Operations Research and Cybernetics

DOI:10.11721/cqnuj20130606

Numerical Performance of Subgradient Methods in Solving Nonsmooth Optimization Problems*

LONG Qiang[1], LI Jue-you[2]

(1. School of SITE, University of Ballarat, Victoria 3350, Australia

2. College of Mathematics, Chongqing Normal University, Chongqing 401331, China)

Abstract: The subgradient methods in solving nonsmooth optimization problems are studied in this paper. Firstly, we present a brief review of the available subgradient methods. Then, different sorts of step size rules are introduced and the corresponding convergence is established. Finally, some convex nonsmooth optimization problems are computed to test the numerical performance of provided subgradient methods. Numerical comparisons between different subgradient methods are investigated too.

Key words: subgradient methods; nonsmooth optimization problem; step size rules

中图分类号:O224 **文献标志码**:A **文章编号**:1672-6693(2013)06-0025-06

In this paper, we consider the following optimization problem(P)

$$(\mathrm{P})\begin{cases} min\ \mathrm{f(x)} \\ s.t.\ \ \mathrm{x}\in\mathbf{R}^n \end{cases},$$

where $f:\mathbf{R}^n\to\mathbf{R}$ is a Lipschitz continuous convex function. Note that f could be a nonsmooth function, and if it is, we call Problem (P) a nonsmooth optimization problem. The aim of this paper is to investigate the numerical performance of subgradient methods in solving nonsmooth optimization problems.

The first series of numerical approaches introduced to solve nonsmooth optimization problems are subgradient methods. They were mainly developed in the Soviet Union and an excellent overview can be found in Shor [1]. More discussion can refer to [2-8]. The idea of subgradient methods is directly extended from smooth optimization. They consider one opposite subgradient vector at the current iteration point as the next search direction, and apply step sizes which are supplied beforehand. This simple idea poses two ctritical questions: 1) How to choose step size rules; 2) When to stop the numerical implementable.

For nonsmooth optimization, an important issue is that the opposite direction of a subgradient at a point may not be a descent direction. So one cannot apply a line search method in the iteration as what we did in solving smooth optimization problems. The strategy of subgradient methods to handle this issue is to supply a step size rule beforehand. For example, a series of step sizes are supplied as $\{\alpha_k\}_{k=1}^{\infty}$, and the iteration style is as follows

$$\begin{cases} x_{k+1}=x_k+\alpha_k d_k \\ \alpha_k\in\{\alpha_k\}_{k=1}^{\infty} \end{cases} \tag{1}$$

where $d_k=-g_k\in\partial f(x_k)$ is an opposite direction of a subgradient at x_k. It is no doubt that the convergence rate of the algorithm closely depends on step sizes $\{\alpha_k\}_{k=1}^{\infty}$. Fortunately, for some particular step size rules, we can prove that iteration strategy (1) is convergence, although the convergence rate is quite slow.

* **Received**: 09-13-2012 **Accepted**: 03-08-2013

Foundation: Scientific and Technoloogical Resarch Program of Chongqing Municipal Education Commission (No. KJ120616)

The first author biograghy: LONG Qiang, male, doctor student, research area: Optimal theory and algorithm, E-mail: 27131375@qq.com

收稿日期:2012-09-13 **修回日期**:2013-03-08 **网络出版时间**:2013-11-20 14:46

资助项目:重庆市教委科学技术研究项目 (No. KJ20616)

作者简介:龙强,男,博士研究生,研究方向为最优化理论与算法、非光滑优化及全局最优化,E-mail:27131375@qq.com

网络出版地址:http://www.cnki.net/kcms/detail/50.1165.N.20131120.1446.201306.25_033.html

The rest of this paper is arranged as follows. In section 2, we will discuss some particular step size rules, and their convergence properties will be analysed. In section 3, a general structure of subgradient methods will be proposed. In section 4, some numerical examples will be tested by subgradient methods, andtheir numerical results will be analysed. Section 5 concludes this paper.

1 Convergence of subgradient methods

To proceed it further, we suppose that

Assumption 1 At least one subgradient of $f(x)$ at any point x can be calculated.

