Journal of Computational and Applied Mathematics 234 (2010) 2986–2992

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Notes on the Dai–Yuan–Yuan modified spectral gradient method☆

Yunhai Xiao [a,*], Qiuyu Wang [b], Dong Wang [b]

[a] Institute of Applied Mathematics, College of Mathematics and Information Science, Henan University, Kaifeng, 475004, PR China

[b] Department of Public Computer Teaching and Research, Henan University, Kaifeng, 475004, PR China

ARTICLE INFO

Article history:
Received 21 April 2009
Received in revised form 8 April 2010

MSC:
65H10
90C26

Keywords:
Spectral gradient method
Quasi-Newton method
Secant equation
Nonmonotone line search
Global convergence

ABSTRACT

In this paper, we give some notes on the two modified spectral gradient methods which were developed in [10]. These notes present the relationship between their stepsize formulae and some new secant equations in the quasi-Newton method. In particular, we also introduce another two new choices of stepsize. By using an efficient nonmonotone line search technique, we propose some new spectral gradient methods. Under some mild conditions, we show that these proposed methods are globally convergent. Numerical experiments on a large number of test problems from the CUTEr library are also reported, which show that the efficiency of these proposed methods.

1. Introduction

We consider the unconstrained optimization problem

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

where $f : \mathbf{R}^n \to \mathbf{R}$ is a continuously differentiable function. n is the number of variables, which is assumed to be large. Large-scale optimization is an important research area in both optimization theory and algorithm design. There are some kinds of effective methods that are available for solving (1.1), for instance, the inexact Newton method, limited memory quasi-Newton method, nonlinear conjugate gradient method and spectral gradient method.

The spectral gradient method (also named the two-point stepsize method) was originated in [1]. This method consists essentially of a steepest descent method, where the choice of the stepsize along the negative gradient direction is potentially derived from a two-point approximation to the secant equation underlying the quasi-Newton method [2]. The spectral gradient method can be described as the iterative form

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

where g_k is the gradient vector of f at x_k, and the two choices of the scalar α_k are

$$\alpha_k^1 = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}} \tag{1.3}$$

☆ This work was supported by Chinese NSF grant 10761001, Chinese Post-doctoral Foundation grant 20090461094, and the Natural Science Foundation of Henan Province Education Department grant 2010B110004.

* Corresponding address: Department of Mathematics, Nanjing University, Nanjing, 210093, PR China.
E-mail addresses: yhxiao@henu.edu.cn (Y. Xiao), wqy@henu.edu.cn (Q. Wang), wdyumei@henu.edu.cn (D. Wang).

0377-0427/$ – see front matter
doi:10.1016/j.cam.2010.04.012

and

$$\alpha_k^2 = \frac{s_{k-1}^T y_{k-1}}{y_{k-1} y_{k-1}}, \tag{1.4}$$

where $s_{k-1} = x_k - x_{k-1}, y_{k-1} = g_k - g_{k-1}$. Moreover, some practical experiments have shown that choice (1.3) is a promising alternative to choice (1.4).

By incorporating the nonmonotone line search of Grippo, Lampariello, and Lucidi [3], Raydan [4] has extended the spectral gradient method with choice (1.3) for solving general unconstrained optimization. Reported numerical experiments in [4] showed that the method is competitive and sometimes superior to several famous conjugate gradient algorithms. Moreover, Birgin, Martínez, and Raydan [5] have extended the work of Raydan and proposed the well-known spectral projected gradient (SPG) method for the minimization of differential functions on a closed convex set. The work of [5] has inspired many studies on the methods for bound constrained optimization problems (e.g. [6]).

