Journal of Engineering and Applied Sciences 13 (Special Issue 6): 5442-5445, 2018
ISSN: 1816-949X

A New Scaled Steepest Descent Method for Unconstrained Optimization with Global Convergence Properties

[1]Rashidah Johari, [2]Mohd Rivaie and [1]Mustafa Mamat
[1]Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin (UniSZA), Terengganu, Malaysia
[2]Department of Computer Sciences and Mathematics, Universiti Teknologi Mara (UiTM), Terengganu, Malaysia

Abstract: The steepest descent method is the simplest gradient method for solving unconstrained optimization problems. In this study, a new scaled search direction of steepest descent method is proposed. The proposed method is motivated by Andrei's approach of scaled conjugate gradient method. The numerical results show that the proposed method outperforms than the other classical steepest descent method.

Key words: Andrei's, optimization, motivated, outperforms, classical steepest, numerical results

INTRODUCTION

The Steepest Descent (SD) method is a well-known optimization method for solving unconstrained optimization problems. The SD method has low computational cost and matrix storage requirement considering that it does not need the computation of second derivatives to be solved to compute the search direction. Generally, the unconstrained optimization problem to be minimized is considered below:

$$\min_{x \in R^n} f(x) \quad (1)$$

Where:

$f: R_n \rightarrow R$ = Continuously differentiable function. Suppose that an initial point
x_0 = Selected, then the next iterate point
x_{k+1} = That approaches a solution point is obtained by the iterative Eq:

$$x_{k+1} = x_k + \alpha_k d_k \quad (2)$$

Where:

x_k = The current iterate point
a_k = A positive step-size
d_k = The search direction

The search direction is in the opposite of gradient:

$$d_k = -g(x_k) = -\nabla f(x_k) \quad (3)$$

The SD method was first introduced by Cauchy (1847) who proposed the use of negative gradient direction to find the local minimizers of a differentiable function. The SD method produces successive directions that are orthogonal to each other. However, the choice of exact line search with the direction of SD may cause the algorithm to zigzag when the point is near to the optimum (Abidin *et al*., 2014; Nocedal and Wright, 2006). Thus, the convergence speed of SD becomes very slow. Abidin *et al*. (2014) presented a new search direction from Cauchy's method in the form of two parameters which is known as Zubai'ah-Mustafa-Rivaie-Ismail (ZMRI) method:

$$d_k^{ZMRI} = -g_k - \| g_k \| g_{k-1} \quad (4)$$

It is known that the choices of step-size and search direction affect numerical performance of gradient method (Nocedal and Wright, 2006). The stepsize can be obtained by exact line search:

$$\alpha_k = \min_{\alpha>0} f\left(x_k + \alpha_k(-g_k)\right) \quad (5)$$

Barzilai and Borwein (1988) presented a new formula of step-size through two-point step-size for gradient method. Interestingly, the Barzilai-Borwein (BB) method converges with R-superlinear convergence for two-dimensional strictly convex quadratic functions. Due

Corresponding Author: Rashidah Johari, Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin (UniSZA), Terengganu, Malaysia

to its simplicity and numerical efficiency, the BB method has attracted the attention of many researchers on the choices of the step-size such as Raydan (1993a, b), Dai and Liao (2002), Mamat *et al* (2009) and many others. Wen *et al*. (2012) stated that only exact line search can give the greatest possible reduction to the objective function even though several inexact line searches have been proposed. Therefore, we use exact line search as the line search procedure in this study.

MATERIALS AND METHODS

New scaled search direction: In this study, a new scaled Search Direction of SD method is proposed. This modification is inspired from Andrei (2007)'s approach of scaled conjugate gradient method into the search direction of ZMRI. The proposed method is named as where RRM denotes Rashidah, Rivaie and Mamat:

$$d_k = \begin{cases} -g_k & \text{if } k = 0, \\ -\theta g_k - \| g_k \| g_{k-1} & \text{if } k \geq 1 \end{cases} \quad (6)$$

where, θ_k (Zhang *et al*., 2006) is a scaling parameter to be:

$$\theta_k = \frac{d_{k-1}^T y_{k-1}}{\|g_{k-1}\|^2} \quad (7)$$

Where:

$$y_{k-1} = g_k - g_{k-1}$$

Algorithm of SD^{RRM} method 1:

Step 1: Given initial point, x_0. Set k = 0

Step 2: Compute the search direction by Eq.6
If $d_k = 0$, then stop. Declare x_k as a stationary point

