b_{13}}{b_{43}} - \frac{b_{12}}{b_{42}} \right),\ b_{12}^{-2} b_{22}^{-1} \left(\frac{b_{11}}{b_{41}} - \frac{b_{13}}{b_{43}} \right),\ b_{13}^{-2} b_{23}^{-1} \left(\frac{b_{12}}{b_{42}} - \frac{b_{11}}{b_{41}} \right) \right]^{\mathsf{T}}, \tag{3.59}$$

where $t > 0$ is a scaling factor such that (3.57) holds. Therefore, if we choose $\{\alpha_1, \alpha_3, \alpha_5, \alpha_7\}$ satisfying (3.54) and (3.58) and furthermore u_1 from (3.59), relation (3.52) holds. Hence, we have that $u_{8+i} = u_i$ and $\alpha_{8+i} = \alpha_i$ for all $i \geq 1$.

Now we discuss a possible choice of $\tau > 0$ in (3.54). Specifically, we are interested in the maximal value τ^* of τ such that (3.54) and (3.58) hold. By continuity assumption, we know that suitable solutions exist for any $\tau \in (0, \tau^*)$. This leads to the maximization problem

$$\max \left\{ \tau : \prod_{i=1}^{4} b_{ij} = \tau \, (j = 1, 2, 3);\ \alpha_1^{-1},\ \alpha_3^{-1} \in (\lambda_1, \lambda_2), \alpha_5^{-1},\ \alpha_7^{-1} \in (\lambda_2, \lambda_3) \right\}. \tag{3.60}$$

To solve (3.60), we consider the Lagrangian function

$$L(\tau, \alpha_1, \alpha_3, \alpha_5, \alpha_7, \mu_1, \mu_2, \mu_3) = \tau + \sum_{j=1}^{3} \mu_j \left[\tau - \prod_{i=1}^{4} (1 - \alpha_{2i-1}\lambda_j) \right], \tag{3.61}$$

where $\{\mu_j\}$ are the multipliers corresponding to equality constraints. Since at a KKT point of (3.60) the partial derivatives of L are zero, we require $\{\mu_i\}$ to satisfy the relation (3.54), $\mu_1 + \mu_2 + \mu_3 = 1$, and

$$\sum_{j=1}^{3} \mu_j \lambda_j \prod_{\substack{l\,=\,1 \\ l\,\neq\, i}}^{4} (1 - \alpha_{2l-1}\lambda_j) = 0 \quad (i = 1, 2, 3, 4). \tag{3.62}$$

Dividing each relation in (3.62) by τ and using (3.54), we obtain the following linear equations for $\mu = (\mu_1, \mu_2, \mu_3)^{\mathsf{T}}$,

$$H\mu = 0, \quad \text{where } H \in \Re^{4\times 3} \text{ with } h_{ij} = \lambda_j b_{ij}^{-1}. \tag{3.63}$$

To guarantee that the system (3.63) has a nonzero solution μ, the rank of the coefficient matrix H must be at most 2. Let $H_{3,3}$ denote the submatrix formed by the first three rows of H. By direct calculation, we obtain

$$\det(H_{3,3}) = \frac{\lambda_1\lambda_2\lambda_3(\lambda_1 - \lambda_2)(\lambda_2 - \lambda_3)(\lambda_3 - \lambda_1)(\alpha_1 - \alpha_3)(\alpha_3 - \alpha_5)(\alpha_5 - \alpha_1)}{\prod_{i,j\in\{1,2,3\}} b_{ij}} \tag{3.64}$$

Thus, $\det(H_{3,3}) = 0$ and inequality constraints (3.58) lead to $\alpha_1 = \alpha_3$. Similarly, we can get $\alpha_5 = \alpha_7$. From (3.54) we know that (3.60) achieves its maximum

$$\tau^* = \frac{(\lambda_1 - \lambda_2)^2(\lambda_2 - \lambda_3)^2}{(\lambda_1\lambda_2 + \lambda_2\lambda_3 + \lambda_3\lambda_1 - \lambda_2^2)^2} \tag{3.65}$$

at

$$\alpha_1^* = \alpha_3^* = (M - \gamma)^{-1}, \qquad \alpha_5^* = \alpha_7^* = (M + \gamma)^{-1}, \tag{3.66}$$

where $M = \frac{\lambda_1+\lambda_3}{2}$, $\gamma = \sqrt{\frac{\xi_1+\xi_2}{2}}$, $\xi_1 = (M - \lambda_1)^2$ and $\xi_2 = (M - \lambda_2)^2$. From the continuity argument we know that there exist cyclic examples of the CBB method with $m = 2$ for any $\tau \in (0, \tau^*)$. For example, we may consider the following symmetric subfamily of examples with $\eta \in (0, \frac{1}{2}]$,

$$\alpha_1, \alpha_5 = \left[M \mp \sqrt{\eta\xi_1 + (1-\eta)\xi_2} \right]^{-1}, \ \alpha_3, \alpha_7 = \left[M \mp \sqrt{(1-\eta)\xi_1 + \eta\xi_2} \right]^{-1}. \tag{3.67}$$

It is easy to check that the above $\{\alpha_i\}$ satisfies (3.54) and (3.58). When η moves from 0 to $\frac{1}{2}$, we can see that the value τ moves from 0 to τ^*.