Assumption 2 The value of $f(x)$ is computable at any point.

We suppose that the general iteration style of subgradient methods is

$$\begin{cases} x_{k+1}=x_k-t_k\dfrac{\xi_k}{\|\xi_k\|},\xi_k\in\partial f(x_k) \\ t>0,\text{suitable} \end{cases} \tag{2}$$

The following theorem provides a sufficient condition for convergence of subgradient methods.

Theoren 1[2] Suppose that x_k is not optimal and x^* be an optimal solution of (P), then

$$\|x_{k+1}-x^*\|<\|x_k-x^*\| \tag{3}$$

whenever

$$0<t_k<2\frac{f(x)-f(x^*)}{\|\xi_k\|} \tag{4}$$

where $\xi_k\in\partial f(x_k)$.

Proof By the iteration rule (2), we have

$$\|x_{k+1}-x^*\|^2=\|x_k-t_k\frac{\xi_k}{\|\xi_k\|}-x^*\|^2=\|x_k-x^*\|^2-2\left\langle x_k-x^*,t_k\frac{\xi_k}{\|\xi_k\|}\right\rangle+t_k^2=\|x_k-x^*\|^2-2b_kt_k+t_k^2$$

where $b_k=\left\langle x_k-x^*,\dfrac{\xi_k}{\|\xi_k\|}\right\rangle$. It is easy to check that (3) holds when $0<t_k<2b_k$.

In light of the definition of subdifferential at x:$\partial f(x)=\{\xi\in\mathbf{R}^n\mid f(z)\geqslant\langle\xi,z-x\rangle+f(x)\}$.

Replacing x,ξ and z by x_k,ξ_k and x^*, respectively, we have $f(x^*)-f(x_k)\geqslant\langle x^*-x_k,\xi_k\rangle$, which yields $b_k\geqslant\dfrac{f(x_k)-f(x^*)}{\|\xi_k\|}$. This gives $0<t_k<2\dfrac{f(x_k)-f(x^*)}{\|\xi_k\|}$.

Remark 1 Theorem 1 can be interpreted by the definition of subgradient. If $\xi_k\in\partial f(x_k)$ and x^* is an optimal point, then we have $\langle-\xi^{\mathrm{T}},x^*-x_k\rangle\geqslant f(x_k)-f(x^*)\geqslant0$, which means that the angle between $-\xi_k$ and x^*-x_k is less than $\dfrac{\pi}{2}$. Hence, if $t_k>0$ is small enough, $x_{k+1}=x_k-t_k\dfrac{\xi_k}{\|\xi_k\|}$ is closer to x^* than x_k.

Remark 2 Although theorem 1 gives a domain of t_k, it is not applicable in numerical practice since we don't know the value of $f(x^*)$. Here it is worth to mention that, as long as t_k small enough, it can always satisfy the condition (4). And furthermore, if we do know the value of $f(x^*)$, (which is true for some nonsmooth optimization problems, such as nonsmooth equation), then a computable step size rule can be designed according to condition (4).

Given the discussion of Theorem 1, we provide four different step size rules as follows.

The simplest step size rule could be $t_k=\lambda$, with $\lambda>0$ small enough (5)

The choice of constant λ is a trick issue. A large λ may disobey condition (4), which makes the iteration stop at a fake optimal point; on the other hand, a small λ may increase iteration time. To be on the safe side, we tend to choose a small λ in numerical practice.

Now let us consider the step size rule

$$t_k>0,\lim_{k\to\infty}t_k=0,\sum_{k=1}^{\infty}t_k<\infty \tag{6}$$

Examples of this rule are $t_k=\dfrac{1}{k^2}$ and $t_k=t_0q^k$, with $t_0>0,q\in(0,1)$. One drawback of this step size rule is that the iteration always stays in a circle with a constant radius, say $A=\sum_{i=0}^{\infty}t_k$. This is because

$$\|x_0-x_k\|\leqslant\|x_0-x_1\|+\|x_1-x_2\|+\cdots+\|x_{k-1}-x_k\|=t_0+t_1+t_2+\cdots+t_{k-1}\leqslant A$$

To overcome this drawback, we adjust the step size rule by

$$t_k > 0, \sum_{k=1}^{\infty} t_k = \infty, \sum_{k=1}^{\infty} t_k^2 < \infty \tag{7}$$

For example, $t_k = \frac{a}{b+k}$ with $a>0$ and $b\geqslant 0$, respectively. In this way, some optimal x^*, even far from the starting point x_0 will be reached.

