Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 283679, 8 pages
http://dx.doi.org/10.1155/2015/283679

Research Article

A Transformation of Accelerated Double Step Size Method for Unconstrained Optimization

Predrag S. Stanimirović,[1] Gradimir V. Milovanović,[2,3] Milena J. Petrović,[4] and Nataša Z. Kontrec[4]

[1]*Faculty of Sciences and Mathematics, University of Niš, Višegradska 33, 18000 Niš, Serbia*
[2]*Serbian Academy of Sciences and Arts, Kneza Mihaila 35, 11000 Beograd, Serbia*
[3]*State University of Novi Pazar, 36300 Novi Pazar, Serbia*
[4]*Faculty of Sciences and Mathematics, University of Priština, Lole Ribara 29, 28000 Kosovska Mitrovica, Serbia*

Correspondence should be addressed to Predrag S. Stanimirović; pecko@pmf.ni.ac.rs

Received 19 January 2015; Accepted 24 February 2015

Academic Editor: Alejandro Ortega-Moñux

A reduction of the originally double step size iteration into the single step length scheme is derived under the proposed condition that relates two step lengths in the accelerated double step size gradient descent scheme. The proposed transformation is numerically tested. Obtained results confirm the substantial progress in comparison with the single step size accelerated gradient descent method defined in a classical way regarding all analyzed characteristics: number of iterations, CPU time, and number of function evaluations. Linear convergence of derived method has been proved.

1. Introduction and Background

The SM iteration from [1] is defined by the iterative process

$$\mathbf{x}_{k+1} = \mathbf{x}_k - t_k \gamma_k^{-1} \mathbf{g}_k, \tag{1}$$

where $\mathbf{x}_{k+1}$ is a new iterative point, $\mathbf{x}_k$ is the previous iterative point, $\mathbf{g}_k$ is the gradient vector (search direction), t_k is a step length, and $\gamma_k > 0$ is the acceleration parameter. In [1] it is verified that the accelerated gradient SM iteration (1) outperforms the gradient descent, GD, as well as the Andrei's accelerated gradient descent AGD method from [2].

Double direction and double step size accelerated methods, denoted by ADD and ADSS methods, respectively, for solving the problems of unconstrained optimization are presented in [3, 4]. These two algorithms can be generally formulated through the next merged expression:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{s}_k + \beta_k \mathbf{d}_k, \tag{2}$$

where $\mathbf{x}_k$ is the previous iterative point and real values α_k and β_k denote two step lengths while the vectors $\mathbf{s}_k$ and $\mathbf{d}_k$ are two vector directions. The values of step lengths are determined by backtracking line search techniques. The gradient is basically used for defining a search direction, but some new suggestions for deriving a descending vector direction are given in [3, 5]. Taking the substitutions

$$\mathbf{s}_k = -\gamma_k^{-1} \mathbf{g}_k, \qquad \beta_k = \alpha_k^2, \tag{3}$$

into (2) produces the ADD iterative scheme from [3]:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \gamma_k^{-1} \mathbf{g}_k + \alpha_k^2 \mathbf{d}_k, \tag{4}$$

where γ_k represents the acceleration parameter for the iteration (4). The benefits of the acceleration properties that arise from the usage of the parameter γ_k are explained in [3]. The so called nonaccelerated version of ADD method (NADD method shortly) is defined in order to numerically verify the acceleration property of the parameter γ_k. Three methods, SM, ADD, and NADD, are numerically compared in [3]. Results show the enormous efficiency of ADD scheme in comparison with its nonaccelerated counterpart NADD. Derivation of the direction vector $\mathbf{d}_k$ is explained by the Algorithm 3.2 in [3]. The ADD outperforms its competitive SM method from [1] with respect to the number of iterations.

By replacing the vectors $\mathbf{s}_k$ and $\mathbf{d}_k$ from (2) by $-\gamma_k^{-1}\mathbf{g}_k$ and $-\mathbf{g}_k$, respectively, the next iteration is defined as

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\alpha_k\gamma_k^{-1} + \beta_k\right)\mathbf{g}_k. \tag{5}$$

The previous scheme is noted as ADSS model and it is proposed in [4]. In the same paper, a huge improvement in performances of this accelerated gradient descent method when compared to the accelerated gradient descent SM method from [1] is numerically confirmed.

The main contribution of the present paper is a transformation of the double step size iterative scheme (5) for unconstrained optimization into an appropriate accelerated single step size scheme (called TADSS shortly). Convergence properties of the introduced method are investigated. A special contribution is given by the numerical confirmation that the TADSS algorithm developed from the double step size ADSS model (5) is evidently more efficient than the accelerated SM method obtained in a classical way. Surprisingly, numerical experiments show that the TADSS method overcomes the initial ADSS method.

The paper is organized in the following way. The reduction of the double step size ADSS model into the single step size iteration TADSS and the presentation of defined accelerated gradient decent model are given in Section 2. Section 3 contains the convergence analysis of derived algorithm for uniformly convex and strictly convex quadratic functions. The results of numerical experiments as well as their comparative analysis of developed method and its forerunners are illustrated in Section 4.