Due to its simplicity and numerical efficiency, the spectral gradient method has received a great deal of attention in recent decades. The effectiveness of the classical spectral gradient method has been significantly improved by incorporating it with new and fast nonmonotone line search techniques (e.g. [7,2]). The spectral gradient method does not guarantee a descent in the objective function at each iteration, but performs better than the classical steep descent (SD) method in practice. An interesting fact is that an alternating strategy that uses the SD step and spectral gradient step alternately can accelerate the rate of the spectral gradient method. An important work on this scheme was due to the cycle Barzilai–Borwein (CBB) method (see [8,9]). An implementation of the CBB method, combined with a nonmonotone line search, shows that this method performs better than the existing spectral gradient method. It is even competitive with some other well-known standard codes (see [9]).

We note that all the developments of spectral gradient method are based on the classical choice of the stepsize (1.3) or (1.4), which only exploits the gradient information at the current and previous step, while neglecting the information of function values available. To the best of our knowledge, the first choice of stepsize with function value information is due to Dai, Yuan, and Yuan (hereafter DYY) [10], in which the choice of stepsize is deduced from the viewpoint of interpolation. The numerical results reported in [10] suggest that improvements have been achieved. In this paper, we further study the spectral gradient method for solving large-scale unconstrained optimization. Specifically, we give two notes on the stepsize choice of DYY. Our main ideas come from some recent works on quasi-Newton methods for unconstrained optimization, e.g., [11–13]. Additionally, we also give another two new stepsize formulae. The global convergence of the corresponding methods based on these stepsize are established. Numerical comparisons between them are also reported.

We organized this paper as follows. In the next section, we simply recall some modified secant equations and related modified quasi-Newton methods for unconstrained optimization. In Section 3, we turn our attention to the modified stepsize formulae of DYY, and consequently give our notes. We also state the steps of our new algorithms with an efficient line search, and show that the new proposed methods converge globally under some mild assumptions in the same section. In Section 4, we report some numerical results. Throughout this paper, the symbol $\|\cdot\|$ denotes the Euclidean norm of a vector.

2. Some secant equations

Quasi-Newton secant methods for unconstrained optimization obey the recursive formula

$$x_{k+1} = x_k - B_k^{-1} g_k,$$

where B_k is an approximation Hessian of f at x_k. The sequence of matrix $\{B_k\}$ satisfies the secant equation

$$B_k s_{k-1} = y_{k-1}. \tag{2.1}$$

Obviously, only two gradients are exploited in the secant equation (2.1), while the function values available are neglected. Hence, techniques using gradients as well as function values have been studied by several authors. An efficient attempt is due to Zhang, Deng, and Chen [13]. They developed a new secant equation which used both gradients and function values. This equation is

$$B_k s_{k-1} = \widetilde{y}_{k-1}, \tag{2.2}$$

in which $\widetilde{y}_{k-1} = y_{k-1} + \widetilde{\gamma} s_{k-1}$, where

$$\widetilde{\gamma} = \frac{3(g_k + g_{k-1})^T s_{k-1} + 6(f_{k-1} - f_k)}{\|s_{k-1}\|^2}.$$

The new secant equation is superior to the usual one (2.1) in the sense that $\widetilde{y}_{k-1}$ better approximates $\nabla^2 f(x_k) s_{k-1}$ than y_{k-1} (see [13]). Consequently, the matrix which is obtained from the corresponding modified quasi-Newton update better approximates the objective function Hessian matrix (see [13]).

Another significant attempt that modified the usual secant equation by using both function values and gradient information is due to Wei, Li, and Qi [11]. Their ideas come from the following simple observation:

$$f_{k-1} \simeq f_k + g_k^T s_{k-1} + \frac{1}{2} s_{k-1}^T \nabla^2 f(x_k) s_{k-1},$$

which shows

$$s_{k-1}^T \nabla^2 f(x_k) s_{k-1} \simeq 2(f_{k-1} - f_k) + 2g_k^T s_{k-1},$$
$$= 2(f_{k-1} - f_k) + (g_k - g_{k-1})^T s_{k-1} + s_{k-1}^T y_{k-1}.$$