Finally, we consider a special situation in which the optimal value of f, say f^{**}, is known. The step size rule in this situation is $t_k = \lambda \frac{f(x_k) - f^*}{\| g_k \|}$ with $\lambda \in [0,2]$ (8)

This computable step size, which incorporates the knowledge of f, x_k and f^*, guarantees a monotonic decrease of the distance from x_k to x^* (for all k) and actually the whole sequence $\{x_k\}$ does converge to some $x^* \in X^*$.

2 Algorithms of subgradient methods

All the different subgradient methods have the same operation process, except they use different step size rules. Therefore, in this section, we first provide the general structure of subgradient methods and then apply it to different step size rules.

Algorithm General Subgradient Method(GSM)

Step 1: Initialization. Set x_0 as a starting point and let $x^* = x_0$, $f^* = f(x^*)$. Set $m=10\ 000$ as the largest iteration time. Set $k=1$ and a tolerance parameter $\delta=0.000\ 1$. Choose a step size rule from (5)～(8).

Step 2: Store f^* and x^*. If $f^* > f(x_k)$, then let $f^* = f(x_k)$ and $x^* = x_k$.

Step 3: Iteration. Compute the next step size t_k using the selected step size rule in step 1 and calculate one subgradient $\xi_k \in \partial f(x_k)$. Then calculate the next point $x_{k+1} = x_k - t_k \frac{\xi_k}{\| \xi_k \|}$.

Step 4: Stop criteria. If $\| \xi_k \| < \delta$, then stop and output f^* and x^*; else if $k>m$, then stop and output f^* and x^*; otherwise, set $k=k+1$ and go to step 2.

Remark 3　Except using the well-known necessary optimal condition $0 \in \partial f(x_k)$ as a stop criteria, we still restrict the largest iteration time as a coercive stop criteria. This is because, for some special nonsmooth optimization problems, the condition $0 \in \partial f(x_k)$ is impossible to satisfy. A convictive example could be $f(x) = |x|$. No matter how close x_k to the single optimal point 0, the norm of subgradient at x_k is $\| \xi_k \| = 1$.

Remark 4　The reason for introducing step 2 is because subgradient methods are not descent methods, so the final point we obtained by subgradient methods may close to one optimal point x^*, but the function value of the point may not the smallest one. Therefore, we chose the one with the smallest function value as the result of subgradient methods. For the special situation that f^* is already known, we cancel step 2.

Remark 5　To avoid iteration points staying in a circle with constant radius, we can reset step size t_k at a suitable time. For example, when $\| x_{k+1} - x_k \| < \gamma$ (a small positive number), we set $t_k = t_0$.

3 Numerical tests

In this section, we consider some test problems to investigate the numerical performance of subgradient methods. Note that it is the step size rules that make subgradient methods different, so we firstly test step size rules proposed in (5)～(8) individually. Then, numerical comparisons among those step size rules are presented. All the test problems are cited from [3] and computed on an ACER 4730Z laptop with 2 GB RAM and 2.16 GHz CPU in Matlab 2010 environment.

First of all, we illustrate some signs which are used in the following: x_0 is starting point for numerical tests; t_k is step size for the k^{th} iteration step; x^* is approximation optimal point obtained by subgradient methods; f^* is approximation optimal value obtained by subgradient methods; f^{**} is the already known best optimal value.

Example 1[3]　$\min \max_{1\leqslant i\leqslant 3} f_i(x)$

$$f_1(x) = x_1^2 + x_2^4, f_2(x) = (2-x_1)^2 + (2-x_2)^2, f_3(x) = 2\exp(x_2 - x_1), x_0 = (0,0)$$

Example 1 is solved by applying constant step size rule (5). The current known best optimal solution is $f^{**}=1.952\ 224\ 5$. From Tab. 1, which illustrates numerical results, we can see that the smaller the step size the better optimal value we can get. This result agrees with the analysis we did in section 2.

