Iranian Journal of Numerical Analysis and Optimization
Vol. 14, No. 3, 2024, pp 970–990
https://doi.org/10.22067/ijnao.2024.86481.1379
https://ijnao.um.ac.ir/

Research Article

A stabilized simulated annealing-based Barzilai–Borwein method for the solution of unconstrained optimization problems

H. Sharma* and R.K. Nayak

Abstract

The Barzilai–Borwein method offers efficient step sizes for large-scale unconstrained optimization problems. However, it may not guarantee global convergence for nonquadratic objective functions. Simulated annealing-based on Barzilai–Borwein (SABB) method addresses this issue by incorporating a simulated annealing rule. This work proposes a novel step-size strategy for the SABB method, referred to as the SABBm method. Furthermore, we introduce two stabilized variants: SABB*stab* and SABB*mstab*. SABB*stab* combines a simulated annealing rule with a stabilization step to ensure convergence. SABB*mstab* builds upon SABB*stab*,

*Corresponding author

Received 19 January 2024; revised 08 May 2024; accepted 11 May 2024

Hitarth Sharma
Department of Mathematics, International Institute of Information Technology, Bhubaneswar, Odisha, India, 751029. e-mail: hitarth@iiit-bh.ac.in

Rupaj Kumar Nayak
Department of Mathematics, International Institute of Information Technology, Bhubaneswar, Odisha, India, 751029. e-mail: rupaj@iiit-bh.ac.in

How to cite this article
Sharma, H. and Kumar Nayak, R., A stabilized simulated annealing-based Barzilai–Borwein method for the solution of unconstrained optimization problems. *Iran. J. Numer. Anal. Optim.*, 2024; 14(3): 970-990. https://doi.org/10.22067/ijnao.2024.86481.1379

incorporating the modified step size derived from the SABBm method. The effectiveness and competitiveness of the proposed methods are demonstrated through numerical experiments on CUTEr benchmark problems.

AMS subject classifications (2020): Primary 65K05; Secondary 90C06, 90C30.

Keywords: Unconstrained optimization; Barzilai–Borwein method; Simulated annealing method; Stabilized BB method.

1 Introduction

Researchers have shown considerable interest in unconstrained optimization problems due to their significant theoretical importance and practical applicability in the field of optimization. Its applications span various fields, including engineering, physics, finance, machine learning, and more. Furthermore, it is applicable for addressing a range of problems, including parameter estimation, function fitting, optimization of cost functions, and various others. The general form of an unconstrained optimization problem is

$$\min_{x\in\mathbb{R}^n} f(x), \tag{1}$$

where $f:\mathbb{R}^n\to\mathbb{R}$ is continuously differentiable. The iterative formula in the classical steepest-descent method [6] for the problem (1) is of the form

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

where the search direction $d_k\in\mathbb{R}^n$ is determined as the negative gradient of f at x_k as

$$d_k = -\nabla f(x_k), \tag{3}$$

and the step size α_k is determined by

$$\alpha_k = \arg\min_{\alpha} f(x_k+\alpha d_k). \tag{4}$$

The above method is simple. However, it performs poorly as it exhibits linear convergence and is influenced by ill-conditioning [1]. Barzilai and Borwein [3] introduced two novel step sizes to be utilized together with the direction

of the negative gradient. Equation (2) requires less computation than (3), and the algorithm was also less sensitive to ill-conditioning. The step size α_k of the Barzilai–Borwein (BB) method [3] is given by

$$\alpha_k = \frac{s_{k-1}^\top s_{k-1}}{s_{k-1}^\top y_{k-1}} \tag{5}$$

or

$$\tilde{\alpha}_k = \frac{s_{k-1}^\top y_{k-1}}{y_{k-1}^\top y_{k-1}}, \tag{6}$$

where $s_{k-1} = x_k - x_{k-1}$, $y_{k-1} = \nabla f_k - \nabla f_{k-1}$. This approach has an R-superlinear convergence rate for a two-dimensional strictly convex quadratic objective function, and it outperforms the linearly convergent classical steepest-descent technique [6]. Subsequently, Raydan [20] established the global convergence of this method for n-dimensional strictly convex quadratic objective functions. The R-linear convergence rate was demonstrated by Dai and Liao [7]. On the other hand, Fletcher [11] contended that R-linear convergence is only anticipated in general cases. It is further confirmed that the BB method does not guarantee global convergence when the objective function is nonquadratic unless it is integrated with certain globalizing schemes.

The first nonmonotone line search framework for Newton's methods was presented by Grippo, Lampariello, and Lucidi [13], which has been applied in a number of later papers on nonmonotone line search techniques (see [4, 14, 16, 26]). Furthermore, nonmonotone line search approaches are also used by certain spectral gradient methods for optimization problems (see [23, 25]). Although these techniques are often effective, they do have certain drawbacks. Specifically, in the nonmonotone line search technique, the maximum value discards any good function value generated in an iteration. Furthermore, as shown in [13, 22], the selection of integer $M(M > 0)$ can have a significant impact on the numerical performance of nonmonotonic line search algorithms. Dai and Zhang [8] created an adaptive nonmonotone line search to overcome these two shortcomings, which is utilized in conjunction with the two-point gradient approach for optimization problems. A novel nonmonotone line search technique was later proposed by Zhang and Hager

[24], which is shown by numerical experiments to perform better than both monotone and traditional nonmonotone strategies.

Kirkpatrick, Gelatt, and Vecchi [15] first used the Simulated Annealing (SA) method initially introduced by Metropolis et al. [17], for addressing combinatorial optimization problems. Numerous authors have thoroughly examined the later SA approach for both discrete and continuous optimization issues. The SA approach is popular among researchers because it is capable of avoiding trapping in local minima. However, for large-scale problems, it is not acceptable because of the enormous computing cost.

