UNIVERSITY THOUGHT
Publication in Natural Sciences, Vol. 7, No. 1, 2017, pp. 41-45.

doi:10.5937/univtho7-14337
Original Scientific Paper

DETERMINATION OF ACCELERATED FACTORS IN GRADIENT DESCENT ITERATIONS BASED ON TAYLOR'S SERIES

MILENA PETROVIĆ[1*], NATAŠA KONTREC[1], STEFAN PANIĆ[1]

[1]Faculty of Natural Science and Mathematics, University of Priština, Kosovska Mitrovica, Serbia

ABSTRACT

In this paper the efficiency of accelerated gradient descent methods regarding the way of determination of accelerated factor is considered. Due to the previous researches we assert that the use of Taylor's series of posed gradient descent iteration in calculation of accelerated parameter gives better final results than some other choices. We give a comparative analysis of efficiency of several methods with different approaches in obtaining accelerated parameter. According to the achieved results of numerical experiments we make a conclusion about the one of the most optimal way in defining accelerated parameter in accelerated gradient descent schemes.

Keywords: Line search, gradient descent methods, quasi-Newton method, convergence rate.

INTRODUCTION

We analyze nonlinear optimization methods for the minimization of an objective function f: $\Re^n \to \Re$:

$$\min f(x), \quad x \in \Re^n \tag{1}$$

In this paper we suppose that f is uniformly convex and twice continuously differentiable function. For these classes of functions, furthermore we adopt the next often used representation of optimization models:

$$x_{k+1} = x_k + t_k d_k \tag{2}$$

where x_{k+1} denotes the function value in the next iterative point, x_k presents the function value in the current iteration, t_k is the iterative step size value and d_k is the search direction vector.

Since the first methods for solving non-linear minimization problems have been developed it was clear that the two main characteristics of these iterations: the step length and the search direction (parameters t_k and d_k in (2)) crucially determine accelerated features of the optimization method. Therewith, we expect the value of an iterative step size to be optimal at the sense that it is not too high to misses out the minima and at the same time not too small to provide unnecessary high number of iterations. In order to obtain the minimal value of the objective function we assume that the search direction vector fulfills the descent condition: this paper we suppose that f is uniformly convex and:

$$g_k{}^T d_k < 0,$$

where with g_k we denote a gradient of the objective function in the $k-$th iteration.

THEORETICAL PART

One of the first choices for descending search direction vector, which satisfies the previous condition, is proposed in classical gradient descent method (GD method). In this iteration $d_k = -g_k$. This is the one of the oldest iterations for solving nonlinear optimization problems. Theoretically, GD method has good convergence properties. In practical sense GD iteration is very slow and not really useful for the problems with large number of variables. In order to prevent these disadvantages but at the same time conserving the descending property of the negative gradient, some authors developed iterative gradient descent schemes more efficient in practical usage than GD method (Fletcher & Reeves, 1964; Polak & Ribiére, 1969; Polyak, 1969).

$$x_{k+1} = x_k - \theta_k t_k d_k \tag{3}$$

where the acceleration parameter θ_k is calculated as $\theta_k = a_k/b_k$. Here $a_k = t_k g_k{}^T g_k$, $b_k = -t_k y_k{}^T g_k$ and $y_k = g_{k+1} - g_k$. The value of iterative step-length t_k is derived using backtracking line search procedure. This iteration is noted as AGD-method. The AGD method is compared with gradient descent GD-method. In numerical experiments, obtained for 340 test problems, notably better results in favor to the AGD scheme are registered. The analyzed characteristics are the number of iterations, the CPU time and the number of function evaluations. Regarding all three tested properties, the AGD iteration has provided considerably reduction of measured values comparing to the GD scheme.