Example 2[3] $\min \max\limits_{1\leqslant i\leqslant 3} f_i(x)$

$$f_1(x)=\frac{1}{2}\left(x_1+\frac{10x_1}{x_1+0.1}+2x_2^2\right), f_2(x)=\frac{1}{2}\left(-x_1+\frac{10x_1}{x_1+0.1}+2x_2^2\right) f_3(x)=\frac{1}{2}\left(x_1-\frac{10x_1}{x_1+0.1}+2x_2^2\right), x_0=(3,1)$$

The computation of example 2 applies step size rule (6). The optimal solution of this example is $f^{**}=0$. In Tab. 2, $d_{\max}$ stand for the farthest distance between the initial point x_0 and the iteration points x_k, i. e., $d_{\max}=\max\{\|x_0-x_k\|\ |k=1,\cdots,m\}$.

From Tab. 2, we can obviously see that the first two step size are failed to find an optimal solution. This is because all the iteration point x_k stay in a circle around x_0. Check the column of $d_{\max}$, we have $d_{\max}<\sum\limits_{i=1}^{\infty} t_k$.

Tab. 1 Numerical result for example 1

t_k	x^*	f^*
1	(1.171 5, 0.284 3)	3.629 938 378 226 757
0.5	(1.220 3, 0.939 7)	2.268 832 647 147 855
0.1	(1.145 7, 0.902 8)	1.976 789 858 737 801
0.05	(1.139 7, 0.899 9)	1.954 604 289 964 668
0.01	(1.137 3, 0.893 8)	1.967 839 355 034 972
0.005	(1.140 5, 0.897 8)	1.953 339 146 087 326
0.001	(1.139 4, 0.899 7)	1.953 463 252 515 693

Example 3[3] $\min \max\{f_1(x), f_2(x)\}$

$$f_1(x)=\left(x_1-\sqrt{x_1^2+x_2^2}\cos\sqrt{x_1^2+x_2^2}\right)^2+0.005(x_1^2+x_2^2)$$

$$f_2(x)=\left(x_2-\sqrt{x_1^2+x_2^2}\sin\sqrt{x_1^2+x_2^2}\right)^2+0.005(x_1^2+x_2^2), x_0=(1,4)$$

Step size rule (7) is used in solving example 3. The optimal solution of this problem is $f^{**}=0$. Tab. 3 shows numerical results for solving example 3, where iter time stands for the iteration time when the approximation optimal solution f^* is achieved. All the step size rule get an acceptable optimal solution. The item of iter time shows that the optimal solution is not obtained in the last iteration, which proves that subgradient methods are not descent methods.

Tab. 2 Numerical result for example 2

t_k	x^*	f^*	$d_{\max}$
$2(1/2)^k$	$\begin{pmatrix}1.739\ 0\\-0.001\ 6\end{pmatrix}$	5.597 6	1.610 3
$2(2/3)^k$	$\begin{pmatrix}0.457\ 5\\0.001\ 5\end{pmatrix}$	4.331 8	2.731 5
$2(3/4)^k$	$\begin{pmatrix}-0.000\ 0\\-0.001\ 2\end{pmatrix}$	1.557 0e−006	3.217 5
$(4/5)^k$	$10^{-3}\times\begin{pmatrix}0.000\ 0\\-0.196\ 7\end{pmatrix}$	3.870 9e−008	3.180 9

Example 4[3] $\min \max\limits_{1\leqslant i\leqslant 6} f_i(x)$

$$f_1(x)=x_1^2+x_2^2+x_3^2-1, f_2(x)=x_1^2+x_2^2+(x_3-2)^2, f_3(x)=x_1+x_2+x_3+1$$

$$f_4(x)=x_1+x_2-x_3+1, f_5(x)=2x_1^3+6x_2^2+2(5x_3-x_1+1)^2, f_6(x)=x_1^2-9x_3$$

$$x_0=(1,1,1)$$

Be solving example 4, we use the step size rule when f^{**} is already known. From reference paper [3], we know that for example 4, $f^{**}=3.599\ 719\ 3$, so the step size rule is written as $t_k=\lambda\dfrac{f(x_k)-f^{**}}{\|\xi_k\|}$, $\lambda\in[0,2]$, $\xi_k\in\partial f(x_k)$.