Combining with (2.1), and noticing that B_k is an approximation of $\nabla^2 f(x_k)$, this yields

$$B_k s_{k-1} = \bar{y}_{k-1}, \tag{2.3}$$

in which $\bar{y}_{k-1} = y_{k-1} + \bar{\gamma} s_{k-1}$ with the exact choice of $\bar{\gamma}$, i.e.,

$$\bar{\gamma} = \frac{(g_k + g_{k-1})^T s_{k-1} + 2(f_{k-1} - f_k)}{\|s_{k-1}\|^2}.$$

A remarkable property of this secant equation (2.3) is that, if f is twice continuously differentiable and B_k is updated by the BFGS method, then the equality

$$f_{k-1} = f_k + g_k^T s_{k-1} + \frac{1}{2} s_{k-1}^T B_k s_{k-1}$$

holds for all k, and this property is independent of any convexity assumption on the objective function. Furthermore, this equality does not hold for any update formula which is based on the usual secant condition (2.1), even for the new one (2.2). Additionally, comparing with the secant equation (2.2), one concludes that $\bar{\gamma} = \frac{1}{3}\widetilde{\gamma}$. This is a very interesting fact. The superlinear convergence theorem of the corresponding BFGS method which is based on the secant equation (2.3) was established in [12]. Moreover, the work of [12] was extended to deal with large-scale problems in a limited memory scheme in [14]. The reported numerical results show that this extension is beneficial to the performance of the algorithm.

3. Algorithm and properties

In this section, we first recall the modified spectral gradient method for unconstrained optimization. For a one-dimensional optimization problem, the spectral gradient method (1.3) and (1.4) is the secant method. In the higher-dimensional case, the formula (1.3) can be derived from interpolation (see [10]). Consequently, DYY [10] presented two new choices for scalar (1.3), namely,

$$\widetilde{\alpha}_k^1 = \frac{s_{k-1}^T s_{k-1}}{6(f_{k-1} - f_k) + 4s_{k-1}^T g_k + 2s_{k-1}^T g_{k-1}}, \tag{3.1}$$

and

$$\bar{\alpha}_k^1 = \frac{s_{k-1}^T s_{k-1}}{2(f_{k-1} - f_k) + 2g_k^T s_{k-1}}. \tag{3.2}$$

It is not difficult to see that the formulae (3.1) and (3.2) are identical to (1.3) if $f(x)$ is quadratic on the line segment between x_{k-1} and x_k.

Recall that the basic idea of the spectral gradient method is to regard the matrix $D_k = \alpha_k I$ as an inverse approximation of the Hessian $\nabla^2 f(x_k)$, and having a certain quasi-Newton property, it is reasonable to require either

$$\min \|D_k^{-1} s_{k-1} - y_{k-1}\|, \tag{3.3}$$

or

$$\min \|s_{k-1} - D_k y_{k-1}\|. \tag{3.4}$$

This is because in the quasi-Newton method the quasi-Newton matrix satisfies secant equation (2.1). Then from $D_k = \alpha_k I$ and relations (3.3)–(3.4) we obtain the two stepsize formulae, (1.3) and (1.4), respectively.

Now we are ready to give our notes on the modified stepsize formulae (3.1)–(3.2).

Note 3.1. From the above analysis, we know that the generation of the stepsize (1.3) is based on the standard secant equation (2.1). If we use the new secant equation (2.2) to take the place of the general one (2.1), then we have

$$\widetilde{\alpha}_k^1 = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T \widetilde{y}_{k-1}} = \frac{s_{k-1}^T s_{k-1}}{6(f_{k-1} - f_k) + 4s_{k-1}^T g_k + 2s_{k-1}^T g_{k-1}},$$

which is exactly the stepsize (3.1). In a similar way, using the new secant equation (2.3), we can obtain another stepsize (3.2), namely

$$\bar{\alpha}_k^1 = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T \bar{y}_{k-1}} = \frac{s_{k-1}^T s_{k-1}}{2(f_{k-1} - f_k) + 2g_k^T s_{k-1}}.$$

More directly, if we let $\widetilde{y}_{k-1}$ and $\bar{y}_{k-1}$ take the place of y_{k-1} in formula (1.3), then we get (3.1) and (3.2), respectively.

