Computational Optimization and Applications, 22, 103–109, 2002

Modified Two-Point Stepsize Gradient Methods for Unconstrained Optimization*

YUHONG DAI dyh@lsec.cc.ac.cn
LSEC, Institute of Computational Mathematics, Academy of Mathematics and System Sciences, Chinese Academy of Sciences, Beijing 100080, People's Republic of China

JINYUN YUAN jin@mat.ufpr.br
Departamento de Matemática, Universidade Federal do Paraná, Centro Politécnico, CP: 19.081, CEP: 81531-990, Curitiba, Brazil

YA-XIANG YUAN yyx@lsec.cc.ac.cn
LSEC, Institute of Computational Mathematics, Academy of Mathematics and System Sciences, Chinese Academy of Sciences, Beijing 100080, People's Republic of China

Received; Revised ; Accepted

Abstract. For unconstrained optimization, the two-point stepsize gradient method is preferable over the classical steepest descent method both in theory and in real computations. In this paper we interpret the choice for the stepsize in the two-point stepsize gradient method from the angle of interpolation and propose two modified two-point stepsize gradient methods. The modified methods are globally convergent under some mild assumptions on the objective function. Numerical results are reported, which suggest that improvements have been achieved.

Keywords: unconstrained optimization, steepest descent method, two-point stepsize gradient method, nonmonotone line search

1. Introduction

Consider the unconstrained optimization problem

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

where f is smooth and its gradient is available. The gradient method for solving (1.1) is an iterative method of the form

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

where $g_k = \nabla f(x_k)$ and α_k is a stepsize.

It is well-known that the negative gradient direction has the following optimal property

$$-g_k = \min_{d \in R^n} \lim_{\alpha \to 0+} \left[f(x_k) - f\left(x_k + \alpha d / \|d\|_2^2\right)\right] / \alpha. \tag{1.3}$$

*Research was partly supported by Chinese NSF grants 19525101, 19731010, 19801033 and the Sino-Brazil joint research project "Numerical Linear Algebra and Optimization."

In the classical steepest descent method (see Cauchy [6]), the stepsize is obtained by carrying out an exact line search, namely,

$$\alpha_k = \arg\min_{\alpha} f(x_k - \alpha g_k). \tag{1.4}$$

However, despite the optimal properties (1.3) and (1.4), the steepest descent method converges slowly and is badly affected by ill conditioning (see Akaike [1] and Forsythe [9]).

Barzilai and Borwein [2] proposed two-point stepsize gradient methods by regarding $H_k = \alpha_k I$ as an approximation to the Hessian inverse of f at x_k and imposing some quasi-Newton property on H_k. Denote $s_{k-1} = x_k - x_{k-1}$ and $y_{k-1} = g_k - g_{k-1}$. By minimizing $\|H_k^{-1} s_{k-1} - y_{k-1}\|_2$, they obtained the following choice for the stepsize:

$$\alpha_k = s_{k-1}^T s_{k-1} \big/ s_{k-1}^T y_{k-1}. \tag{1.5}$$

If f is a strictly convex quadratic, Raydan [15] proved that the two-point stepsize gradient method (1.2) and (1.5) is globally convergent. If the dimension n is two, Barzilai and Borwein [2] established R-superlinear convergence result for the method and their analyses indicate that the convergence rate is faster as the matrix A is more ill conditioned. For the nonquadratic case, Raydan [16] incorporated a globalization scheme of the two-point stepsize gradient method using the technique of nonmonotone line search. The resulted algorithm is competitive and sometimes preferable to several famous conjugate gradient algorithms for large scale unconstrained optimization. Due to its simplicity and numerical efficiency, the two-point stepsize gradient method has received many studies, see [3–5, 7, 8, 10, 13, 15] etc.

In this paper we will view the choice (1.5) of the stepsize from the angle of interpolation. Along this line, two new choices of the stepsize are proposed in the next section. The corresponding modified two-point stepsize gradient algorithms are described in Section 3. The modified algorithms are globally convergent under some mild assumptions on the objective function. Numerical results are reported in Section 4, which suggest that improvements have been achieved.