Now we present some numerical examples. Suppose that $\lambda_1 = 1$, $\lambda_2 = 5$ and $\lambda_3 = 8$. Because of (3.66), we choose $\alpha_1^* = \alpha_3^* = \frac{1}{2}$ and $\alpha_5^* = \alpha_7^* = \frac{1}{7}$ from where the maximizer $\tau^* = \frac{9}{49}$ is found. From (3.55) we get $u_1 = (\frac{972}{1001}, \frac{28}{1001}, \frac{1}{1001})^{\mathsf{T}}$. By the definition of u_1, the previous discussions and by choosing $g_1 = \bar{t}\,(\pm 18\sqrt{3}, \pm 2\sqrt{7}, \pm 1)^{\mathsf{T}}$ with any $\bar{t} > 0$ and $\alpha_1 = \frac{1}{2}$, the CBB method with $m = 2$ produces cycling of the sequence given by $\{u_i\}$ and $\{\alpha_i\}$.

By assuming that the Hessian matrix is $A = \text{diag}(1, 5, 8)$, we also compute the sequences $\{u_{2i-1}\}$ and $\{\alpha_{2i-1}\}$ generated by (3.50) and (3.51). Initial values for u_1 and α_1 are obtained by a steepest descent step at u_0, i.e.,

$$\alpha_1 = \alpha_0 = \frac{u_0^{\mathsf{T}} u_0}{u_0^{\mathsf{T}} A u_0}; \quad u_1^{(i)} = \frac{(1 - \alpha_0 \lambda_i)^2 (u_0^{(i)})^2}{\sum_l (1 - \alpha_0 \lambda_i)^2 (u_0^{(i)})^2} \ (i = 1, 2, 3).$$

For different u_0, we see that different cycles are obtained, which are numerically stable. In Table 2, the index $\bar{k}$ can be different for each vector u_0 so that $\alpha_{\bar{k}+1}^{-1}$, $\alpha_{\bar{k}+3}^{-1} \in (\lambda_1, \lambda_2)$.

u_0^{T}	$\alpha_{\bar{k}+1}^{-1}$	$\alpha_{\bar{k}+3}^{-1}$	$\alpha_{\bar{k}+5}^{-1}$	$\alpha_{\bar{k}+7}^{-1}$	τ
$(1, 2, 3)$	4.9103	1.0000	8.0000	5.0008	4.2186E-6
$(1, 3, 2)$	3.2088	1.3409	6.9100	7.2058	1.2890E-1
$(2, 1, 3)$	1.1099	1.2764	5.0197	7.9938	1.5024E-2
$(2, 3, 1)$	1.5797	2.0807	5.7248	7.7683	1.3706E-1
$(3, 1, 2)$	4.9846	1.0026	7.9086	7.7458	1.6018E-3
$(3, 2, 1)$	1.0015	4.9912	7.8776	7.8866	9.4127E-4

Table 2: Different choices of u_0 generate different cycles

□

3.3. Comparison with steepest descent

The analysis in Section 2.2 shows that CBB with $m = 2$ is at best linearly convergent. By (3.48) and (3.54), we obtain

$$\|g_{k+8}\|_2 = \tau^2 \|g_k\|_2, \quad \text{for all } k \geq 1, \tag{3.68}$$

where τ is the parameter in (3.54). The above relation implies that the convergence rate of the method only depends on the value τ. Furthermore, Table 2 tells us that this value of τ is related to the starting point. It may be very small or relatively large. The maximal possible value of τ is the τ^* in (3.65). In the 3-dimensional case, we get

$$\|g_{k+1}\|_2 \leq \frac{\lambda_3 - \lambda_1}{\lambda_3 + \lambda_1} \|g_k\|_2 \tag{3.69}$$

for the steepest descent method, see [1]. It is not difficult to show that

$$\tau^* < \left[\frac{\lambda_3 - \lambda_1}{\lambda_3 + \lambda_1}\right]^4. \tag{3.70}$$

Thus, we see that CBB with $m = 2$ is faster than the steepest descent method if $n = 3$. This result could be extended to the arbitrary dimensions since we observe that CBB with $m = 2$ generates similar cycles for higher-dimensional quadratics.