Considering the obtained results from Andrei (2006) in Stanimirovic & Miladinovic (2010) the authors identify a class of accelerated gradient descent methods. On their opinion all gradient descent schemes with somehow defined accelerated factor belong to this class of methods. They have offered their way of deriving accelerated factor and described it in (Stanimirovic & Miladinovic, 2010). For that purpose they have constructed accelerated gradient descent SM-method as next:

[*] Corresponding author: milena.petrovic@pr.ac.rs

$$x_{k+1} = x_k - t_k \gamma_k^{-1} d_k \tag{4}$$

To define an accelerated factor, noted in (4) as γ_k, the authors used a Taylor's expansion of the objective function f_{k+1}:

$$f(x_{k+1}) = f(x_k) - t_k g_k^T \gamma_k^{-1} g_k + \frac{1}{2} t_k^2 \left(\gamma_k^{-1} g_k\right)^T \nabla^2 f(\xi) \gamma_k^{-1} g_k \tag{5}$$

where $\xi \in [x_k, x_{k+1}]$ is a point:

$$\xi = x_k + \alpha\left(x_{k+1} - x_k\right) = x_k - \alpha t_k \gamma_k^{-1} d_k, \quad 0 \le \alpha \le 1 \tag{6}$$

Hessian $\nabla^2 f(\xi)$ in (5) is replaced by $\nabla^2 f(\xi) = \gamma_{k+1} I$, which transforms expression (5) into:

$$f(x_{k+1}) = f(x_k) - t_k \gamma_k^{-1} \|g_k\|^2 + \frac{1}{2} t_k^2 \gamma_{k+1} \gamma_k^{-2} \|g_k\|^2 \tag{7}$$

From the previous relation arises the value of the accelerated parameter γ_{k+1} of the *SM*-method:

$$\gamma_{k+1} = \frac{\gamma_k \left| f(x_{k+1}) - f(x_k) \right| + t_k \|g_k\|^2}{t_k^2 \|g_k\|^2} \tag{8}$$

Linear convergence for so defined SM iteration is proven as well as improvements in performances comparing to the GD and AGD- methods. Achieving a reduction in number of iterations, CPU time and in number of function evaluations in comparison with GD scheme was expected. But prominently reduction of resulted values achieved by the SM-method in all three tested characteristics compered to the accelerated AGD iteration leads us to conclusion that defining an accelerated parameter using the Taylor's expansion gives better practical results.

Later on in (Petrovic & Stanimirovic 2014; Petrovic 2015; Stanimirovic et al., 2015) authors use a favorable results from Stanimirovic & Miladinovic (2010) and continue to explore the way of defining an accelerated parameter using the Taylor's expansion and different forms of iterations for solving nonlinear unconstrained optimization problems. For this investigation we consider results from (Petrovic, 2015). In Petrovic (2015), an iteration with two step lengths, α_k and β_k is presented as:

$$x_{k+1} = x_k - \left(\alpha_k \gamma_k^{-1} + \beta_k\right) d_k \tag{9}$$

Using the Taylor's expansion applied on a scheme (9) and an approximation of Hessian in a current iterative point by the product $\gamma_{k+1} I$ where an accelerated parameter γ_{k+1} is included, the value of accelerated factor of the ADSS-method in $(k + 1)$-iterative point is:

$$\gamma_{k+1} = 2\frac{f(x_{k+1}) - f(x_k) + \left(\alpha_k \gamma_k^{-1} + \beta_k\right)\|g_k\|^2}{\left(\alpha_k \gamma_k^{-1} + \beta_k\right)^2 \|g_k\|^2} \tag{10}$$

Like in Stanimirovic & Miladinovi (2010) we assume that $\gamma_{k+1} > 0$ otherwise the Second-Order Necessary Condition and Second-Order Sufficient Condition will not be fulfilled. However, in the case when $\gamma_{k+1} < 0$ we take $\gamma_{k+1} = 1$. This way we ensure that when Gk is not positive definite matrix than taking $\gamma_{k+1} = 1$ produces that the search direction is $-g_k$ and that is a descent direction indeed. The next iterative point x_{k+2} is then calculated as:

$$x_{k+2} = x_{k+1} - \left(\alpha_{k+1} \gamma_{k+1}^{-1} + \beta_{k+1}\right) d_{k+1}$$

which presents an accelerated gradient descent iteration. In all three mentioned accelerated gradient descent models (AGD, SM, ADSS) the values of iterative step sizes are computed by the Armio's backtracking line search procedures which is generally described through the next three steps. Applying the SM and the ADSS methods linear convergence is proven on the set of uniform convex functions and under additional assumptions for the strictly convex quadratics as well. Algorithms (0.2) and (0.3) display the SM and the ADSS accelerated models respectively.