2. Derivation of new stepsizes

For a one-dimensional optimization problem, the two-point stepsize gradient method (1.2) and (1.5) is the secant method. In the higher dimensional case, the formula (1.5) can be derived not only from quasi-Newton equation but also from interpolation. In fact, let α_k be defined by (1.5) and $t_k = \alpha_k^{-1}$, then

$$t_k = s_{k-1}^T y_{k-1} \big/ s_{k-1}^T s_{k-1}. \tag{2.1}$$

The quadratic model

$$q_k(\theta) = f_k + \theta g_k^T s_{k-1} + 0.5\, t\theta^2 \|s_{k-1}\|^2 \tag{2.2}$$

is an approximation to $f(x_k + \theta s_{k-1})$. For any $t \in R$, the above model satisfies interpolation conditions

$$q_k(0) = f_k \tag{2.3}$$

and

$$\nabla q_k(0) = g_k^T s_{k-1}. \tag{2.4}$$

It is easy to test that, if t is given by (2.1), the quadratic model (2.2) satisfies the interpolation condition

$$\nabla q_k(-1) = g_{k-1}^T s_{k-1}. \tag{2.5}$$

Therefore the stepsize α_k given by (1.5) is an approximation to the inverse of the second directional derivative of $f(x)$ at x_k along s_{k-1}.

If (2.5) is replaced with another interpolation condition

$$q_k(-1) = f_{k-1}, \tag{2.6}$$

then we can solve from (2.2) that

$$t_k = \big[2\big(f_{k-1} - f_k + g_k^T s_{k-1}\big)\big]\big/s_{k-1}^T s_{k-1}, \tag{2.7}$$

which yields

$$\alpha_k = s_{k-1}^T s_{k-1}\big/\big[2\big(f_{k-1} - f_k + g_k^T s_{k-1}\big)\big]. \tag{2.8}$$

In addition, to let conditions (2.5) and (2.6) be all satisfied, we can construct the following conic model,

$$c_k(\theta) = f_k + \theta g_k^T s_{k-1} + 0.5\, t_k \theta^2 s_{k-1}^T s_{k-1} + \xi_k \theta^3, \tag{2.9}$$

in which case we can solve that

$$t_k = \big[6(f_{k-1} - f_k) + 4 g_k^T s_{k-1} + 2 g_{k-1}^T s_{k-1}\big]\big/s_{k-1}^T s_{k-1} \tag{2.10}$$

and

$$\alpha_k = s_{k-1}^T s_{k-1}\big/\big[6(f_{k-1} - f_k) + 4 g_k^T s_{k-1} + 2 g_{k-1}^T s_{k-1}\big]. \tag{2.11}$$

It is easy to see that the formulae (2.8) and (2.11) are identical to (1.5) if $f(x)$ is quadratic on the line segment between x_{k-1} and x_k. This indicates that if the objective function $f(x)$ is a two-dimensional quadratic function, the method (1.2) with either (2.8) or (2.11) still has R-superlinear convergence.

For general nonlinear functions, since both the function value information and the gradient information at x_{k-1} and x_k are used in deriving (2.8) and (2.11), it is reasonable to expect that the two formulae will be better than (1.5). See Yuan [17, 18] for some one-dimensional evidences showing that the choices (2.8) and (2.11) of the stepsize prefer over (1.5). Specifically, as analyzed in [18], the local convergence rates of formulae (2.8) and

(2.11) for one-dimensional functions are faster than that of (1.5). We guess that Algorithms 3.1 and 3.2 also have R-superlinear convergence rate for a class of nonlinear functions, which is larger than the class of quadratic functions.

3. Modified two-point stepsize gradient algorithms

In this section, we present the algorithms corresponding to the formulae (2.8) and (2.11) with the technique of nonmonotone line search.