The examples provided in Section 2.2 for CBB with $m = 2$ are helpful in understanding and analyzing the behavior of other nonmonotone gradient methods. For example, we can also use the same technique to construct cyclic examples for the alternate step (AS) gradient method, at least theoretically. The AS method corresponds to the cyclic steepest descent method (1.4) with $m = 2$. In fact, if we define u_k as in (3.49), we obtain for all $k \geq 1$

$$\alpha_{2k-1} = \frac{\sum u_{2k-1}^{(l)}}{\sum \lambda_l u_{2k-1}^{(l)}}, \quad u_{2k+1}^{(i)} = \frac{(1 - \alpha_{2k-1}\lambda_i)^4 u_{2k-1}^{(i)}}{\sum (1 - \alpha_{2k-1}\lambda_l)^4 u_{2k-1}^{(l)}} \tag{3.71}$$

for $i = 1, \ldots, n$. For any n with $u_{2n+1} = u_1$ and $\alpha_{2n+1} = \alpha_1$, we require the stepsizes $\{\alpha_{2k-1} : k = 1, \ldots, n-1\}$ to satisfy

$$\left|\prod_{i=1}^{n-1} b_{ij}\right| = \tau, \quad j = 1, \ldots, n, \tag{3.72}$$

where b_{ij} is given by (3.53). At the same time, we obtain the following linear equations for u_1

$$T u_1 = 0, \quad \text{where } T \in R^{(n-1)\times n} \text{ with } T_{ij} = b_{ij} \textstyle\prod_{l=1}^{i-1} b_{lj}^4. \tag{3.73}$$

The above system (3.73) has n variables, but $n - 1$ equations. If there is a positive solution $\bar{u}_1$, then we may scale the vector and obtain another positive solution $u_1 = c\bar{u}_1$ with $\sum_l u_1^{(l)} = 1$, which completes the construction of a cyclic example. Here we present a 5-dimensional example. We first fix $\alpha_1 = 1$, $\alpha_3 = 0.1$, $\alpha_5 = 0.2$ and $\alpha_7 = 0.0625$, and then choose

$$\lambda = (0.73477,\ 1.3452,\ 4.2721,\ 10.554,\ 16.154)$$

which are five roots of the equation $\prod_{k=1}^{4}(1 - \alpha_{2k-1}w) = 0.2$. Therefore, we get the matrix

$$T = \begin{pmatrix} 0.26523 & -0.34515 & -3.2721 & -9.5537 & -15.154 \\ 0.00458 & 0.01228 & 65.659 & -461.26 & -32451 \\ 0.00311 & 0.00582 & 1.7964 & -0.08696 & -16870 \\ 0.00184 & 0.00208 & 0.00406 & 0.04056 & -1800.5 \end{pmatrix}.$$

The system $Tu_1 = 0$ has the positive solution

$$\bar{u}_1 = (5.6163\text{E+}5, 3.3397\text{E+}5, 7.3848\text{E+}3, 9.9533\text{E+}2, 1.0)^{\mathsf{T}}$$

which leads to

$$u_1 = (6.2128\text{E-}1, 3.6945\text{E-}1, 8.1693\text{E-}3, 1.1011\text{E-}3, 1.1062\text{E-}6)^{\mathsf{T}}.$$

Therefore, if we choose the above initial vector u_1, we get $u_{10k+1} = u_1$ and $\alpha_{10k+1} = \alpha_1$ for all $k \geq 1$, and hence the AS method falls into a cycle.

4 An adaptive cyclic BB method

In this section, we examine the convergence speed of CBB for different values of $m \in [1, 7]$, using quadratic programming problems of the form:

$$f(x) = \frac{1}{2} x^{\mathsf{T}} \text{diag}(\lambda_1, \cdots, \lambda_n) x, \quad x \in \Re^n. \tag{4.74}$$

We will see that the choice for m has a significant impact on performance. This leads us to propose an adaptive choice for m. The BB algorithm with this adaptive choice for m and a nonmonotone line search is called ACBB. Numerical comparisons with SPG2 and with conjugate gradient codes using the CUTEr test problem library are given later in Section 4.