Step 3: Compute the step-size a_k by exact line search, based on Eq. 5

Step 4: Update the next point x_{k+1} by Eq. 2

Step 5: Check for convergence and stopping criteria
If $f(x_{k+1})<f(x_k)$ and $\|g_k\| \leq \varepsilon$ where $\varepsilon = 10^{-6}$ satisfied, then terminate
Otherwise, consider, k = k+1 continue by Step 2

RESULTS AND DISCUSSION

Numerical results and discussion: In this study, the comparison results between RRM with ZMRI and SD by using standard test functions (Andrei, 2008; Jamil and Yang, 2013; Witte and Hoist, 1964) are observed. Four different initial points are chosen for each test functions to study the global convergence properties of the proposed method. The initial points are assigned from the closer initial point to the further away from the solution point. The stopping criteria are set to $\|g_k\| \leq 10^{-6}$. All problems in Table 1 and 2 are solved by using Maple 18 programming. Figure 1 and 2 show the performance results based on the performance profile introduced by Dolan and More (2002).

Table 1: Test function comparisons based on number of iterations

		No. of iterations		
Function	Initial point	SD	ZMRI	RRM
Shalow	(-10, -10)	109	31	37
	(1.5, 1.5)	19	18	20
	(2, 2)	79	16	20
	(10, 10)	121	27	25
Rosenbrock	(-2, 2)	>1000	101	36
	(0.01, 0.01)	>1000	175	151
	(0.3, 1.6)	>1000	137	92
	(5, 5)	>1000	295	67
Himmelblau	(5,5)	19	22	10
	(10, 10)	17	17	13
	(15, 15)	17	15	13
	(20, 20)	17	13	13
3 hump camel	(0.35, 0.35)	16	13	12
	(0.65, 0.65)	14	11	9
	(0.85, 0.85)	5	13	9
	(2, -2)	9	13	12
6 hump camel	(0.15, 0.15)	12	10	8
	(0.45, 0.45)	8	6	5
	(3, 3)	12	14	10
	(4, 4)	13	12	10
Quartic	(2, 2)	19	14	11
	(4, 4)	19	15	11
	(6, 6)	19	15	11
	(8, 8)	19	15	11
Zettl	(2, 2)	136	42	36
	(3, 3)	160	38	36
	(4, 4)	168	38	36
	(5, 5)	170	38	36
Engvall	(3, 3)	19	15	11
	(6, 6)	19	15	11
	(15, 15)	19	13	11
	(20, 20)	19	15	11
Liarwhd	(7, 7)	20	17	15
	(15, 15)	16	19	13
	(17, 17)	16	19	16
	(20, 20)	16	19	18
Goldstein-Price's	(5, 5)	211	19	18
	(7, 7)	58	26	25
	(11,11)	321	36	34
	(24, 24)	>1000	43	29
Cube	(-0.001, 0.001)	>1000	Fail	57
	(-0.25, 0.25)	>1000	54	49
	(-0.5, 0.5)	>1000	249	71
	(-5, 5)	>1000	275	60
Beale's	(3, 3)	599	55	21
	(5, 0.5)	33	79	31
	(9, 3)	>1000	43	37
	(11, 0.5)	97	100	82
Price	(0.15, 0.15)	115	50	50
	(0.55, 0.55)	>1000	46	7
	(14, 14)	>1000	Fail	25
	(27.5, 27.5)	39	Fail	5
Nonscomp	(1.25,1.25)	25	22	20
	(6, 6)	341	30	22
	(9, 9)	437	198	35
	(13, 13)	535	31	20