Algorithm0.1 The backtracking line search starting from $t = 1$.

```
Require: Objective function $f(x)$, the direction of the search ($d_k$) at the point $x_k$ and numbers $0 < \sigma < 0.5$ and $\beta \in (0, 1)$.
1: $t = 1$.
2: While $f(x_k + td_k) > f(x_k) + \sigma t g_k^T d_k$, take $t := t\beta$.
3: Return $t_k = t$.
```

Algorithm0.2 SM-method.

```
Require: Objective function $f(x)$ and chosen initial point $x_0 \in dom(f)$.
1: Set $k = 0$ and compute $f(x_0)$, $g_0 = \nabla f(x_0)$ and take $\gamma_0 = 1$.
2: If test criteria are fulfilled then stop the iteration; otherwise, go to the next step.
3: (Backtracking) Find the step size $t_k \in (0, 1]$ using Backtracking procedure with $d_k = -\gamma_k^{-1} g_k$.
4: Compute $x_{k+1} = x_k - t_k \gamma_k^{-1} g_k$, $f(x_{k+1})$ and $g_{k+1} = \nabla f(x_{k+1})$.
5: Determine the scalar approximation $\gamma_{k+1}$ of the Hessian of $f$ at the point $x_{k+1}$ using (8).
6: If $\gamma_{k+1} < 0$, then take $\gamma_{k+1} = 1$.
7: Set $k := k + 1$, go to the step 2.
8: Return $x_{k+1}$ and $f(x_{k+1})$.
```

Algorithm0.3 Accelerated Double Step Size method (*ADSS* method).

```
Require: $0 < \rho < 1$, $0 < \tau < 1$, $x_0$, $\gamma_0 = 1$.
1: Set $k = 0$, compute $f(x_0)$, $g_0$ and take $\gamma_0 = 1$.
2: If $\|g_k\| < \xi$, then go to Step 9, else continue by the next step.
3: Find the step size $\alpha_k$ applying Backtracking1 procedure.
4: Find the step size $\beta_k$ applying Backtracking2 procedure.
5: Compute $x_{k+1}$ using (9).
6: Determine the scalar $\gamma_{k+1}$ using (10).
7: If $\gamma_{k+1} < 0$, then take $\gamma_{k+1} = 1$.
8: Set $k := k + 1$, go to Step 2.
9: Return $x_{k+1}$ and $f(x_{k+1})$.
```

NUMERICAL COMPARATIONS

In Stanimirovic & Miladinovic (2010) authors have tested 30 test functions from Andrei (2008) given in generalized or extended form as a large scale unconstrained test problems. They considered for each test function 10 different numerical experiments with the number of variables: 100, 500, 1000, 2000, 3000, 5000, 7000, 8000, 10000 and 15000. A substantial outcome of these numerical experiments is that the SM method shows the best results for 20 test functions in the sense of number of iterations needed to achieve requested accuracy, while AGD is the best in the remaining 10 test problems. Considering the CPU time and the number of function evaluations, the SM method shows better performance for 23 test functions.

Since the both of the methods, the SM and the AGD, belong to the class of accelerated gradient descent methods we can conclude that computing an accelerated variable using the Taylor's expansion provides a far better practical results in reducing the number of iterations, the number of function evaluations and needed CPU time. That was the reason to continue with this way of computing the parameter of acceleration. The similar approaches are applied in (Petrovic & Stanimirovic, 2014; Petrovic, 2015; Stanimirovic et al., 2015). In this work we consider results published in Petrovic (2015).

Table 1. Summary of numerical results for SM and ADSS tested on 25 large scale test functions regarding number of iteration.