4.1. A numerical investigation of cyclic BB

We consider the test problem (4.74) with four different condition numbers C for the diagonal matrix: $C = 10^2$, $C = 10^3$, $C = 10^4$, and $C = 10^5$; and with three different dimensions $n = 10^2$, $n = 10^3$, and $n = 10^4$. We let $\lambda_1 = 1$, $\lambda_n = C$, the condition number. The other diagonal elements λ_i, $2 \leq i \leq n-1$, are randomly generated on the interval $(1, \lambda_n)$. The starting points $x_1^{(i)}$, $i = 1, \cdots, n$, are randomly generated on the interval $[-5, 5]$. The stopping condition is

$$\|g_k\|_2 \leq 10^{-8}.$$

For each case, 10 runs are made and the average number of iterations required by each algorithm is listed in Table 3 (under the columns labeled BB and CBB). The upper bound for the number of iterations is 9999. If this upper bound is exceeded, then the corresponding entry in Table 3 is F.

In Table 3 we see that $m = 2$ gives the worst numerical results – in Section 3 we saw that as m increases, convergence became superlinear. For each case, a suitably chosen m drastically improves the efficiency the BB method. For example, in case of $n = 10^2$ and $cond = 10^5$, CBB with $m = 7$ only requires one fifth of the iterations of the BB method. The optimal choice of m varies from one test case to another. If the problem condition is relatively small ($cond = 10^2$, 10^3), a smaller value m (3 or 4) is preferred. If the problem condition is relatively large ($cond = 10^4$, 10^5), a larger value of m is more efficient. This observation is the motivation for introducing an adaptive choice for m in the CBB method.

Our adaptive idea arises from the following considerations. If a stepsize is used infinitely often in the gradient method; namely, $\alpha_k \equiv \alpha$, then under the assumption that the function Hessian A has no multiple eigenvalues, the gradient g_k must approximate an eigenvector of A, and $g_k^\top A g_k / g_k^\top g_k$ tends to the corresponding eigenvalue of A, see [10]. Thus, it is reasonable to assume that repeated use of a BB stepsize leads to good approximations of eigenvectors of A. First, we define

$$\nu_k = \frac{g_k^\top A g_k}{\|g_k\| \, \|A g_k\|}. \tag{4.75}$$

If g_k is exactly an eigenvector of A, we know that $\nu_k = 1$. If $\nu_k \approx 1$, then g_k can be regarded as an approximation of an eigenvector of A and $\alpha_k^{BB} \approx \alpha_k^{SD}$. In this case, it is worthwhile to calculate a new BB stepsize α_k^{BB} so that the method accepts a step close to the steepest descent step. Therefore, we test the condition

$$\nu_k \geq \beta, \tag{4.76}$$

where $\beta \in (0, 1)$ is constant. If the above condition holds, we calculate a new BB stepsize. We also introduce a parameter $\bar{M}$, and if the number of cycles $m > \bar{M}$, we calculate a new BB stepsize. Numerical results for this adaptive CBB with $\beta = 0.95$ are listed under the column *adaptive* of Table 3, where two values $\bar{M} = 5$, 10 are tested.

From Table 3, we see that the adaptive strategy makes sense. The performance with $\bar{M} = 5$ or $\bar{M} = 10$ is uniformly better than that of the BB method. This motivates the use of a similar strategy for designing an efficient gradient algorithms for unconstrained optimization.

		BB	CBB						*adaptive*	
n	*cond*		$m=2$	$m=3$	$m=4$	$m=5$	$m=6$	$m=7$	$\bar{M}=5$	$\bar{M}=10$
10^2	10^2	147	219	156	145	150	160	166	136	134
	10^3	505	2715	468	364	376	395	412	367	349
	10^4	1509	*F*	1425	814	852	776	628	878	771
	10^5	5412	*F*	5415	3074	1670	1672	1157	2607	1915
10^3	10^2	147	274	160	158	162	166	181	150	145
	10^3	505	1756	548	504	493	550	540	481	460
	10^4	1609	*F*	1862	1533	1377	1578	1447	1470	1378
	10^5	5699	*F*	6760	4755	3506	3516	2957	4412	3187
10^4	10^2	156	227	162	166	167	170	187	156	156
	10^3	539	3200	515	551	539	536	573	497	505
	10^4	1634	*F*	1823	1701	1782	1747	1893	1587	1517
	10^5	6362	*F*	6779	5194	4965	4349	4736	4687	4743

Table 3: Comparing CBB(m) method with an adaptive CBB method

4.2. Nonmonotone line search and cycle number

As mentioned in Section 1, the choice of the stepsize α_k is very important for the performance of a gradient method. For the BB method, function values do not decrease monotonically. Hence, when implementing BB or CBB, it is important to use a nonmonotone line search.