2. Transformation of ADSS Scheme into a Single Step Size Iteration

Very advanced numerical results obtained in [4] motivated further research on this topic. An idea is to investigate the properties of a single step size method developed as a reduction of the double step size ADSS model. This reduction is defined by an additional assumption which represents a trade-off between two step length parameters α_k and β_k in the ADSS scheme:

$$\alpha_k + \beta_k = 1. \tag{6}$$

Taking into account assumption (6) into expression (5), which defines the ADSS iteration, leads to the iterative process

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left[\alpha_k\left(\gamma_k^{-1} - 1\right) + 1\right]\mathbf{g}_k. \tag{7}$$

The iteration (7) is noted as transformed ADSS method, or shortly TADSS method. Defined TADSS iteration represents not only a reduction of the double step size ADSS model into the corresponding single step size method, but also a sort of modification of the single step size SM iteration from [1]. This modification can be explained as the substitution of the product $t_k\gamma_k^{-1}$, from the SM iteration (1), by the multiplying factor $\alpha_k(\gamma_k^{-1}-1)+1$ of the gradient from the TADSS iteration (7).

For the sake of simplicity, we use the notation $\phi_k = \alpha_k(\gamma_k^{-1} - 1) + 1$ whenever it is possible. The value of the acceleration parameter γ_{k+1} in $(k+1)$th iteration can be derived by using Taylor's expansion, similarly as described in [1, 3, 4]:

$$f\left(\mathbf{x}_{k+1}\right) \approx f\left(\mathbf{x}_k\right) - \mathbf{g}_k^T\phi_k\mathbf{g}_k + \frac{1}{2}\phi_k\mathbf{g}_k^T\nabla^2 f\left(\boldsymbol{\xi}\right)\phi_k\mathbf{g}_k. \tag{8}$$

The vector $\boldsymbol{\xi}$ in (8) satisfies

$$\boldsymbol{\xi} \in \left[\mathbf{x}_k, \mathbf{x}_{k+1}\right], \quad \boldsymbol{\xi} = \mathbf{x}_k + \delta\left(\mathbf{x}_{k+1} - \mathbf{x}_k\right) = \mathbf{x}_k - \delta\phi_k\mathbf{g}_k, \quad 0 \le \delta \le 1. \tag{9}$$

Further, it is reasonable to replace in (8) the Hessian $\nabla^2 f(\boldsymbol{\xi})$ by the diagonal matrix $\gamma_{k+1}I$, where γ_{k+1} is an appropriately chosen real number. This replacement implies

$$f\left(\mathbf{x}_{k+1}\right) \approx f\left(\mathbf{x}_k\right) - \phi_k\left\|\mathbf{g}_k\right\|^2 + \frac{1}{2}\phi_k^2\gamma_{k+1}\left\|\mathbf{g}_k\right\|^2. \tag{10}$$

The relation (10) allows us to compute the acceleration parameter γ_{k+1}:

$$\gamma_{k+1} = 2\frac{f\left(\mathbf{x}_{k+1}\right) - f\left(\mathbf{x}_k\right) + \phi_k\left\|\mathbf{g}_k\right\|^2}{\phi_k^2\left\|\mathbf{g}_k\right\|^2}. \tag{11}$$

Next, the natural inequality $\gamma_{k+1} > 0$ is inevitable. This condition is required in order to fulfill second-order necessary condition and second-order sufficient condition. The choice $\gamma_{k+1} = 1$ is reasonable in the case when the inequality $\gamma_{k+1} < 0$ appears for some k. This choice produces the next iterative point as

$$\mathbf{x}_{k+2} = \mathbf{x}_{k+1} - \left(\alpha_{k+1}(1-1) + 1\right)\mathbf{g}_{k+1} = \mathbf{x}_{k+1} - \mathbf{g}_{k+1}, \tag{12}$$

which evidently represents the classical gradient descent step.

We consider now the $(k+1)$th iteration, $\mathbf{x}_{k+2}$, which is given by

$$\mathbf{x}_{k+2} = \mathbf{x}_{k+1} - \left[\alpha_{k+1}\left(\gamma_{k+1}^{-1} - 1\right) + 1\right]\mathbf{g}_{k+1} = \mathbf{x}_{k+1} - \phi_{k+1}\mathbf{g}_{k+1}. \tag{13}$$

Examine the function $\Phi_{k+1}(\alpha)$:

$$\Phi_{k+1}(\alpha) = f\left(\mathbf{x}_{k+1}\right) - \left[\alpha\left(\gamma_{k+1}^{-1} - 1\right) + 1\right]\left\|\mathbf{g}_{k+1}\right\|^2 + \frac{1}{2}\left[\alpha\left(\gamma_{k+1}^{-1} - 1\right) + 1\right]^2\gamma_{k+1}\left\|\mathbf{g}_{k+1}\right\|^2, \tag{14}$$

defined as the finite part of the Taylor expansion of the function

$$f\left(\mathbf{x}_{k+1} - \left[\alpha\left(\gamma_{k+1}^{-1} - 1\right) + 1\right]\mathbf{g}_{k+1}\right) \tag{15}$$

under the assumption $\gamma_{k+2} = \gamma_{k+1}$. This function is convex when $\gamma_{k+1} > 0$, and its derivative $\Phi_{k+1}(\alpha)'_\alpha$ is calculated in the following way:

$$\begin{aligned}(\Phi_{k+1})'_\alpha &= -\left(\gamma_{k+1}^{-1} - 1\right)\|\mathbf{g}_{k+1}\|^2 \\ &\quad + \left(\alpha\left(\gamma_{k+1}^{-1} - 1\right) + 1\right)\gamma_{k+1}\|\mathbf{g}_{k+1}\|^2\left(\gamma_{k+1}^{-1} - 1\right) \\ &= \left(\gamma_{k+1}^{-1} - 1\right)\left(-1 + \left(\alpha\left(\gamma_{k+1}^{-1} - 1\right) + 1\right)\gamma_{k+1}\right)\|\mathbf{g}_{k+1}\|^2 \\ &= \left(\gamma_{k+1}^{-1} - 1\right)\left(-1 + \alpha - \alpha\gamma_{k+1} + \gamma_{k+1}\right)\|\mathbf{g}_{k+1}\|^2 \\ &= \left(\gamma_{k+1}^{-1} - 1\right)(\alpha - 1)\left(1 - \gamma_{k+1}\right)\|\mathbf{g}_{k+1}\|^2 \\ &= \gamma_{k+1}^{-1}\left(1 - \gamma_{k+1}\right)^2(\alpha - 1)\|\mathbf{g}_{k+1}\|^2. \end{aligned} \tag{16}$$

Since the inequality $\gamma_{k+1} > 0$ is achieved, the following is valid:

$$(\Phi_{k+1})'_\alpha < 0 \Longleftrightarrow \alpha < 1, \qquad (\Phi_{k+1})'_\alpha = 0 \Longleftrightarrow \alpha = 1. \tag{17}$$

Therefore, the function $\Phi_{k+1}(\alpha)$ decreases in the case $(\Phi_{k+1})'_\alpha < 0$ and achieves its own minimum in the case $\alpha = 1$. According to the criteria given by (17), desirable values for α are within the interval $(-\infty, 1]$. Now, (7) is a kind of the gradient descent process in the case $\phi_k = \alpha_k(\gamma_k^{-1} - 1) + 1 > 0$. Since $\gamma_k > 0$, it is easy to verify the following condition for the step length α_k:

$$\alpha_k \le \frac{\gamma_k}{\gamma_k - 1}. \tag{18}$$

Since $\gamma_k/(\gamma_k - 1) > 1$ in the case $\gamma_k > 1$, this fractional number is not appropriate upper bound for α_k in this case. On the other hand, the inequality $\gamma_k/(\gamma_k - 1) < 0$ holds in the case $\gamma_k < 1$, so that $\gamma_k/(\gamma_k - 1)$ is an appropriate upper bound for α_k in this case.

Further analysis of (10) in the case $\gamma_k < 1$ gives

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \approx \phi_k\|\mathbf{g}_k\|^2 - \frac{1}{2}\phi_k^2\gamma_{k+1}\|\mathbf{g}_k\|^2, \tag{19}$$

which implies

$$f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k) \Longleftrightarrow \phi_k \le \frac{2}{\gamma_{k+1}}. \tag{20}$$

Taking into account $\gamma_k < 1$, it is not difficult to verify that the criterion (20) restricts desirable values for α_k within the interval

$$\alpha_k \le \frac{\gamma_k(2 - \gamma_{k+1})}{\gamma_{k+1}(1 - \gamma_k)}. \tag{21}$$

Final conclusion is that the upper bound for α_k is defined in the case $\gamma_k < 1$, $\gamma_{k+1} < 1$, as the minimum between the upper bounds defined in (18) and (21):

$$\alpha_k \le \min\left\{\frac{\gamma_k}{\gamma_k - 1}, \frac{\gamma_k(2 - \gamma_{k+1})}{\gamma_{k+1}(1 - \gamma_k)}\right\}. \tag{22}$$

According to the previous discussion, the iterative step α_k is derived by the backtracking line search procedure presented in Algorithm 1.

Algorithm 1 (calculation of the step size α_k by the backtracking line search which starts from the upper bound defined in (22)). *Requirement*: objective function $f(\mathbf{x})$, the direction $\mathbf{d}_k$ of the search at the point $\mathbf{x}_k$, and real numbers $0 < \sigma < 0.5$ and $\eta \in (\sigma, 1)$.

(1) Set $\alpha = \min\{\gamma_k/(\gamma_k - 1), \gamma_k(2 - \gamma_{k+1})/\gamma_{k+1}(1 - \gamma_k)\}$.

(2) While $f(\mathbf{x}_k + \alpha\mathbf{d}_k) > f(\mathbf{x}_k) + \sigma\alpha\mathbf{g}_k^T\mathbf{d}_k$ take $\alpha := \eta\alpha$.

(3) Return $\alpha_k = \alpha$.

Finally, the TADSS algorithm of the defined accelerated gradient descent scheme (7) is presented.

Algorithm 2 (transformed accelerated double step size method (TADSS method)). *Requirement*: $0 < \rho < 1$, $0 < \tau < 1$, $\mathbf{x}_0$, $\gamma_0 = 1$.

(1) Set $k = 0$, compute $f(\mathbf{x}_0)$, $\mathbf{g}_0$, and take $\gamma_0 = 1$.