Test function	Nomber of iterations	
	ADSS	SM
Extended Penalty	589	50
Perturbed quadratic	111972	397
Raydan-1	21125	34
Diagonal 1	10417	37
Diagonal 3	10574	49
Generalized Tridiagonal -1	278	77
Extended Tridiagonal -1	3560	70
Extended Three Expon	164	40
Diagonal 4	80	780
Extended Himmelblau	168	70
Quadr. Diag. Perturbed	53133	393
Quadratic QF1	114510	425
Extended Quad. Penalty QP1	224	60
Extended Quad. Penalty QP2	162	60
Quadratic QF2	118801	60
Extended EP1	68	40
Extended Tridiagonal - 2	584	80
Arwhead	10	64
Almost Perturbed Quadratic	110121	397
Engval1	185	70
Quartc	190	70
Generalized quartic	156	70
Diagonal 7	90	2201
Diagonal 8	96	2213
Diagonal 9	11235	43

Numerical tests presented in Petrovic (2015) confirm noticeably better performance of accelerated double step size ADSS method comparing to the accelerated gradient descent SM scheme with one step length parameter. This also lead us to conclusion that properly defined values of iterative step lengths (as well as the number of step sizes involved in method) substantially causes the level of efficiency of analyzed method.

Table 2. Summary of numerical results for SM and ADSS tested on 25 large scale test functions regarding CPU time.

Test function	CPU time	
	ADSS	SM
Extended Penalty	5	4
Perturbed quadratic	1868	0
Raydan-1	178	4
Diagonal 1	116	0
Diagonal 3	209	1
Generalized Tridiagonal -1	2	0
Extended Tridiagonal -1	35	0
Extended Three Expon	1	0
Diagonal 4	0	0
Extended Himmelblau	0	0
Quadr. Diag. Perturbed	1193	1
Quadratic QF1	2127	0
Extended Quad. Penalty QP1	12	2
Extended Quad. Penalty QP2	7	4
Quadratic QF2	2544	1
Extended EP1	1	0
Extended Tridiagonal - 2	0	0
Arwhead	0	3
Almost Perturbed Quadratic	2148	0
Engval1	7	0
Quartc	0	0
Generalized quartic	0	0
Diagonal 7	0	33
Diagonal 8	0	40
Diagonal 9	118	0

To confirm this ascertainment in this work the SM and the ADSS schemes are numerically compared and the numerical outcomes from Petrovic (2015) are used. First, let us point it out on the common features of these two iterations:

- Computation of accelerated factor for each of these models is achieved in a similar way, using the Taylor's expansion and in accordance with the posed formulation of the iteration;
- The value of the single step size in the *SM* is obtained by the Backtracking line search technique as well as each of the value of two needed step length parameters in the *ADSS* scheme;
- Both methods have a negative gradient for the search direction, i.e. both methods are gradient descent.

For each of the 25 test functions ten experiments are taken for the larger number of variables:1000, 2000, 3000, 5000, 7000, 8000, 10000, 15000, 20000 and 30000. Analyzed characteristics are number of iterations, needed CPU time of execution and the number of function evaluations. Taking the next exit criteria:

$$\|g_k\| \le 10^{-6} \;\; and \;\; \frac{|f(x_{k+1}) - f(x_k)|}{1 + |f(x_k)|} \le 10^{-6}$$

results are displayed in the following Tables 1, 2, 3.

From the given Tables the next conclusions follows:

- By using the ADSS scheme in 21 out of 25 test functions considerably lower number of iterations is realized then by using the SM scheme;
- The ADSS iteration is faster than the SM for 17 functions and for 5 test functions both methods need the same CPU time; which presents an accelerated.
- By using the *ADSS* scheme in 20 out of 25 test functions much lower number of function evaluations is achieved then using the *SM* scheme.

Expand view of the previous statements is given trough the average values, arranged in Table 4.

Table 3. Summary of numerical results for SM and ADSS tested on 25 large scale test functions regarding number of evaluation..