Assuming that d_k is a descent direction at the k-th iteration ($g_k^\mathsf{T} d_k < 0$), a common termination condition for the steplength algorithm is

$$f(x_k + \alpha_k d_k) \le f_r + \delta \alpha_k g_k^\mathsf{T} d_k, \tag{4.77}$$

where f_r is the so-called reference function value and $\delta \in (0, 1)$ a constant. If $f_r = f(x_k)$, then the line search is monotone since $f(x_{k+1}) < f(x_k)$. The nonmonotone line search proposed in [26] chooses f_r to be the maximum function value for the M most recent iterates. That is, at the k-th iteration, we have

$$f_{\max} = \max_{0 \le i \le \min\{k, M-1\}} f(x_{k-i}). \tag{4.78}$$

This nonmonotone line search is used by Raydan [34] to obtain GBB. Dai and Schittkowski [15] extended the same idea to constrained nonlinear optimization and a sequential quadratic programming method. An even more adaptive way of choosing f_r is proposed by Toint [38] for trust region algorithms and then extended by Dai and Zhang [18]. Compared with (4.78), the new adaptive way of choosing f_r allows big jumps in function values, and is therefore very suitable for the BB algorithm, see [18], [12], and [13].

The numerical results which we report in this section are based on the nonmonotone line search algorithm given in [18]. The only difference with the algorithm given in [18] is that the starting guess for the stepsize coincides with the prior BB step until the cycle length has been reached, at which point we recompute the step using the BB formula; moreover, in each subsequent subiteration, after computing a new BB step, we replace (4.77) by

$$f(x_k + \bar{\alpha}_k d_k) \le \min\{f_{\max}, f_r\} + \delta \bar{\alpha}_k g_k^\mathsf{T} d_k,$$

where f_r is the reference value given in [18] and $\bar{\alpha}_k$ is the initial trial stepsize (the previous BB step). It is proved in [18, Thm. 3.2] that the criteria given in [18] for choosing the nonmonotone stepsize ensures convergence in the sense that

$$\liminf_{k \to \infty} \|g_k\| = 0.$$

We now explain how we decided to terminate the current cycle, and recompute the stepsize using the BB formula. Notice that the reinitialization of the stepsize has no effect on convergence, it only effects the initial stepsize used in the line search. Loosely, we would like to compute a new BB step in any of the following cases:

R1. The number of times m the current BB stepsize has been reused is sufficiently large: $m \ge \overline{M}$, where $\overline{M}$ is a constant.

R2. The following nonquadratic analogue of (4.76) is satisfied:

$$\frac{s_k^\mathsf{T} y_k}{\|s_k\|_2 \|y_k\|_2} \geq \beta, \tag{4.79}$$

where $\beta < 1$ is near 1. We feel that the condition (4.79) should only be used in a neighborhood a local minimizer, where f is approximately quadratic. Hence, we only use the condition (4.79) when the stepsize is sufficiently small:

$$\|s_k\|_2 < \min\{\frac{c_1 f_{k+1}}{\|g_{k+1}\|_\infty}, 1\}, \tag{4.80}$$

where c_1 is a constant.

R3. The current step s_k is sufficiently large:

$$\|s_k\|_2 \geq \max\{c_2 \frac{f_{k+1}}{\|g_{k+1}\|_\infty}, 1\}, \tag{4.81}$$

where c_2 is a constant.

R4. In the previous iteration, the BB step was truncated in the line search. That is, the BB step had to be modified by the nonmonotone line search routine to ensure convergence.

Nominally, we recompute the BB stepsize in any of the cases R1–R4. One case where we prefer to retain the current stepsize is the case where the iterates lie in a region where f is not strongly convex. Notice that if $s_k^\mathsf{T} y_k < 0$, then there exists a point between x_k and x_{k+1} where the Hessian of f has negative eigenvalues. In detail, our rules for terminating the current cycle and reinitializing the BB stepsize are the following:

Cycle termination/Stepsize initialization

T1. If any of the condition R1 through R4 are satisfied and $s_k^\mathsf{T} y_k > 0$, then the current cycle is terminated and the initial stepsize for the next cycle is given by

$$\alpha_{k+1} = \max\{\alpha_{\min}, \min\{\frac{s_k^\mathsf{T} s_k}{s_k^\mathsf{T} y_k}, \alpha_{\max}\},$$

where $\alpha_{\min} < \alpha_{\max}$ are fixed constants.