(2) If $\|\mathbf{g}_k\| < \epsilon$, then go to Step 8; else continue by the next step.

(3) Find the step size α_k applying Algorithm 1.

(4) Compute $\mathbf{x}_{k+1}$ using (7).

(5) Determine the scalar γ_{k+1} using (11).

(6) If $\gamma_{k+1} < 0$ or $\gamma_{k+1} > 1$, then take $\gamma_{k+1} = 1$.

(7) Set $k := k + 1$; go to Step 2.

(8) Return $\mathbf{x}_{k+1}$ and $f(\mathbf{x}_{k+1})$.

3. Convergence of TADSS Scheme

The content of this section is the convergence analysis of the TADSS method. In the first part of this section a set of uniformly convex functions is considered. The proofs of the following statements can be found in [6, 7] and have been omitted.

Proposition 3 (see [6, 7]). *If the function $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable and uniformly convex on $\mathbb{R}^n$ then*

(1) *the function f has a lower bound on $L_0 = \{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) \le f(\mathbf{x}_0)\}$, where $\mathbf{x}_0 \in \mathbb{R}^n$ is available;*

(2) *the gradient $\mathbf{g}$ is Lipschitz continuous in an open convex set B which contains L_0; that is, there exists $L > 0$ such that*

$$(\forall \mathbf{x}, \mathbf{y}) \in B \quad \|\mathbf{g}(\mathbf{x}) - \mathbf{g}(\mathbf{y})\| \le L\|\mathbf{x} - \mathbf{y}\|. \tag{23}$$

Lemma 4. *Under the assumptions of Proposition 3 there exist real numbers m, M satisfying*

$$0 < m \le 1 \le M, \tag{24}$$

such that $f(\mathbf{x})$ has a unique minimizer $\mathbf{x}^*$ and

$$\begin{aligned}
&(\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n) \quad m\|\mathbf{y}\|^2 \le \mathbf{y}^T \nabla^2 f(\mathbf{x})\, \mathbf{y} \le M\|\mathbf{y}\|^2, \\
&(\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n) \quad \frac{1}{2} m\|\mathbf{x}-\mathbf{x}^*\|^2 \le f(\mathbf{x}) - f(\mathbf{x}^*) \\
&\qquad\qquad \le \frac{1}{2} M\|\mathbf{x}-\mathbf{x}^*\|^2, \\
&(\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n) \quad m\|\mathbf{x}-\mathbf{y}\|^2 \le (\mathbf{g}(\mathbf{x}) - \mathbf{g}(\mathbf{y}))^T (\mathbf{x}-\mathbf{y}) \\
&\qquad\qquad \le M\|\mathbf{x}-\mathbf{y}\|^2.
\end{aligned} \tag{25}$$

The value of decreasing of analyzed function through each iteration is given by the next lemma which is restated and proven in [1]. The same estimation can similarly be found considering iteration (7). Theorem 6 is approved in [1] and confirms a linear convergence of the constructed method.

Lemma 5. *For twice continuously differentiable and uniformly convex function f on $\mathbb{R}^n$ and for the sequence $\{\mathbf{x}_k\}$ generated by Algorithm (7) the following inequality is valid:*

$$f(\mathbf{x}_k) - f(\mathbf{x}_{k+1}) \ge \mu\|\mathbf{g}_k\|^2, \tag{26}$$

where

$$\mu = \min\left\{\frac{\sigma}{M}, \frac{\sigma(1-\sigma)}{L}\beta\right\}. \tag{27}$$

Theorem 6. *If the objective function f is twice continuously differentiable as well as uniformly convex on $\mathbb{R}^n$ and the sequence $\{\mathbf{x}_k\}$ is generated by Algorithm 2 then*

$$\lim_{k\to\infty} \|\mathbf{g}_k\| = 0, \tag{28}$$

and the sequence $\{\mathbf{x}_k\}$ converges to $\mathbf{x}^*$ at least linearly.

In the following review the case of strictly convex quadratic functions is analyzed. This set of functions is given by

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x}. \tag{29}$$

In the previous expression A is a real $n\times n$ symmetric positive definite matrix and $\mathbf{b} \in \mathbb{R}^n$. It is assumed that the eigenvalues of the matrix A are given and lined as $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$. Since the convergence for the most gradient methods is quite difficult to analyze, in many research articles of this profile convergence analysis is reduced on the set of convex quadratics [8–10]. The convergence of TADSS method is also analyzed under similar presumptions.

Lemma 7. *By applying the gradient descent method defined by (7) in which parameters γ_k and α_k are given by relation (11) and Algorithm 1 on the strictly convex quadratic function f expressed by relation (29) where $A \in \mathbb{R}^{n\times n}$ presents a symmetric positive definite matrix, the next inequalities hold:*

$$\frac{1}{2\lambda_n} \le \alpha_{k+1}\left(\gamma_{k+1}^{-1} - 1\right) + 1 = \phi_{k+1} \le \frac{1}{\lambda_1}, \tag{30}$$

where λ_1 and λ_n are, respectively, the smallest and the largest eigenvalues of A.