Note 3.2. If the objective function is twice continuously differentiable, then

$$f_{k-1} = f_k - g_k^T s_{k-1} + \frac{1}{2} s_{k-1}^T \nabla^2 f(x_{k-1}) s_{k-1}. \tag{3.5}$$

Let the Hessian matrix $\nabla^2 f(x_{k-1})$ be approximated by an identify matrix times a scalar, i.e., $\nabla^2 f(x_{k-1}) \simeq \frac{1}{\alpha} I$. Combining with (3.5) yields

$$\alpha = \frac{s_{k-1}^T s_{k-1}}{2(f_{k-1} - f_k) + 2 g_k^T s_{k-1}}.$$

In this case, we also get stepsize (3.2).

From Note 3.2, we see that the formulae (3.1) and (3.2) are only concerned with (1.3). Furthermore, if we turn our attention to stepsize (1.4), and substitute $\widetilde{y}_{k-1}$ and $\bar{y}_{k-1}$ for $\bar{y}_k$ in (1.4), respectively, we can get another two new stepsize formulae:

$$\widetilde{\alpha}_k^2 = \frac{s_{k-1}^T \widetilde{y}_{k-1}}{\widetilde{y}_{k-1}^T \widetilde{y}_{k-1}}, \tag{3.6}$$

and

$$\bar{\alpha}_k^2 = \frac{s_{k-1}^T \bar{y}_{k-1}}{\bar{y}_{k-1}^T \bar{y}_{k-1}}. \tag{3.7}$$

By using the Grippo–Lampariello–Lucide (GLL) nonmonotone line search [3], the methods which are based on (3.1) and (3.2) converge globally. In what follows, we discuss the construction of our new algorithms which are based on these new formulae (3.1), (3.2), (3.6) and (3.7).

To construct our algorithms, we first review the efficient nonmonotone line search of Zhang and Hager [15]. The earliest nonmonotone line search was developed in [3]; it permits some growth in the function value as the iteration process. Although these nonmonotone techniques work well in many cases, as pointed out in [15,16], there are some drawbacks. For instance, some good function values may be discarded; the numerical performance depends very much on the choice of a pre-fixed memory constant. To overcome these drawbacks, Zhang and Hager [15] proposed a new nonmonotone line search which requires that the average of the successive function values decreases. The reported numerical results showed that the new nonmonotone line search was superior to either the monotone or the traditional GLL line search [3].

Suppose that $d_k \in \mathbf{R}^n$ is a descent direction of f at x_k, i.e., $\nabla f(x_k)^T d_k < 0$. In the Zhang–Hager line search, the stepsize t_k satisfies the following Armijo-type condition:

$$f(x_k + t_k d_k) \le C_k + \rho t_k \nabla f(x_k)^T d_k, \tag{3.8}$$

where $\rho \in (0, 1)$, $C_0 = f(x_0)$, and C_k is updated by the following rules:

$$Q_{k+1} = \eta_k Q_k + 1, \qquad C_{k+1} = \frac{\eta_k Q_k C_k + f_{k+1}}{Q_{k+1}},$$

with $Q_0 = 1$ and $\eta_k \in [0, 1]$. This line search strategy makes the average of the successive function values decrease. The choice of η_k controls the degree of nonmonotonicity. In fact, if $\eta_k = 0$ for all k, then the line search is the usual GLL monotone line search [3]. As $\eta_k \to 1$, the line search becomes more nonmonotone, treating all the previous function values with equal weight when we compute C_k.

We now formally state the steps of the modified spectral gradient algorithm as follows.