T2. If the length m of the current cycle satisfies $m \geq 1.5\overline{M}$, then the current cycle is terminated and the initial stepsize for the next cycle is given by

$$\alpha_{k+1} = \max\{1/\|g_{k+1}\|_\infty, \alpha_k\}.$$

4.3. Numerical results

In this subsection, we compare the performance of our adaptive cyclic BB stepsize algorithm, denoted ACBB, with the SPG2 algorithm of Birgin, Martínez, and Raydan [5], with the PRP+ conjugate gradient code developed by Gilbert and Nocedal [24], and with the CG_DESCENT code of Hager and Zhang [28, 29]. The SPG2 algorithm is an extension of Raydan's [34] GBB algorithm. In our tests, we set the bounds in SPG2 to infinity. The PRP+ code is available at:

http://www.ece.northwestern.edu/∼nocedal/software.html

The CG_DESCENT code is found at:

http://www.math.ufl.edu/∼hager/papers/CG

The line search in the PRP+ code is a modification of subroutine CSRCH of Moré and Thuente [32], which employs various polynomial interpolation schemes and safeguards in satisfying the strong Wolfe conditions. CG_DESCENT employs an "approximate Wolfe" line search. All codes are written in Fortran and compiled with f77 under the default compiler settings on a Sun workstation.

The parameters of the ACBB algorithm are $\alpha_{\min} = 10^{-30}$, $\alpha_{\max} = 10^{30}$, $c_1 = c_2 = 0.1$, and $\overline{M} = 4$. For the initial iteration, the starting stepsize for the line search was $\alpha_1 = 1/\|g_1\|_\infty$. The parameter values for the nonmonotone line search routine from [18] were $\delta = 10^{-4}$, $\sigma_1 = 0.1$, $\sigma_2 = 0.9$, $\beta = 0.975$, $L = 3$, $M = 8$, and $P = 40$.

Our numerical experiments are based on the entire set of 160 unconstrained optimization problem available from CUTEr in the Fall, 2004. As explained in [29], we deleted problems that were small, or problems where different solvers converged to different local minimizers. After the deletion process, we were left with 111 test problems with dimension ranging from 50 to 10^4.

Nominally, our stopping criterion was the following:

$$\|\nabla f(x_k)\|_\infty \le \max\{10^{-6}, 10^{-12}\|\nabla f(x_0)\|_\infty\}. \tag{4.82}$$

In a few cases, this criterion was too lenient. For example, with the test problem PENALTY1, the computed cost still differs from the optimal cost by a factor of 10^5 when the criterion (4.82) is satisfied. As a result, different solvers obtain completely different values for the cost, and the test problem would be discarded. By changing the convergence criterion to $\|\nabla f(x_k)\|_\infty \le 10^{-6}$, the computed costs all agreed to 6 digits. The problems for which the convergence criterion was strengthened were DQRTIC, PENALTY1, POWER, QUARTC, and VARDIM.

The CPU time in seconds and the number of iterations, function evaluations, and gradient evaluations for each of the methods are posted at the following web site:

http://www.math.ufl.edu/∼hager/papers/CG (4.83)

Here we analyze the performance data using the profiles of Dolan and Moré [19]. That is, we plot the fraction P of problems for which any given method is within

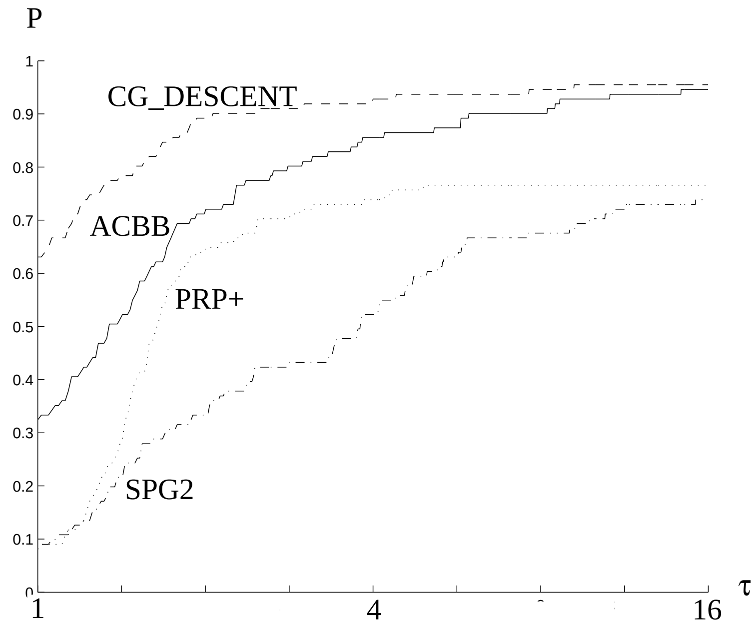

Figure 2: Performance based on CPU time

a factor τ of the best time. In a plot of performance profiles, the top curve is the method that solved the most problems in a time that was within a factor τ of the best time. The percentage of the test problems for which a method is the fastest is given on the left axis of the plot. The right side of the plot gives the percentage of the test problems that were successfully solved by each of the methods. In essence, the right side is a measure of an algorithm's robustness.