Proof. Considering expression (29), the difference between function value in the current and the previous point is

$$f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k) = \frac{1}{2}\mathbf{x}_{k+1}^T A\mathbf{x}_{k+1} - \mathbf{b}^T\mathbf{x}_{k+1} - \frac{1}{2}\mathbf{x}_k^T A\mathbf{x}_k + \mathbf{b}^T\mathbf{x}_k. \tag{31}$$

Applying expression (7) the following is obtained:

$$\begin{aligned}
f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k) &= \frac{1}{2}(\mathbf{x}_k - \phi_k\mathbf{g}_k)^T A(\mathbf{x}_k - \phi_k\mathbf{g}_k) \\
&\quad - \mathbf{b}^T(\mathbf{x}_k - \phi_k\mathbf{g}_k) - \frac{1}{2}\mathbf{x}_k^T A\mathbf{x}_k + \mathbf{b}^T\mathbf{x}_k \\
&= -\frac{1}{2}\phi_k\mathbf{g}_k^T A\mathbf{x}_k - \frac{1}{2}\phi_k\mathbf{x}_k^T A\mathbf{g}_k + \frac{1}{2}\phi_k^2\mathbf{g}_k^T A\mathbf{g}_k \\
&\quad + \phi_k\mathbf{b}^T\mathbf{g}_k.
\end{aligned} \tag{32}$$

Using the facts that the gradient of the function (29) is $\mathbf{g}_k = A\mathbf{x}_k - \mathbf{b}$ in conjunction with the equality $\mathbf{b}^T\mathbf{g}_k = \mathbf{g}_k^T\mathbf{b}$, one can verify the following:

$$\begin{aligned}
&f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k) \\
&= -\frac{1}{2}\phi_k\left(\mathbf{g}_k^T A\mathbf{x}_k + \mathbf{x}_k^T A\mathbf{g}_k - \phi_k\mathbf{g}_k^T A\mathbf{g}_k - 2\mathbf{b}^T\mathbf{g}_k\right) \\
&= -\frac{1}{2}\phi_k\left(\mathbf{g}_k^T(A\mathbf{x}_k - \mathbf{b}) + \mathbf{g}_k^T(A\mathbf{x}_k - \mathbf{b}) - \phi_k\mathbf{g}_k^T A\mathbf{g}_k\right) \\
&= -\phi_k\mathbf{g}_k^T\mathbf{g}_k + \frac{1}{2}\phi_k^2\mathbf{g}_k^T A\mathbf{g}_k.
\end{aligned} \tag{33}$$

Substituting (33) into (11), the parameter γ_{k+1} becomes

$$\gamma_{k+1} = 2\frac{-\phi_k\mathbf{g}_k^T\mathbf{g}_k + (1/2)\phi_k^2\mathbf{g}_k^T A\mathbf{g}_k + \phi_k\mathbf{g}_k^T\mathbf{g}_k}{\phi_k^2\mathbf{g}_k^T\mathbf{g}_k} = \frac{\mathbf{g}_k^T A\mathbf{g}_k}{\mathbf{g}_k^T\mathbf{g}_k}. \tag{34}$$

The last relation confirms that γ_{k+1} is the Rayleigh quotient of the real symmetric matrix A at the vector $\mathbf{g}_k$, so the next inequalities hold:

$$\lambda_1 \le \gamma_{k+1} \le \lambda_n, \quad k \in \mathbb{N}, \tag{35}$$

which combined with the fact that $0 \le \alpha_{k+1} \le 1$ prove the right hand side in (30):

$$\phi_{k+1} = \alpha_{k+1}\left(\gamma_{k+1}^{-1} - 1\right) + 1 \le \gamma_{k+1}^{-1} - 1 + 1 = \frac{1}{\gamma_{k+1}} \le \frac{1}{\lambda_1}. \tag{36}$$

The estimation

$$\alpha_k > \frac{\beta(1-\sigma)\gamma_k}{L}, \tag{37}$$

proved in [1], is considered in order to prove the left hand side of (30). Using the notation adopted in this paper, expression (37) becomes

$$\alpha_k > \frac{\eta(1-\sigma)\gamma_k}{L}. \tag{38}$$

Inequality (38) and the facts that $\eta \in (\sigma, 1)$, $\sigma \in (0, 0.5)$ and $\alpha_{k+1} \in (0, 1)$ lead to the following conclusion:

$$\begin{aligned}\phi_{k+1} &= \alpha_{k+1}\left(\gamma_{k+1}^{-1} - 1\right) + 1 > \frac{\alpha_{k+1}}{\gamma_{k+1}} - \alpha_{k+1} \\ &\geq \frac{\alpha_{k+1}}{\gamma_{k+1}} \geq \frac{\eta(1-\sigma)}{L} \geq \frac{(1-\sigma)}{L} \geq \frac{1}{2L}.\end{aligned} \tag{39}$$

In the last estimation, Lipschitz constant L can be replaced by λ_n. The conclusion that the eigenvalue λ_n of matrix A has the property of Lipschitz constant L is to be derived from the next analysis. Since matrix A is symmetric and $\mathbf{g}(\mathbf{x}) = A\mathbf{x} - \mathbf{b}$ the following inequality can be provided:

$$\begin{aligned}\left\|\mathbf{g}(\mathbf{x}) - \mathbf{g}(\mathbf{y})\right\| &= \left\|A\mathbf{x} - A\mathbf{y}\right\| = \left\|A(\mathbf{x} - \mathbf{y})\right\| \leq \|A\| \left\|\mathbf{x} - \mathbf{y}\right\| \\ &= \lambda_n \left\|\mathbf{x} - \mathbf{y}\right\|.\end{aligned} \tag{40}$$