The first column of tab. 4 listed the choice of λ. gap in the last column illustrates the difference between the obtained approximation optimal value f^* and the already known optimal value f^{**}, i. e. $gap=f^*-f^{**}$.

Tab. 3 Numerical results for example 3

t_k	x^*	f^*	iter time
$1/k$	$\begin{pmatrix}0.406\ 5\\1.165\ 2\end{pmatrix}$	0.007 616 536 232 980	226
$1/(1+k)$	$\begin{pmatrix}0.548\ 4\\0.530\ 3\end{pmatrix}$	0.002 920 361 053 007	580
$2/(2+k)$	$\begin{pmatrix}0.548\ 4\\0.531\ 3\end{pmatrix}$	0.002 926 270 308 370	702
$4/(4+k)$	$\begin{pmatrix}0.548\ 6\\0.531\ 5\end{pmatrix}$	0.002 927 465 964 806	903
$8/(8+k)$	$\begin{pmatrix}0.548\ 3\\0.531\ 6\end{pmatrix}$	0.002 928 175 294 248	223 9

Example 5[3] $\min \max\limits_{1\leqslant i\leqslant 4} f_i(x)$

$$f_1(x)=x_1^2+x_2^2+2x-362+x-4^2-5x$$

$$-1-5x-2-21x_3+7x_4$$

$$f_2(x)=f_1(x)+10(x_1^2+x_2^2+x_3^2+x_4^2+$$

$$x_1-x-2+x_3-x_4-8)$$

$$f_3(x)=f_1(x)+10(x_1^2+x_2^2+x_3^2+2x_4^2-$$

$$x_1-x_4-10)$$

$$f_4(x)=f_1(x)+10(2x_1^2+x_2^2+x_3^2+2x_1-$$

$$x_2-x_4-5), x_0=(0,0,0,0)$$

Example 5 compares all the step size rules presented in (5)～(8). The already known best optimal value of this problem is $f^{**}=-44$. From the *gap* listed in the last column of tab. 5, we can see that step size rule (7) and (8) tend to experience better results than step size rule (5) and (6).

Tab. 4　Numerical results for example 4

λ	x^*	f^*	*gap*
0.5	$\begin{pmatrix}0.345\ 8\\0.005\ 0\\0.134\ 4\end{pmatrix}$	3.600 138 583 736 212	4.192 8e−004
1	$\begin{pmatrix}0.337\ 0\\0.001\ 8\\0.132\ 9\end{pmatrix}$	3.599 827 401 482 616	1.081 0e−004
1.5	$\begin{pmatrix}0.333\ 4\\0.000\ 9\\0.132\ 2\end{pmatrix}$	3.599 792 602 333 767	7.330 2e−005
2	$\begin{pmatrix}0.328\ 3\\0.000\ 0\\0.131\ 3\end{pmatrix}$	3.599 941 107 606 238	1.218 0e−004