In Figure 2, we use CPU time to compare the performance of the four codes ACBB, SPG2, PRP+, and CG_DESCENT. The best performance, relative to the CPU time metric, was obtained by CG_DESCENT, the top curve in Figure 2, followed by ACBB. For this collection of methods, the number of times any method achieved the best time is shown in Table 1. The column total in Table 1 exceeds 111 due to ties for some test problems.

The results of Figure 2 indicate that ACBB is much more efficient than SPG2, while it performed better than PRP+, but not as well as CG_DESCENT. From the experience in [34], the GBB algorithm, with a traditional nonmonotone line search [26], may be affected significantly by nearly singular Hessians at the solution. We observe that nearly singular Hessians do not affect ACBB significantly. In fact, ACBB becomes more efficient as the problem becomes more singular. Furthermore, since ACBB does not need to calculate the BB stepsize at every iteration, CPU time is saved, which can be significant when the problem dimension is large. For this test set, we found that the average cycle length for ACBB was 2.59. In other words, the

Method	Fastest
CG_DESCENT	70
ACBB	36
PRP+	9
SPG2	9

Table 1: Number of times each method was fastest (time metric, stopping criterion (4.82))

BB step is reevaluated after 2 or 3 iterations, on average.

Even though ACBB did not perform as well as CG_DESCENT for the complete set of test problems, there were some cases where it performed exceptionally well (see Table 2). One important advantage of the ACBB scheme over conjugate gradient routines such as PRP+ or CG_DESCENT is that in many cases, the stepsize for ACBB is either the previous stepsize or the BB sizesize (1.5). In contrast, with conjugate gradient routines, each iteration requires a line search. Due to the simplicity of the ACBB stepsize, it can be more efficient when the iterates are in a regime where the function is irregular and the asymptotic convergence properties of the conjugate gradient method are not in effect. One such application is bound constrained optimization problems – as components of x reach the bounds, these components are often held fixed, and the associated partial derivative change discontinuously.

Problem	Dimension	ACBB	CG_DESCENT
FLETCHER	5000	9.14	989.55
FLETCHER	1000	1.32	27.27
BDQRTIC	1000	.37	3.40
VARDIM	10000	.05	2.13
VARDIM	5000	.02	.92

Table 2: CPU times for selected problems

5 Conclusion and discussion

In this paper, we analyze the cyclic Barzilai-Borwein method. For general nonlinear functions, we prove linear convergence. For convex quadratic functions, our numerical results indicate that when $m > n/2 \geq 3$, CBB is likely R-superlinearly. For the special case $n = 3$ and $m = 2$, the convergence rate, in general, is no better than linear. By utilizing nonmonotone line search techniques, we develop an adaptive cyclic BB stepsize algorithm (ACBB) for general nonlinear unconstrained optimization problems.

The test results in Figure 2 indicate that ACBB is significantly faster than SPG2. Since the mathematical foundations of ACBB and the conjugate gradient algorithms are completely different, the performance seems to depend on the problem. Roughly speaking, if the objective function is “close” to quadratic, the conjugate gradient routines seem to be more efficient; if the objective function is highly nonlinear, then ACBB is comparable to or even better than conjugate gradient algorithms.

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, 11 (1959), pp. 1–17.

[2] J. Barzilai and J. M. Borwein, *Two point step size gradient methods*, IMA J. Numer. Anal., 8 (1988), pp. 141–148.

[3] E. G. Birgin, I. Chambouleyron, and J. M. Martínez, *Estimation of the optical constants and the thickness of thin films using unconstrained optimization*, J. Comput. Phys., 151 (1999), pp. 862–880.

[4] E. G. Birgin and Y. G. Evtushenko, *Automatic differentiation and spectral projected gradient methods for optimal control problems*, Optim. Methods Softw., 10 (1998), pp. 125–146.

[5] E. G. Birgin, J. M. Martínez, and M. Raydan, *Nonmonotone spectral projected gradient methods for convex sets*, SIAM Journal on Optimization, 10:4 (2000), pp. 1196–1211.

[6] M. Brendel, *On two algorithms for the minimization of box constrained quadratic functions*, MS344 honours project, Department of Mathematics, University of Dundee, 2001.

[7] A. Cauchy, *Méthode générale pour la résolution des systèms d'equations simultanées*, Comp. Rend. Sci. Paris, 25 (1847), pp. 46–89.

[8] I. Bongartz, A. R. Conn, N. I. M. Gould, and P. L. Toint, *CUTE: Constrained and unconstrained testing environments*, ACM Trans. Math. Software, 21 (1995), pp. 123–160.