Substituting L by λ_n, inequality (39) becomes

$$\frac{1}{2\lambda_n} \leq \alpha_{k+1}\left(\gamma_{k+1}^{-1} - 1\right) + 1, \tag{41}$$

and this proves the left hand side of inequalities (30). □

Remark 8. Comparing the estimations resulting from the similarly proposed lemma in [1, 3, 4] with the estimation derived from the previous lemma, considering the TADSS method, it can be concluded that the estimation provided by the TADSS scheme involves only the eigenvalues λ_1 and λ_n and not the parameter σ from the backtracking procedure.

Theorem 9. *Let the additional assumptions $\lambda_n < 2\lambda_1$ for the eigenvalues of matrix A be imposed and let f be the strictly convex quadratic function given by (29). Assume $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ is the orthonormal eigenvectors of symmetric positive definite matrix A and suppose that $\{\mathbf{x}_k\}$ is the sequence of values constructed by Algorithm 2. The gradients of convex quadratics defined by (29) are $\mathbf{g}_k = A\mathbf{x}_k - \mathbf{b}$ and can be expressed as*

$$\mathbf{g}_k = \sum_{i=1}^{n} d_i^k \mathbf{v}_i, \tag{42}$$

for some real constants $d_1^k, d_2^k, \ldots, d_n^k$ and for some integer k. Then the application of the gradient descent method (7) on the goal function (29) satisfies the following two statements:

$$\left(d_i^{k+1}\right)^2 \leq \delta^2 \left(d_i^k\right)^2, \quad \delta = \max\left\{1 - \frac{\lambda_1}{2\lambda_n}, \frac{\lambda_n}{\lambda_1} - 1\right\}, \tag{43}$$

$$\lim_{k \to \infty} \left\|\mathbf{g}_k\right\| = 0. \tag{44}$$

Proof. Taking into account (7) one can verify

$$\mathbf{g}_{k+1} = A\left(\mathbf{x}_k - \phi_k \mathbf{g}_k\right) - \mathbf{b} = \mathbf{g}_k - \phi_k A\mathbf{g}_k = \left(I - \phi_k A\right)\mathbf{g}_k, \tag{45}$$

and by taking (42) we get

$$\mathbf{g}_{k+1} = \sum_{i=1}^{n} d_i^{k+1}\mathbf{v}_i = \sum_{i=1}^{n} \left(1 - \phi_k \lambda_i\right) d_i^k \mathbf{v}_i. \tag{46}$$

In order to prove (43), it is enough to show that $|1 - \phi_k\lambda_i| \leq \delta$. So, two cases have to be analyzed. In the first one, it is supposed that $\lambda_i\phi_k \leq 1$. Applying (30) leads to

$$1 > \lambda_i\phi_k \geq \frac{\lambda_1}{2\lambda_n} \Longrightarrow 1 - \lambda_i\phi_k \leq 1 - \frac{\lambda_1}{2\lambda_n} \leq \delta. \tag{47}$$

In the other case, it is assumed that $\lambda_i\phi_k \geq 1$. From this condition arrives the following conclusion:

$$1 < \lambda_i\phi_k \leq \lambda_n \frac{1}{\lambda_1} \Longrightarrow \left|\lambda_i\phi_k - 1\right| \leq \frac{\lambda_n}{\lambda_1} - 1 \leq \delta. \tag{48}$$

Expression (42) implies

$$\left\|\mathbf{g}_k\right\|^2 = \sum_{i=1}^{n} \left(d_i^k\right)^2. \tag{49}$$

The fact that the parameter δ, from (43), satisfies $0 < \delta < 1$ confirms expression (44). □

4. Numerical Experience

Numerical results provided by applying the implementation of TADSS, ADSS, and SM methods on 22 test functions for unconstrained test problems, proposed in [2, 11], are presented and investigated. We chose most of the functions from the set of test functions presented in [3, 4] and, as proposed in these papers, also investigated the experiments with a large number of variables in each function: 1000, 2000, 3000, 5000, 7000, 8000, 10000, 15000, 20000, and 30000. The stopping criteria are the same as in [1, 3, 4]. Backtracking procedure is developed using the values $\sigma = 0.0001$, $\eta = 0.8$ of needed parameters. Three main indicators of the efficiency are observed: number of iterations, CPU time, and number of function evaluations. First, we compare the performance of the TADSS scheme with the ADSS method. The reasons for this selection is obvious: the TADSS scheme presents a one-step version of ADSS method. Also, the intention to examine behavior of TADSS and compare it with its forerunner is natural. Obtained numerical values are displayed in Table 1 and refer to the number of iterative steps, the CPU time of executions computed in seconds, and the number of evaluations of the objective function.