Algorithm 3.1. **Step 0.** Given starting point x_0, constants $\lambda_{\max} \ge \lambda_{\min} \ge 0$, $\gamma \in (0, 1)$, $\eta_k \in [0, 1]$, $0 < \sigma_1 < \sigma_2 < 1$. Set $C_0 = f(x_0)$, $Q_0 = 1$, $\lambda_0 = 1$, and $k = 0$.

Step 1. Stop if $\|g_k\| = 0$.

Step 2. Set $d_k = -\lambda_k g_k$.

Step 3. Nonmonotone line search

Step 3.1. Set $t = 1$;

Step 3.2. If

$$f(x_k + t d_k) \le C_k + \gamma t g_k^T d_k, \tag{3.9}$$

then set $x_{k+1} = x_k + t d_k$. Go to Step 4. Else choose $t \in [\sigma_1 t, \sigma_2 t]$. Go to Step 3.2.

Step 4. Compute stepsize α_k. If $\alpha_k < 0$, set $\alpha_k = \lambda_{\max}$. Else set

$$\lambda_{k+1} = \min\{\lambda_{\max}, \max\{\lambda_{\min}, \alpha_k\}\}.$$

Step 5. Choose $\eta_k \in [\eta_{\min}, \eta_{\max}]$ and compute

$$Q_{k+1} = \eta_k Q_k + 1, \qquad C_{k+1} = \frac{\eta_k Q_k C_k + f_{k+1}}{Q_{k+1}}, \tag{3.10}$$

Step 6. Let $k = k + 1$. Go to Step 1.

Remarks. (1) The object in Step 4 is to avoid uphill directions and keep the sequence $\{\lambda_k\}$ uniformly bounded. In fact, for all k, $\lambda_{\min} \le \lambda_k \le \lambda_{\max}$. This also ensures that there exist positive numbers c_1 and c_2 such that the search direction d_k satisfies $g_k^T d_k \le -c_1 \|g_k\|^2$ and $\|d_k\| \le c_2 \|g_k\|$ for all k.

(2) The stepsize α_k in Step 4 can be chosen as any of those which have been mentioned above, and different choice reduce to different method. In particular, if α_k is chosen as (1.3) or (1.4), and the line search (3.9) is replaced by the GLL method, then Algorithm 3.1 is exactly the classical spectral gradient [4]. Moreover, for choice (3.1) or (3.2), and the GLL line search, Algorithm 3.1 reduces to the DYY method [10]. In brief, the remarkable difference between the above algorithm and other existing methods is the choice of line search step.

Now we turn to consider the convergence of Algorithm 3.1. Our convergence result utilizes the following assumptions.

Assumption 3.1. The level set $\Omega = \{x : f(x) \le f(x_0)\}$ is bounded.

Assumption 3.2. The gradient $g(x)$ of the objective function is Lipschitz continuous, i.e., there exists a constant $L > 0$ such that

$$\|g(x) - g(y)\| \le L\|x - y\|, \quad \forall\, x,\ y \in \mathbf{R}^n. \tag{3.11}$$

Let A_k be the average function value for k, i.e.,

$$A_k = \frac{1}{k+1} \sum_{i=0}^{k} f_i. \tag{3.12}$$

The following result shows that, for any choice of η_k, C_k lies between f_k and A_k. The proof of the following lemma can be found in [15, Lemma 1.1].

Lemma 3.1. *The iterates generated by Algorithm 3.1 satisfy $f_k \le C_k \le A_k$ for all k.*

The above result also indicates that, for each k, t_k can be chosen to satisfy the nonmonotone line search condition (3.9), which means that Step 3 is well defined.

Notice that there exist positive constants c_1 and c_2 such that

$$g_k^T d_k \le -c_1 \|g_k\|^2$$

and

$$\|d_k\| \le c_2 \|g_k\|$$

for all sufficiently large k; then we obtain our desirable convergence theorem of Algorithm 3.1. The proof of this theorem can be found in [15, Theorem 2.2].