For convenience, we denote the stepsizes given by (1.5) and (2.8) as $\tilde{\alpha}_k$ and $\bar{\alpha}_k$ respectively. As mentioned in Section 2, we have $\tilde{\alpha}_k = \bar{\alpha}_k$ if the objective function $f(x)$ is quadratic on the line segment between x_{k-1} and x_k. If $f(x)$ is strictly convex, we always have that $0 \le \tilde{\alpha}_k \le 2\bar{\alpha}_k$. Since (2.8) is a formula with precisions higher than (1.5) ([18]), an integer variable j is used in our algorithms to decide whether the use of (2.8) is worthwhile. The value of j depends on the sequence $\{u_k\}$, where $u_k = |\frac{\tilde{\alpha}_k}{\bar{\alpha}_k} - 1|$ is a quantity showing how $f(x)$ is close to a quadratic on the line segment between x_{k-1} and x_k. More exactly, assume that $c_3 > c_2 > c_1$ are three positive constants. If $u_k \le c_1$, or $\max\{u_k, u_{k-1}\} \le c_2$, or $\max\{u_k, u_{k-1}, u_{k-2}\} \le c_3$, we believe that $f(x)$ is very close to a quadratic on the line segment between x_{k-1} and x_k, indicating the use of a formula with higher precisions. Thus in this case we set $j = 1$ and use the formula (2.8). Otherwise, we set $j = 0$ and use the formula (1.5). See Step 2 in Algorithm 3.1.

Now we state the detailed algorithm corresponding to the formula (2.8), in which the technique of nonmonotone line search of Grippo et al. [12] is used. The algorithm is a modification of the SPG2 algorithm in [5]. Here we should note that the SPG2 algorithm was designed for optimization with bound constraints. By setting the bounds to infinity, the SPG2 algorithm can solve unconstrained optimization problems.

Algorithm 3.1.

Step 0: Given $x_1 \in R^n$, $\alpha_1 = \|g_1\|_\infty^{-1}$, $M = 10$, $\gamma = 10^{-4}$, $\delta = 10^{30}$, $\epsilon \ge 0$,

$$c_1 = 5.0 * 10^{-4}, \quad c_2 = 0.1, \quad c_3 = 0.5. \quad \text{Set } k = 1.$$

Step 1: If $\|g_k\|_\infty \le \epsilon$, stop.

Step 2: (a) If $k = 1$, go to Step 3. If $s_{k-1}^T y_{k-1} \le 0$, $\alpha = \delta$, $u_k = 1$, go to Step 3.
(b) Calculate $\tilde{\alpha}_k$ and $\bar{\alpha}_k$ by (1.5) and (2.8) respectively; $u_k = |\frac{\tilde{\alpha}_k}{\bar{\alpha}_k} - 1|$.
(c) $j = 0$; if $u_k \le c_1$, $\max_{i=0,1} u_{k-i} \le c_2$ or $\max_{i=0,1,2} u_{k-i} \le c_3$, $j = 1$.
(d) If $j = 1$, $\alpha = \bar{\alpha}_k$; otherwise $\alpha = \max\{\delta^{-1}, \min\{\tilde{\alpha}_k, \delta\}\}$.

Step 3: (nonmonotone line search) If

$$f(x_k - \alpha g_k) \le \max_{0 \le i \le \min\{k-1, M\}} f_{k-i} - \gamma\alpha\|g_k\|_2^2,$$

then set $\alpha_k = \alpha$, $x_{k+1} = x_k - \alpha_k g_k$, $k = k + 1$, go to Step 1.

Step 4: Choose $\sigma \in [0.1, 0.9]$, set $\alpha = \sigma\alpha$, go to Step 3.

The denominator $\bar{\alpha}_k$ of the fraction in u_k may be zero. However, by (1.5) and (2.8), we have that

$$\frac{\tilde{\alpha}_k}{\bar{\alpha}_k} = \frac{2\big(f_{k-1} - f_k + g_k^T s_{k-1}\big)}{s_{k-1}^T y_{k-1}}, \tag{3.1}$$

which is well defined since we must have $s_{k-1}^T y_{k-1} > 0$ in Step 2(b). If $\bar{\alpha}_k$ is computed by (2.11) instead of (2.8) in Step 2, the resulting algorithm is called as Algorithm 3.2. Since $g_k^T s_{k-1} = s_{k-1}^T y_{k-1} - \alpha_{k-1}^{-1} s_{k-1}^T s_{k-1}$, we see that the required storage and computation amounts of Algorithms 3.1 and 3.2 are not much more than those of the SPG2 algorithm. They require to store only three n-dimensional vectors.