Test function	No. of funct. evaluation	
	ADSS	SM
Extended Penalty	2987	1780
Perturbed quadratic	632724	1714
Raydan-1	116757	4844
Diagonal 1	56135	1448
Diagonal 3	59425	1048
Generalized Tridiagonal -1	989	719
Extended Tridiagonal -1	30686	420
Extended Three Expon	1224	350
Diagonal 4	530	2590
Extended Himmelblau	566	480
Quadr. Diag. Perturbed	547850	1719
Quadratic QF1	649643	1755
Extended Quad. Penalty QP1	2566	841
Extended Quad. Penalty QP2	2057	843
Quadratic QF2	662486	836
Extended EP1	764	487
Extended Tridiagonal - 2	2144	420
Arwhead	30	1082
Almost Perturbed Quadratic	1712	1712
Engval1	2177	460
Quartc	430	390
Generalized quartic	423	614
Diagonal 7	220	6633
Diagonal 8	594	6709
Diagonal 9	60566	3646

Generally, from 250 testings we can conclude that the *ADSS* outperforms the *SM* respect to all tested characteristics: number of iterations, CPU time and number of function evaluations. Considering the number of iterations, the *ADSS* exceeds *SM* about 70 times. Needed CPU time is averagely 113 times less in favor to *ADSS* comparing to *SM* and regarding function evaluations about 80 times lower number is needed with regard to the *SM*.

Remark 0.1. *In Stanimirovic et al. (2015); Petrovic et al. (2016) two more accelerated gradient descent schemes with accelerated parameters derived from the features of the Taylor's series are presented. Linear convergence of these methods and their numerical efficiency is proved and tested. All of these improved results that are mentioned or analyzed confirm that it is reasonable to detected a class of accelerated gradient descent methods with accelerated parameter obtained by the Taylor's expansion of posed iteration.*

Table 4. Average numerical outcomes for 25 test functions tried out on 10 numerical experiments in each iteration..

Average performances	ADSS	SM
Number of iterations	314	22739.68
CPU time (sec)	3.72	422.84
Number of function evaluations	1741.6	138450.4

CONCLUSION

In this paper the efficiency of an accelerated gradient descent method with accelerated parameter achieved using the properties of the Taylor's expansion is described and numerically proved. For that purpose we have pointed out on a different ways of defining a factor of acceleration. The results of the numerical tests in Stanimirovic & Miladinovic (2010) which show the benefits of calculating the acceleration parameter trough the Taylor's expansion instead of means stated in Andrei (2006), were the reason to continue investigation of developing an accelerated parameter this way applied on a different form of a gradient descent iteration.

Double step size gradient descent ADSS model proposed in Petrovic (2015), with all three crucial elements of iteration (an accelerated parameter, step sizes and search direction) similarly defined, is compared with the SM method. The efficiency of the ADSS model regarding all analyzed characteristics (number of iterations, CPU time and number of function evaluations) in comparison to the accelerated gradient descent single step size SM method has been numerically confirmed.

At the end we can indicate that the problem stated in this paper can be exploited in some new ways. More precisely, the problem of finding an acceleration parameter with some different forms of gradient descent iterations is still actual.

REFERENCES

Andrei, N. 2006. An acceleration of gradient descent algoritham with backtracing for unconstrained optimization. Numer. Algor, 42.

Andrei, N. 2008. An unconstrained optimization test functions collection. Advanced Modeling and Optimization, 10(1).

Fletcher, R., & Reeves, C. 1964. Function minimization by conjugate gradients. Comput. J., 7, pp. 149-154.

Petrović, M. 2015. An Accelerated Double Step Size Method In Unconstrained Optimization. Applied Mathematics and Computation, .

Petrović, M., Rakočević, V., Kontrec, N., Panić, S., & Ilić, D. 2016. Hibridization of accelearted gradient descent method. Numer. Algor., . under review.

Petrović, M., & Stanimirović, P. 2014. Accelerated Double Direction Method For Solving Unconstrained Optimization Problems. Mathematical Problems in Engineering, 2014, pp. 309-319.

Polak, E., & Ribiére, G. 1969. Note sur la convergence de méthodes de directions conjuguées. Revue Francaise Informat. Reserche Opérationnelle, 16, pp. 35-43.

Polyak, B.T. 1969. The conjugate gradient method in extreme problems. USSR Comp. Math. Math. Phys., 9, pp. 94-112.

Stanimirović, P.S., & Miladinović, M.B. 2010. Accelerated gradient descent methods with line search. Numer. Algor, 54, pp. 503-520.

Stanimirović, P.S., Milovanović, G.V., Petrović, M., & Kontrec, N.Z. 2015. Transformation of Accelerated Double Step Size Method for Unconstrained Optimization. Mathematical Problems in Engineering, .