[9] Y. H. Dai, *On the nonmonotone line search*, Journal of Optimization Theory and Applications 112:2 (2002), pp. 315–330.

[10] Y. H. Dai, *Alternate stepsize gradient method*, Optimization, 52 (2003), pp. 395–415.

[11] Y. H. Dai and R. Fletcher, *On the asymptotic behaviour of some new gradient methods*, Numerical Analysis Report NA/212, Department of Mathematics, University of Dundee, 2003.

[12] Y. H. Dai and R. Fletcher, *Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming*, University of Dundee Report NA/215, 2003.

[13] Y. H. Dai and R. Fletcher, *New algorithms for singly linearly constrained quadratic programs subject to lower and upper bounds*, Numerical Analysis Report NA/216, Department of Mathematics, University of Dundee, 2003.

[14] Y. H. Dai and L. Z. Liao, *R-linear convergence of the Barzilai and Borwein gradient method*, IMA Journal of Numerical Analysis, 22 (2002), pp. 1–10.

[15] Y. H. Dai and K. Schittkowski, *Solving constrained nonlinear programming problems by sequential quadratic programming with non-monotone line search*, submitted for publication

[16] Y. H. Dai and X. Q. Yang, *A new gradient method with an optimal stepsize property*, Research report, Institute of Computational Mathematics and Scientific/Engineering Computing, Chinese Academy of Sciences, 2001 .

[17] Y. H. Dai and Y. X. Yuan, *Alternate minimization gradient method*, IMA Journal of Numerical Analysis, 23 (2003), pp. 377–393.

[18] Y. H. Dai and H. Zhang, *Adaptive two-point stepsize gradient algorithm*, Numerical Algorithms, 27 (2001), pp. 377–385.

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

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

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

[22] G. E. Forsythe, *On the asymptotic directions of the s-dimensional optimum gradient method*, Numer. Math., 11 (1968), pp. 57–76.

[23] A. Friedlander, J. M. Martínez, B. Molina, and M. Raydan, *Gradient method with retards and generalizations*, SIAM J. Numer. Anal., 36 (1999), pp. 275–289.

[24] J. C. Gilbert and J. Nocedal, *Global convergence properties of conjugate gradient methods for optimization*, SIAM J. Optim., 2 (1992), pp. 21–42.

[25] W. Glunt, T. L. Hayden, and M. Raydan, *Molecular conformations from distance matrices*, J. Comput. Chem., 14 (1993), pp. 114–120.

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

[27] L. Grippo and M. Sciandrone, *Nonmonotone globalization techniques for the Barzilai-Borwein gradient method*, Comput. Optim. Appl., 23 (2002), pp. 143–169.

[28] W. W. Hager and H. Zhang, *A new conjugate gradient method with guaranteed descent and an efficient line search*, to appear in SIAM J. Optim.

[29] W. W. Hager and H. Zhang, *CG_DESCENT, A conjugate gradient method with guaranteed descent*, submitted to ACM Transactions on Mathematical Software, January, 15, 2004.

[30] Y. L. Lai, *Some properties of the steepest descent method*, Acta Mathematicae Applicatae Sinica, 4 (1981), 106–116 (in Chinese).

[31] W. B. Liu and Y. H. Dai, *Minimization algorithms based on supervisor and searcher cooperation*, Journal of Optimization Theory and Applications, 111 (2001), pp. 359–379.

[32] J. J. Moré and D. J. Thuente, *Line search algorithms with guaranteed sufficient decrease*, ACM Trans. Math. Software, 20 (1994), pp. 286–307.

[33] M. Raydan, *On the Barzilai and Borwein choice of steplength for the gradient method*, IMA J. Numer. Anal., 13 (1993), pp. 321–326.

[34] M. Raydan, *The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem*, SIAM J. Optim., 7 (1997), pp. 26–33.

[35] M. Raydan, *Nonmonotone spectral methods for large-scale nonlinear systems* Report in the International Workshop on "Optimization and Control with Applications", Erice, Italy, July 9–17, 2001

[36] M. Raydan and B. F. Svaiter, *Relaxed steepest descent and Cauchy-Barzilai-Borwein method*, Comput. Optim. Appl., 21 (2002), pp. 155–167.

[37] T. Serafini, G. Zanghirati, L. Zanni, *Gradient projection methods for quadratic programs and applications in training support vector machines*, to appear in: Optimization Methods and Software, available at (http://www.optimization-online.org), 2003.

[38] Ph. L. Toint, *A non-monotone trust region algorithm for nonlinear optimization subject to convex constraints*, Math. Program., 77 (1997), pp. 69–94.