Table 2: Test function comparisons based on CPU time

Function	Initial point	CPU time (sec) SD	ZMRI	RRM
Shalow	(-10, -10)	12.1525	4.8672	5.8656
	(1.5, 1.5)	3.6348	3.8064	3.9936
	(2, 2)	5.2728	3.3384	3.7440
	(10, 10)	14.0713	5.1168	4.4928
Rosenbrock	(-2, 2)	NA	11.2477	5.1012
	(0.01, 0.01)	NA	18.6889	16.7857
	(0.3, 1.6)	NA	14.6953	10.1869
	(5, 5)	NA	30.4358	8.2213
Himmelblau	(5, 5)	2.9484	3.3540	2.4336
	(10, 10)	2.8704	2.8548	2.7456
	(15, 15)	2.7456	3.0732	2.6832
	(20, 20)	2.7144	2.8392	2.7612
3 hump camel	(0.35, 0.35)	5.4600	3.6504	3.6504
	(0.65, 0.65)	3.7128	3.4008	2.9328
	(0.85, 0.85)	2.2620	3.6348	2.7456
	(2, -2)	3.0888	3.8220	3.3072
6 hump camel	(0.15, 0.15)	4.0716	3.2448	3.1824
	(0.45, 0.45)	2.5740	2.4804	2.4336
	(3, 3)	3.5100	4.0716	3.3540
	(4, 4)	3.6660	3.6660	3.4788
Quartic	(2, 2)	3.5412	2.9640	2.6676
	(4, 4)	3.4788	3.2448	2.8704
	(6, 6)	3.3540	3.6036	3.0420
	(8, 8)	3.7596	3.0576	2.7300
Zettl	(2, 2)	13.8841	5.1480	4.9452
	(3, 3)	15.2101	4.6800	4.9920
	(4, 4)	17.1289	5.0544	4.8360
	(5, 5)	17.2693	4.9764	5.3040
Engval1	(3, 3)	4.8048	3.0264	2.6052
	(6, 6)	3.5256	3.3696	2.7456
	(15, 15)	3.3852	3.2604	2.8080
	(20, 20)	3.6036	3.0888	2.6832
Liarwhd	(7, 7)	3.4008	3.3228	3.0420
	(15, 15)	3.1200	3.5412	2.8704
	(17, 17)	3.2604	3.5100	3.2760
	(20, 20)	2.8392	3.2604	3.1668
Goldstein-Price's	(5, 5)	52.2135	6.2868	6.2868
	(7, 7)	14.0089	7.2540	8.0029
	(11, 11)	83.1485	11.0605	9.6721
	(24, 24)	NA	12.7921	9.6409
Cube	(-0.001, 0.001)	NA	Fail	13.8373
	(-0.25, 0.25)	NA	10.8109	11.0761
	(-0.5, 0.5)	NA	46.1607	15.0073
	(-5, 5)	NA	53.6643	12.9793
Beale's	(3, 3)	129.8396	14.7265	6.5052
	(5, 0.5)	9.2353	22.7761	8.5801
	(9, 3)	NA	11.9185	10.9981
	(11, 0.5)	24.0242	23.9930	21.4345
Price	(0.15, 0.15)	36.1454	15.2413	15.7093
	(0.55, 0.55)	NA	13.6345	3.6036
	(14, 14)	NA	Fail	8.6737
	(27.5, 27.5)	11.3725	Fail	2.9484
Nonscomp	(1.25, 1.25)	6.1152	3.9312	3.8844
	(6, 6)	36.8474	4.8828	4.3212
	(9, 9)	47.5491	22.7761	5.8656
	(13, 13)	55.1932	4.8048	3.7596

Table 1, the iteration numbers in the SD column of Rosenbrock, Goldstein-Price's, Cube, Price and Beale's functions are written as larger than 1000. This is because the value is set as the maximum limit of accepted number of iterations. So, when the number count exceeds 1000, the iteration process is terminated. In Table 2, the symbol 'NA' in the SD column indicates that the CPU time is not available because the iteration number reaches the set limit at 1000. The word 'Fail' in Table 1 and 2 represents that the step-size is negative. In order to compare the performance of RRM with ZMRI and SD, we plot the fraction P of problems for which the method is within a factor t of the best time for Fig. 1 and the minimum number of iterations for Fig. 2. The left side of Fig. 1 and 2 give the percentage of the test problems for which a method is the fastest while the right side gives the percentage of the test problems that are successfully solved by each of the methods. The top curve is the method that solves the most problems in a time that was within a factor t. From Fig. 1, we can say that RRM is the fastest solver for about 76.79% of the test problems whereas ZMRI and SD only at 16.07% and 10.71%, respectively. RRM solves 100% of the entire problem while ZMRI and SD can solve 94.64% and 78.57% of the problems respectively. The left side of Fig. 2 shows that RRM is the fastest which solves about