Tab. 5　Numerical result of example 5

λ	x^*	f^*	*gap*
0.1(5)	$\begin{pmatrix}-0.006\ 8\\1.002\ 0\\2.007\ 5\\-0.981\ 5\end{pmatrix}$	−43.976 316 790 414 08	0.023 6
0.01(5)	$\begin{pmatrix}-0.001\ 3\\1.000\ 0\\2.000\ 9\\-0.998\ 9\end{pmatrix}$	−43.999 735 615 518 411	2.643 8e−004
2/(2+k)(7)	$\begin{pmatrix}-0.000\ 0\\0.999\ 2\\2.000\ 3\\-0.999\ 8\end{pmatrix}$	−43.999 996 735 850 772	3.264 1e−006
5/(5+k)(7)	$\begin{pmatrix}-0.000\ 0\\1.000\ 0\\2.000\ 0\\-0.999\ 9\end{pmatrix}$	−43.999 995 720 274 626	4.279 7e−006
2(2/3)^k(6)	$\begin{pmatrix}0.064\ 3\\0.738\ 9\\2.043\ 5\\-0.880\ 0\end{pmatrix}$	−43.413 771 619 036 787	0.586 2
2(4/5)^k(6)	$\begin{pmatrix}-0.029\ 6\\0.929\ 6\\2.041\ 2\\-0.956\ 2\end{pmatrix}$	−43.947 001 474 082 462	0.053 0
$\frac{f(x_k)-f^*}{\|\xi_k\|}$(8)	$\begin{pmatrix}-0.000\ 0\\0.991\ 8\\2.002\ 3\\-0.998\ 8\end{pmatrix}$	−43.999 668 922 006 904	3.310 8e−004
1.5 $\frac{f(x_k)-f^*}{\|\xi_k\|}$(8)	$\begin{pmatrix}-0.000\ 6\\0.995\ 5\\2.001\ 7\\-0.998\ 9\end{pmatrix}$	−43.999 823 930 916 833	1.760 6e−004

4 Conclusion

This paper investigated the numerical performance of different subgradient methods. We firstly introduced

different styles of step size rules and analysed their convergence properties. And then numerical tests were made to test numerical performance of different step size rules. The numerical results show that subgradient methods are feasible for some unconstrained nonsmoothconvex optimization problems. However, how to choose the step size rule is a difficult issue. From the numerical results, step size rule (7) should be the first choise.

References:

[1] Shor N Z, Kiwiel K C, Ruszczynski A. Minimization methods for non-differentiable functions[M]. Berlin: Springer-verlag, 1985.

[2] Lemarechal C. Nondifferentiable optimization[C]//Nemhauser G L, Rinnooy Kan A H G, Todd M J. Handbooks in Operations Research and Management Science, Optimization. Amsterdam, Netherlands: North Holland, 1989.

[3] Lukšan L, Vlček J. Test problems for nonsmooth unconstrained and linealy constrained optimization[R]. Institute of Computer Science, Academy of Science of the Czech Republic, 2000.

[4] Polyak B T. A general method for solving extremal problem [J]. Dokl Akad Mauk SSSR, 1967, 174: 33-36.

[5] Ermolev Y M. Methods of solution of nonlinear extremal problems[J]. Cybernetics and Systems Analysis, 1966, 2 (2): 1-14.

[6] Polyak B T. Subgradient methods: a survey of soviet union research[C]//Lemarechal C, Mifflin R. Nonsmooth Optimization. Oxford: Pergamon Press, 1977.

[7] Polyak B T. Introduction to optimization[M]. New York: Optimization Software Inc Publications Division, 1987.

运筹学与控制论

次梯度法在求解非光滑最优化问题时的计算效果研究

龙 强[1], 李觉友[2]

(1. 澳大利亚巴拉瑞特大学 科学、信息技术和工程学院, 维多利亚 澳大利亚 3350
2. 重庆师范大学 数学学院, 重庆 401331)

摘要: 本文研究了次梯度法的一些重要问题。次梯度法是梯度法在非光滑优化中的直接推广。在每一步的迭代中, 选取一个负次梯度方向为搜索方向, 并以一定的规则设置搜索步长。次梯度法的每一步迭代不一定都下降, 但是可以证明, 对于非光滑凸优化问题, 次梯度法能够保证全局收敛性。次梯度法的搜索步长是预先设置的, 步长设置准则包括常值步长准则、有限平方和步长准则和已知全局极小值的步长准则。本文对各种步长准则的收敛性进行了证明。为了验证次梯度法在不同的步长准则下的计算效果, 本文应用次梯度法对一系列非光滑最优化问题进行了计算实验, 并分析了他们的计算结果。数值实验结果表明, 常值步长准则收敛速度慢, 精度不高, 而且步长的选择困难。而有限平方和步长准则收敛速度更快, 也能够达到更高的精度。至于已知全局极小值的步长准则, 虽然精度也较高, 但是因为需要事先已知凸优化问题的全局极小值, 所以这种步长准则的应用范围有限。

关键词: 次梯度法; 非光滑最优化问题; 步长准则

(责任编辑 黄 颖)