Theorem 3.1. *Suppose that Assumptions 3.1–3.2 hold. Let $\{x_k\}$ be generated by Algorithm 3.1. Then either $\|g_k\| = 0$ for some finite k, or $\liminf_{k\to\infty} \|g_k\| = 0$. Furthermore, if $\eta_{\max} < 1$, then $\lim_{k\to\infty} \|g_k\| = 0$.*

4. Numerical experiments

In this section, we analyze the feasibility and effectiveness of Algorithm 3.1 with different parameters. The algorithm is implemented in Fortran77 code in double-precision arithmetic. All runs are performed on a PC (`Intel Pentium Dual E2140 1.6 GHz, 256 MB SDRAM`) with the Red Hat 9.03 Linux operation system. The algorithm stops if the maximum norm of the final gradient below 10^{-5}, that is,

$$\|\nabla f(x)\|_\infty \le 10^{-5}. \tag{4.1}$$

The process is also stopped if the number of iterations exceeds 10 000, or the number of function evaluations reaches 20 000. Our experiments are performed on the subset of the nonlinear unconstrained problems from the CUTEr [17] collection, and the second-order derivatives of all the selected problems are available. Since we are interested in large problems, we refined this selection by considering only problems where the number of variables is at least 50. Altogether, we solved 89 problems. The names and characters of these problems which were tested are listed in Table 4.1.

Table 4.1
Test problems and their character.

Problem	Character
argline, arglinb, arglinc, arwhead, bdqrtic, brownal, broydn7d, brybnd, chainwoo, cosine, cragglvy, curly, curly10, curly20, curly30, dixmaana, dixmaanb, dixmaanc, dixmaane, dixmaanf, dixmaang, dixmaanh, dixmaani, dixmaanj, dixmaanl, dixon3dq, edensch, eigenals, eigenbls, eigencls, engval1, errinros, extrosnb, fletcbv2, fletcbv3, fletchcr, freuroth, genhumps, genrose, indef, liarwhd, mancino, msqrtals, msqrtbls, ncb20b, noncvxu2, noncvxun, nondia, nondquar, nonmsqrt, penalty1, penalty2, power, quartc, sbrybnd, schmvett, scosine, scurly10, scurly20, scurly30, sensors, sparsine, sparsqur, spmsrtls, srosenbr, testquad, tointgss, tquartic, tridia, vardim, woods, eg2, jimack, dqrtic, powellsg, chnrosnb, dixmaand, dqdrtic, fletchbv, ncb20, penalty3, sinquad, vareigvl	Academic
fminsrf2, fminsurf, morebv, deconvu, tointgor, tointqor	Modelling