Suppose that the gradient ∇f is Lipschitz continuous. Then we have that

$$s_{k-1}^T y_{k-1} \le L s_{k-1}^T s_{k-1}, \tag{3.2}$$

where L is some positive constant. If $s_{k-1}^T y_{k-1} > 0$, we have by (1.5) and (3.2) that

$$\tilde{\alpha}_k \ge L^{-1}. \tag{3.3}$$

It follows from the above relation and Step 2 that, if $j = 1$, we must have that

$$\alpha = \bar{\alpha}_k \ge (1 + c_3)^{-1} \tilde{\alpha}_k \ge [(1 + c_3)L]^{-1}. \tag{3.4}$$

Otherwise, if $j = 0$ or the α is obtained by Step 2(a), we also have that $\alpha \ge \delta^{-1}$. By this and (3.4), we can similar to Theorem 2.1 in [16] prove the global convergence of Algorithms 3.1 and 3.2.

Theorem 3.1. *Suppose that x_1 is a starting point for which the level set $\mathcal{L} = \{x : f(x) \le f(x_1)\}$ is bounded and ∇f is Lipschitz continuous in some neighborhood $\mathcal{N}$ of $\mathcal{L}$. Consider the iterate $\{x_k\}$ generated by Algorithm* 3.1 (*or Algorithm* 3.2) *with $\epsilon = 0$. Then either $g_k = 0$ for some finite k or $\lim_{k\to\infty} g_k = 0$.*

4. Numerical results and conclusion

We have tested Algorithms 3.1 and 3.2 with double precisions in an SGI Indigo workstation. The code is based on the SPG2 algorithm in [5] and written with FORTRAN language. The test problems were taken from Morè et al. [14], except “Strictly Convex 1” and “Strictly Convex 2” that are provided in [16]. The total number of the test problems are 26. The stopping condition is

$$\|g_k\|_\infty \le 10^{-6}, \tag{4.1}$$

which is stronger than those normally used in real applications. The upper bound for the number of function evaluations is set to 9999.

Table 1. Numerical comparisons of gradient algorithms.

Problem	n	SPG2(I/F)	Algorithm 3.1 (I/F)	Algorithm 3.2 (I/F)
MGH11	3	949/2507	467/1224	926/2462
MGH14	4	163/329	163/329	163/329
MGH18	6	1091/2042	721/1373	660/1319
MGH22	16	466/776	430/750	356/609
MGH24	20	708/1939	407/1008	502/1239
	40	258/527	224/447	242/474
MGH28	20	907/923	907/923	907/923
	50	6967/7018	6967/7018	6967/7018
MGH30	50	38/39	38/39	38/39
	500	36/37	36/37	36/37
MGH31	50	30/31	30/31	30/31
	500	29/30	29/30	29/30
MGH22	100	272/468	249/437	392/711
	500	425/755	289/499	275/475
MGH25	100	1/2	1/2	1/2
	1000	1/2	1/2	1/2
MGH21	1000	53/279	52/184	34/45
	10000	53/279	52/184	34/45
MGH23	1000	56/251	56/251	56/251
	10000	64/163	64/163	64/163
MGH26	1000	89/205	89/205	89/205
	10000	83/107	83/107	83/107
Strictly convex 1	1000	5/6	5/6	5/6
	10000	5/6	5/6	5/6
Strictly convex 2	1000	533/786	367/540	431/642
	10000	2091/3205	1754/2592	1653/2653
Total CPU time (s)		114.15	97.95	96.78

We compared Algorithms 3.1 and 3.2 with the SPG2 algorithm. The numerical results are reported in Table 1, where the test problems from [14] are numbered in the following way: "MGH i" means the i-th problem in [14]. In addition, n denotes the dimension of the problem, and I, F are number of iterations and number of function evaluations respectively. The number of gradient evaluations is equal to that of iterations since no gradient evaluation is required in the line search procedure.