Obtained numerical results, generally, confirm advantages in favor of TADSS, considering all three tested indicators. More precise, regarding the number of iterative steps TADSS shows better results in 17 out of 22 functions, while ADSS outperforms TADSS in 4 out of 22 experiments and for the extended three exponential terms function both methods require the same number of iterations. Results concerning spanned CPU time confirm that both methods, TADSS and ADSS, are very fast. In 9 out of 22 cases TADSS is faster than ADSS, in 2 out of 22 testings ADSS is faster than TADSS, and

Table 1: Summary of numerical results for TADSS and ADSS tested on 22 large scale test functions.

Test function	Number of iterations		CPU time		Number of function evaluations	
	TADSS	ADSS	TADSS	ADSS	TADSS	ADSS
Extended Penalty	40	50	2	4	1280	1780
Raydan 1	466	34	11	4	8504	4844
Diagonal 1	20	37	0	0	335	1448
Diagonal 3	21	49	0	1	417	1048
Generalized Tridiagonal 1	61	77	0	0	431	719
Extended Tridiagonal 1	60	70	0	0	250	420
Extended Three Expon. Term	40	40	0	0	400	350
Diagonal 4	40	780	0	0	270	2590
Extended Himmelblau	60	70	0	0	300	480
Quadratic QF1	4953	425	15	0	13738	1755
Extended Quad. Penalty QP1	50	60	0	2	571	841
Extended Quad. Penalty QP2	50	60	0	4	569	843
Quadratic QF2	50	60	0	1	583	836
Extended EP1	186	40	1	0	758	487
Extended Tridiagonal 2	638	80	0	0	2052	420
Arwhead	50	64	0	3	601	1082
Engval1	60	70	0	0	290	460
Quartc	10	70	0	0	30	390
Generalized Quartic	60	70	0	0	250	614
Diagonal 7	189	2201	0	33	566	6633
Diagonal 8	160	2213	1	40	586	6709
Diagonal 9	20	43	0	0	352	3646

Table 2: Average numerical outcomes of 220 testings of each method among the 22 test functions tried out on 10 numerical experiments in each iteration.

Average performances	TADSS	ADSS
Number of iterations	331.09	302.86
CPU time (sec)	1.36	4.18
Number of function evaluations	1506.05	1745.23

even in half of examinations the CPU time of both iterations equals zero. On the issue of the number of evaluations of an objective function, TADSS improves ADSS in 17 out of 22 testings and the opposite case appears in 5 out of 22 cases. Table 2 displays average results of tested values.

According to results displayed in Table 2, it can be concluded that although TADSS outperforms ADSS in 17 out of 22 testings with regard to the number of iterations, average results show slight advantage of ADSS on this matter. Considering the average number of evaluations, there is an opposite case in favor of TADSS. Consumed CPU time is averagely three times less in favor to the TADSS comparing to the ADSS. Generally, it can be concluded that the one-step variant of the ADSS method, constructed TADSS scheme, behaves slightly better than the original ADSS iteration, especially when we consider the speed of executions.

Some additional experiments have been carried out in further numerical research. These tests show the comparison between the TADSS and the SM iterations. As mentioned before, both of the schemes, TADSS and SM, are accelerated gradient descent methods with one iterative step size parameter. We choose this additional numerical comparison in order to confirm that the accelerated single step size TADSS algorithm, derived from the accelerated double step size ADSS model, gives better performances with respect to the all three analyzed aspects than the classically defined accelerated single step size SM method. Table 3 with 30 displayed test functions verifies the previous assertion.

It can be observed from displayed numerical outcomes that the TADSS method provides better results than the SM method considering the number of iterations in 24 out of 30 testings, while the number of opposite cases is 5 out of 30. For the NONSCOMP test function, both models have the same number of iterations. Concerning the CPU time, both algorithms give the same results for 10 test functions. The TADSS method is faster than SM for 19 test functions, while the SM method is faster than TADSS for one test function only. The greatest progress is obtained with respect to the number of evaluations of the objectives. On this matter, using the TADSS algorithm, better results are obtained in 27 out of 30 test functions, while the opposite case holds for two test functions only. For the NONSCOMP test function both of the compared iterations give the same number of

Table 3: Numerical results for 30 test functions tested by the TADSS and the SM methods.

Test functions	Number of iterations		CPU time		Number of function evaluations	
	TADSS	SM	TADSS	SM	TADSS	SM
Extended Penalty	40	589	2	5	1280	2987
Perturbed Quadratic	4276	111972	13	1868	12017	632724
Raydan 1	466	21125	11	178	8504	116757
Diagonal 1	20	10417	0	116	335	56135
Diagonal 3	21	10574	0	209	417	59425
Generalized Tridiagonal 1	61	278	0	2	431	989
Extended Tridiagonal 1	60	3560	0	35	250	30686
Extended Three Expon. Term	40	164	0	1	400	1224
Diagonal 4	40	80	0	0	270	530
Extended Himmelblau	60	168	0	0	300	566
Quadr. Diag. Perturbed	11542	53133	58	1193	56359	547850
Quadratic QF1	4953	114510	15	2127	13738	649643
Extended Quad. Penalty QP1	50	224	0	12	571	2566
Extended Quad. Penalty QP2	50	162	0	7	569	2057
Quadratic QF2	50	118801	0	2544	583	662486
Extended EP1	186	68	1	1	758	764
Extended Tridiagonal 2	638	584	0	0	2052	2144
Arwhead	50	10	0	0	601	30
Almost Perturbed Quadratic	4202	110121	15	2148	14974	627287
Engval1	60	185	0	7	290	2177
Quartc	10	190	0	0	30	430
Generalized Quartic	60	156	0	0	250	423
Diagonal 7	189	90	0	0	566	220
Diagonal 8	160	96	1	0	586	594
Diagonal 9	20	11235	0	118	352	60566
DIXON3DQ	10	112899	0	1908	30	639957
NONSCOMP	10	10	0	0	30	30
HIMMELH	10	30	0	0	40	80
Power Cute	1455	$>t_e$	4	$>t_e$	4567	$>t_e$
Indef	20	$>t_e$	0	$>t_e$	50	$>t_e$