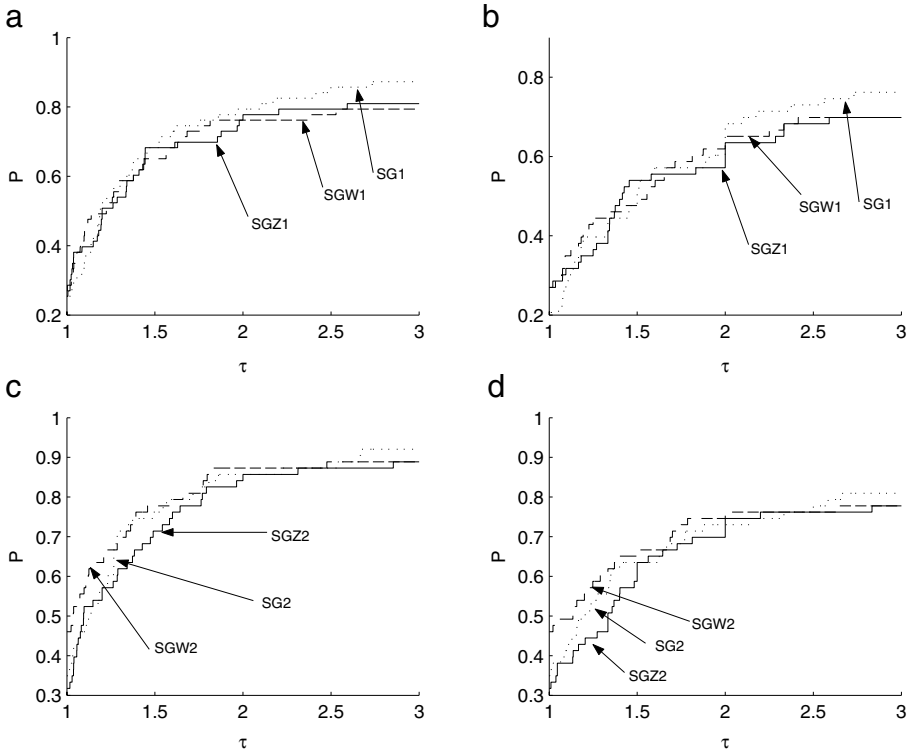

Fig. 1. Performance profiles.

For problems with variable dimension, we used the default dimension that is admissible in the "double large" installation of CUTEr.

We implemented Algorithm 3.1 with the following parameters: $\lambda_{\max} = 10^{30}$, $\lambda_{\min} = 10^{-30}$, $\gamma = 10^{-4}$, $\eta_k = 0.7$, $\sigma_1 = 0.1$, $\sigma_2 = 0.9$. These parameters were also used in [5,4], and it has been tested that the parameter settings are always of benefit to the performance of the algorithm. We implemented Algorithm 3.1 six times, using a different stepsize α_k every time. We named the algorithm SG1 if α_k was defined by (1.3), SG2 if α_k was defined by (1.4), SGW1 if α_k was defined by (3.2), SGW2 if α_k was defined by (3.7), SGZ1 if α_k was defined by (3.1), and SGZ2 if α_k was defined by (3.6), respectively. Since a large set of problems is used, we describe the results fully on the first author's homepage at the following web site: http://maths.henu.edu.cn/szdw/teachers/xyh.htm. Some general observations from these tables on the homepage include the following.

• Problems *curly, ncb20, ncb20b, nonmsqrt, penalty3, scurly30*, and *jimack* were excluded from our tables because they give the "insufficient space" error when evaluated by any tested algorithm.

• SG1 fails to satisfy the stopping criteria (4.1) in 23 problems, SG2 in 21 problems, SGW1 in 24 problems, SGW2 in 25 problems, SGZ1 in 28 problems, and SGZ2 in 26 problems.

• On 19 problems (*arglinc, curly10, curly20, curly30, dixon3dq, errinros, fletcbv3, fletchbv, genhumps, indef, msqrtals, noncvxu2, noncvxun, sbrybnd, scosine, scurly10, scurly20, sparssine, testquad*), each method cannot work successfully to achieve the stationary point based on the termination criteria (4.1).

Concerning the function values of the remaining 63 problems where at least one method runs successfully, we note that the differences of these functional values are pretty small. Therefore, it is reasonable to use the 63 problems to assess approximatively the performance of these methods. The performance of all methods is evaluated using the profile of Dolan and Moré (see [18]). We use the number of function evaluations and CPU time consumed as performance measures, since they reflect the main computational cost and the efficiency for each method. The performance profiles of these methods are plotted in Fig. 1. Fig. 1 contains the profiles of methods SG1, SGW1, and SGZ1 based on function evaluations (a) and CPU time (b), and the profiles of methods SG2, SGW2, and SGZ2 based on function evaluations (c) and CPU time (d).