From Table 1, we see that Algorithms 3.1 and 3.2 require the same numbers of function evaluations and gradient evaluations as the SPG2 algorithm for some problems, whereas for the other problems, both algorithms perform better that the SPG2 algorithm (except for MGH22 with $n = 100$). The gains are sometimes significant, for example, for MGH18 with $n = 6$ and "Strictly Convex 2" with $n = 10000$. The total CPU times needed by Algorithms 3.1 and 3.2 respectively are less than that needed by the SPG2 algorithm. Therefore our numerical results suggest two efficient modified two-point stepsize gradient algorithms, which require few more storages and computations at every iteration but perform better than the SPG2 algorithm.

Acknowledgments

The authors would like to thank Dr. Marcos Raydan very much, who kindly sent us the code of the SPG2 algorithm in [5]. Thanks are also due to two anonymous referees, whose comments and suggestions greatly improved this paper.

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. J. Barzilai and J.M. Borwein, "Two point step size gradient methods," IMA J. Numer. Anal., vol. 8, pp. 141–148, 1988.
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., vol. 151, pp. 862–880, 1999.
4. E.G. Birgin and Y.G. Evtushenko, "Automatic differentiation and spectral projected gradient methods for optimal control problems," Optim. Methods Softw., vol. 10, pp. 125–146, 1998.
5. E.G. Birgin, J.M. Martínez, and M. Raydan, "Nonmonotone spectral projected gradient methods for convex sets," SIAM Journal on Optimization, vol. 10, no. 4, pp. 1196–1211, 2000.
6. A. Cauchy, "Méthode générale pour la résolution des systèms d'equations simultanées," Comp. Rend. Sci. Paris, vol. 25, pp. 46–89, 1847.
7. Y.H. Dai and L.Z. Liao, "R-Linear Convergence of the Barzilai and Borwein Gradient Method," Academy of Mathematics and Systems Sciences, Chinese Academy of Sciences, Research report AMSS-1999-081, 1999. Also in IMA J. Numer. Anal., accepted.
8. R. Fletcher, "Low storage methods for unconstrained optimization," Lectures in Applied Mathematics (AMS), vol. 26, pp. 165–179, 1999.
9. G.E. Forsythe, "On the asymptotic directions of the s-dimensional optimum gradient method," Numerische Mathematik, vol. 11, pp. 57–76, 1968.
10. 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.
11. W. Glunt, T.L. Hayden, and M. Raydan, "Molecular conformations from distance matrices," J. Comput. Chem., vol. 14, pp. 114–120, 1993.
12. 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.
13. W.B. Liu and Y.H. Dai, "Minimization Algorithms based on Supervisor and Searcher Co-operation. I:—Faster and robust gradient algorithms for minimization problems with stronger noises," Academy of Mathematics and Systems Sciences, Chinese Academy of Sciences, Research report AMSS-1999-085, 1999. Also in JOTA, accepted.
14. J.J. Morè, B.S. Garbow, and K.E. Hillstrom, "Testing unconstrained optimization software," ACM Transactions on Mathematical Software, vol. 7, pp. 17–41, 1981.
15. M. Raydan, "On the Barzilai and Borwein choice of steplength for the gradient method," IMA J. Numer. Anal., vol. 13, pp. 321–326, 1993.
16. M. Raydan, "The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem," SIAM J. Optim., vol. 7, no. 1, pp. 26–33, 1997.
17. Y. Yuan, "A modified BFGS algorithm for unconstrained optimization," IMA J. Numer. Anal., vol. 11, pp. 325–332, 1991.
18. Y. Yuan, "Numerical methods for nonlinear programming," Shanghai Scientific and Technical Publishers, 1993 (in Chinese).