Table 4: Average values of numerical results for the TADSS and the SM methods calculated on 280 tests for each method.

Average performance	TADSS	SM
Number of iterations	976.21	24336.82
CPU time (in seconds)	4.14	445.68
Number of evaluations	4163.68	146475.96

evaluations. From Table 3, we can also notice that for 2 out of 30 test functions testings are lasting more than the time limiter constant t_e defined in [3], while for all 30 test functions when the TADSS algorithm is applied the time of execution is far less than t_e.

The results arranged in Table 4 give even more general view on the benefits provided by applying the TADSS method with regard to the SM method. The average values of 28 test functions, which were possible to test by both methods according to the constant t_e, are presented in the table.

The results presented in the previous table confirm that by applying the TADSS method approximately 25 times less iterations and even 35 times less evaluations of the objective function are needed in comparison with the SM method. Finally, when the TADSS is used, the testing is lasting even 107 times shorter than when the SM is applied.

The codes for presented numerical experiments are written in the Visual C++ programming language on a Workstation Intel 2.2 GHz.

5. Conclusion

The accelerated single step size gradient descent algorithm, called TADSS, is defined as a transformation of the accelerated double step size gradient descent model ADSS, proposed in [4]. More precisely, the TADSS scheme is derived from the

accelerated double step size gradient descent scheme ADSS by imposing relation (6) between two step parameters in the ADSS iteration. The efficiency of ADSS model regarding all analyzed characteristics in comparison to the accelerated gradient descent single step size SM method has been numerically proved in [4].

The method defined in this way is comparable with its double step size forerunner, ADSS method, as well as with the single step size accelerated gradient descent SM method which is defined in a classical manner.

Results illustrated in Tables 1 and 2 generally indicate that the TADSS method behaves similarly as the ADSS method. From the point of mean values, the ADSS scheme gives slightly better results considering the number of iterations. On the other hand, a certain improvement regarding the number of function evaluations and needed CPU time is obtained by applying the TADSS iterations.

Even greater advantages of derived TADSS method are presented in Tables 3 and 4 where the comparisons between the TADSS and the SM are given. Evidently, the TADSS scheme improves the SM method with respect to all three analyzed characteristics, which was the prime goal in the research presented herein.

Linear convergence of the TADSS method is proved for the uniformly convex and the strictly convex quadratic functions.

Obtained results motivate further investigations of possible accelerated double step size gradient descent models and its transformations into corresponding single step size variants.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The authors gratefully acknowledge the financial support of the Serbian Ministry of Education, Science and Technological Development.

References

[1] P. S. Stanimirović and M. B. Miladinović, "Accelerated gradient descent methods with line search," *Numerical Algorithms*, vol. 54, no. 4, pp. 503–520, 2010.

[2] N. Andrei, "An acceleration of gradient descent algorithm with backtracking for unconstrained optimization," *Numerical Algorithms*, vol. 42, no. 1, pp. 63–73, 2006.

[3] M. J. Petrović and P. S. Stanimirović, "Accelerated double direction method for solving unconstrained optimization problems," *Mathematical Problems in Engineering*, vol. 2014, Article ID 965104, 8 pages, 2014.

[4] M. J. Petrović, "An accelerated double step size model in unconstrained optimization," *Applied Mathematics and Computation*, vol. 250, pp. 309–319, 2015.

[5] N. I. Djuranović-Miličić and M. Gardaševic-Filipović, "A multi-step curve search algorithm in nonlinear optimization: nondifferentiable convex case," *Facta Universitatis, Series: Mathematics and Informatics*, vol. 25, pp. 11–24, 2010.

[6] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equation in Several Variables*, Academic Press, London, UK, 1970.

[7] R. T. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, NJ, USA, 1970.

[8] J. Barzilai and J. M. Borwein, "Two-point step size gradient methods," *IMA Journal of Numerical Analysis*, vol. 8, no. 1, pp. 141–148, 1988.

[9] Y.-H. Dai and L.-Z. Liao, "*R*-linear convergence of the Barzilai and Borwein gradient method," *IMA Journal of Numerical Analysis*, vol. 22, no. 1, pp. 1–10, 2002.

[10] B. Molina and M. Raydan, "Preconditioned Barzilai-BORwein method for the numerical solution of partial differential equations," *Numerical Algorithms*, vol. 13, no. 1-2, pp. 45–60, 1996.

[11] N. Andrei, "An unconstrained optimization test functions collection," *Advanced Modeling and Optimization*, vol. 10, pp. 147–161, 2008.