Observing Fig. 1(a) and (b), respectively, one concludes that SG1 is always the top performer for most values of τ, which shows that SG1 performs better than SGW1 and SGZ1 did. The left-hand axis of each figure gives the percentage of the test problems for which method is a winner. Clearly, these three methods seem to have the same performance. However, Fig. 1(b) shows that SGW1 and SGZ1 are faster than SG1 by about 8%. When we turn our attention to Fig. 1(c) and (d), it is not clear which method is the winner. However, the left-hand side of both figures shows that SGW2 was superior to SG2 and SGZ2, which saves about 15% in function evaluations and CPU time consumed.

From the above numerical comparisons, we conclude that Algorithm 3.1 with α_k taken as (1.3) and (3.7) seems to performs better. Take everything together, we see that our proposed method provides an efficient approach for solving large-scale unconstrained optimization problems.

Acknowledgements

We would like to thank Professor Z. Wei for his constructive criticisms on this paper. We also thank the two anonymous referees for their careful reading and very useful comments, which improved this paper.

References

[1] J. Barzilai, J.M. Borwein, Two point step size gradient method, IMA J. Numer. Anal. 8 (1988) 141–148.
[2] L. Grippo, M. Sciandrone, Nonmonotone globalization techniques for the Barzilai–Borwein gradient method, Comput. Optim. Appl. 23 (2002) 134–169.
[3] L. Grippo, F. Lampariello, S. Lucidi, A nonmonotone line search technique for Newton's method, SIAM J. Numer. Anal. 23 (1986) 707–716.
[4] M. Raydan, The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem, SIAM J. Optim. 7 (1997) 26–33.
[5] E.G. Birgin, J.M. Martínez, M. Raydan, Nonmonotone spectral projected gradient methods on convex sets, SIAM J. Optim. 10 (2000) 1196–1211.
[6] Y. Xiao, Q. Hu, Subspace Barzilai–Borwein gradient method for large-scale bound constrained optimization, Appl. Math. Optim. 58 (2008) 275–290.
[7] R. Fletcher, On the Barzilai–Borwein method, Numerical Analysis Report NA/207, 2001.
[8] A. Friedlander, J.M. Martínez, B. Molina, Gradient method with restarts and generalizations, SIAM J. Numer. Anal. 36 (1999) 275–289.
[9] Y.H. Dai, W.W. Hager, K. Schittkowski, H. Zhang, The cyclic Barzilai–Borwein method for unconstrained optimization, IMA J. Numer. Anal. 26 (2006) 604–627.
[10] Y.H. Dai, J. Yuan, Y.X. Yuan, Modified two-point stepsize gradient method for unconstrained optimization, Comput. Optim. Appl. 22 (2002) 103–109.
[11] Z. Wei, G. Li, L. Qi, New quasi-Newton methods for unconstrained optimization problems, Appl. Math. Comput. 175 (2006) 1156–1188.
[12] Z. Wei, G. Yu, G. Yuan, Z. Lian, The superlinear convergence of a modified BFGS-type method for unconstrained optimization, Comput. Optim. Appl. 29 (2004) 315–332.
[13] J.Z. Zhang, N.Y. Deng, L.H. Chen, New quasi-Newton equation and related methods for unconstrained optimization, J. Optim. Theory Appl. 102 (1999) 147–167.
[14] Y. Xiao, Z. Wei, Z. Wang, A limited memory BFGS-type method for large-scale unconstrained optimization, Comput. Math. Appl. 56 (2008) 1001–1009.
[15] H. Zhang, W.W. Hager, A nonmonotone line search technique and its application to unconstrained optimization, SIAM J. Optim. 14 (2004) 1043–1056.
[16] Y.H. Dai, On the nonmonotone line search, J. Optim. Theory Appl. 112 (2002) 315–330.
[17] A.R. Conn, N.I.M. Gould, Ph.L. Toint, CUTE: constrained and unconstrained testing environment, ACM Trans. Math. Softw. 21 (1995) 123–160.
[18] E.D. Dolan, J.J. Moré, Benchmarking optimization software with performance profiles, Math. Program. 91 (2002) 201–213.