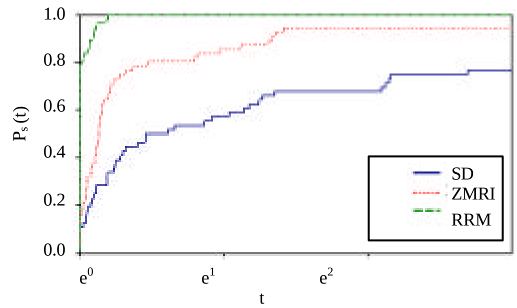

Fig. 1: Performance profile based on CPU time

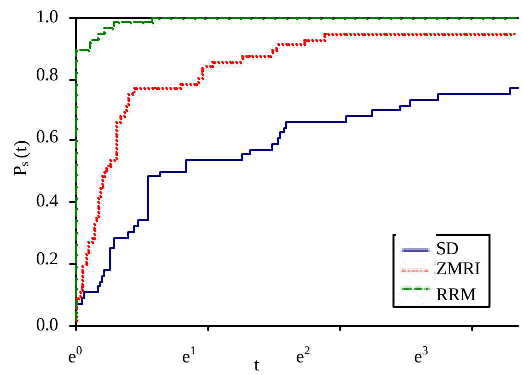

Fig. 2: Performance profile based on number of iterations

89.29% of the test problems with the minimum number of iterations while ZMRI and SD are only about 8.92% and 7.14% of the test problems, respectively. From the right side of Fig. 2, RRM solves 100% of the entire test problems to optimality, followed by ZMRI can solve about 94.64% and lastly SD can solve about 78.57% of the entire test problems to optimality.

CONCLUSION

Several modifications on SD method have been widely used in practical computation as the SD method is the simplest of the gradient methods for solving unconstrained optimization problems. In this study, we proposed a new scaled SD method based on Andrei's approach of scaled conjugate gradient method. The numerical results show that the proposed method performed better than the other classical SD method. For future work, we intend to hybrid our proposed method with other unconstrained optimization methods.

ACKNOWLEDGEMENT

We would like to thank the Ministry of Higher Education of Malaysia for the financial support of this research (FRGS Vot 59256).

REFERENCES

Abidin, Z.A.Z., M. Mamat, M. Rivaie and I. Mohd, 2014. A new steepest descent method. Proceedings of the 3rd International Conference on Mathematical Sciences, Vol. 1602, December 17-19, 2013, AIP, Melville, New York, ISBN:978-0-7354-1236-1, pp: 273-278.

Andrei, N., 2007. Scaled conjugate gradient algorithms for unconstrained optimization. Comput. Optim. Appl., 38: 401-416.

Andrei, N., 2008. An unconstrained optimization test functions collection. Adv. Mod. Optimiz., 10: 147-161.

Barzilai, J. and J.M. Borwein, 1988. Two-point step size gradient methods. IMA. J. Numer. Anal., 8: 141-148.

Cauchy, A., 1847. [General method for solving systems of simultaneous equations (In French)]. Comp. Rend. Sci. Paris, 25: 536-538.

Dai, Y.H. and L.Z. Liao, 2002. R-linear convergence of the Barzilai and Borwein gradient method. IMA. J. Numer. Anal., 22: 1-10.

Dolan, E.D. and J.J. More, 2002. Benchmarking optimization software with performance profiles. Math. Program., 91: 201-213.

Jamil, M. and X.S. Yang, 2013. A literature survey of benchmark functions for global optimisation problems. Intl. J. Math. Modell. Numer. Optimisation, 4: 150-194.

Mamat, M., S.Y. Aw and I. Mohd, 2009. An efficient algorithm for steepest descent method for unconstrained optimization. J. Sci. Technol. UTHM., 1: 13-25.

Nocedal, J. and S.J. Wright, 2006. Numerical Optimization. 2nd Edn., Springer, Berlin, Germany, ISBN-13:978-0387-30303-1, Pages: 664.

Raydan, M., 1993a. On the Barzilai and Borwein choice of steplength for the gradient method. IMA. J. Numer. Anal., 13: 321-326.

Raydan, M., 1993b. On the Barzilai and Borwein choice of steplength for the gradient method. IMA. J. Numer. Anal., 13: 321-326.

Wen, G.K., M. Mamat, I. bin Mohd and Y. Dasril, 2012. A novel of step size selection procedures for steepest descent method. Appl. Math. Sci., 6: 2507-2518.

Witte, B.F. and W.R. Holst, 1964. Two new direct minimum search procedures for functions of several variables. Proceedings of the International Joint Conference on Computer (AFIPS '64), April 21-23, 1964, ACM, New York, USA., pp: 195-209.

Zhang, L., W. Zhou and D. Li, 2006. Global convergence of a modified Fletcher-reeves conjugate gradient method with Armijo-type line search. Numerische Math., 104: 561-572.