Numerous hybrid algorithms that combine the SA method with other optimization techniques have been published, owing to their theoretical guarantee of convergence, enhanced performance in numerous real-world situations, and ease of implementation. Dong, Li, and Peng [10] presented a hybrid method that combines the BB method and the SA method. This nonmonotonic technique is called the Simulated Annealing-Based Barzilai–Borwein (SABB) method. Global convergence is also established under certain moderate assumptions in their research.

Numerous researchers have observed that the BB method might produce steps that deviate iterations too far from the optimal solution. Furthermore, it fails to converge even for strongly convex functions. The stabilized BB method, as proposed by Burdakov, Dai, and Huang [5], is a stabilized version of the BB method. The approach involves constraining the distance between each pair of consecutive iterates, a strategy that frequently reduces the number of BB iterations. In [5], the global convergence of this approach is also demonstrated for strongly convex functions with Lipschitz gradients. Barzilai and Borwein [3], followed by Raydan [20], demonstrated global convergence for the case of a strictly convex quadratic function, irrespective of the number of variables involved.

While SABB outperforms other similar algorithms, such as the adaptive two-point step-size gradient algorithm by Dai and Zhang [8] and the Barzilai–Borwein gradient method (GBB) by Raydan [21], it can generate too long steps, which may lead to the discarding of significant intermediate iterates. Motivated by this challenge, we introduce some novel variants of the SABB method.

In this paper, we first briefly discuss some existing methods to solve unconstrained optimization problems in Section 1. Then, in Section 2, we propose a modified version of the SABB method referred to as the SABBm method and two stabilized versions of the SABB method referred to as SABB*stab* and SABB*mstab*, respectively. In Section 3, we present the numerical results obtained through rigorous experimentation and analysis. This section provides valuable insights into the effectiveness and efficiency of the proposed variants. Finally, Section 4 provides a concise summary of the conclusions derived from our extensive research findings. Within this section, we deliberate on the significance of each variant and its implications concerning unconstrained optimization problems.

1.1 Simulated annealing-based Barzilai–Borwein (SABB) method

When applied to nonquadratic objective functions, the original BB technique does not provide global convergence. To tackle this problem, the BB method was combined with the SA approach and introduced as the SABB method by the authors [10]. In order to approve the BB step, the method incorporates an SA criterion. If the BB step is deemed unacceptable, then an Armijo line search method is employed. Under certain mild conditions, the global convergence of the SABB technique is established. The BB step of the SABB method given by Dong, Li, and Peng [10] is

$$\alpha_k = \max\left\{\alpha_{\min}, \min\left\{\alpha_{\max}, \frac{s_{k-1}^\top s_{k-1}}{s_{k-1}^\top y^{k-1}}\right\}\right\}, \tag{7}$$

where $s_{k-1} = x_k - x_{k-1}, y_{k-1} = \nabla f_k - \nabla f_{k-1}$ and $\alpha_{\min}, \alpha_{\max} > 0$ are two fixed parameters.

1.2 Stabilized BB (BB*stab*) method

Occasionally, the BB approach produces excessively long steps, causing the iterates to deviate too far from the optimal solution. The objective function might not converge even if it is strongly convex. In that situation, a basic modification can make it convergent, and the stabilized version preserves the rapid local convergence of the BB method as the number of stabilization steps is established to be finite. The step size of the stabilized BB method given by Burdakov, Dai, and Huang [5] is

$$\alpha_k = \min\{\alpha_k^{\mathrm{BB}}, \alpha_k^{\mathrm{Stab}}\} \tag{8}$$

where

$$\alpha_k^{BB} = \frac{s_{k-1}^{\top} s_{k-1}}{s_{k-1}^{\top} y_{k-1}} \tag{9}$$

with $s_{k-1} = x_k - x_{k-1}, y_{k-1} = \nabla f_k - \nabla f_{k-1} = g_k - g_{k-1}$, and

$$\alpha_k^{\mathrm{Stab}} = \frac{D}{\|g_{k-1}\|}, \tag{10}$$

where $D > 0$ is a parameter to be chosen in a particular way.

2 Variants of the SABB method

In this section, we introduce our proposed variants of the SABB method. The step-size selection strategy within SABB is critical for achieving efficient convergence towards the minimum. Therefore, the proposed SABB variants are distinguished by their step-size selection criteria.

2.1 Modified SABB (SABB*m*) method

Mu and Liu [18] employed a BB method to solve the unconstrained optimization subproblem arising within the Augmented Lagrangian Method (ALM) for large-scale binary quadratic programming. This choice leverages the BB method's low computational cost and effectiveness in achieving near-optimal

solutions, making the resulting ALM approach a viable option for these large-scale problems. Here, we propose a new step size for the SABB method, which is a modified version of the step size given in [18]. The step size of our method has an additional parameter $\exp(d)$, where $d \in (0,1)$. The inclusion of $\exp(d)$ and $\xi^{l_{\min}}$ in the SABB step size is motivated by their demonstrated effectiveness. They enable the modified step size to achieve the same results in fewer iterations and with reduced computation time. Here we introduce a new step for the SABB technique, which is presented as

$$\alpha_k^{\text{SABBm}} = \begin{cases} \exp(d)\xi^{l_{\min}}\alpha_0 & \text{if } k = 0, \\ \exp(d)\xi^{l_{\min}} \min\left\{\alpha_{\max}, \max\left\{\alpha_{\min}, \frac{\|s_k\|_2^2}{s_k^\top y_k}\right\}\right\} & \text{if } k \geq 1, \end{cases} \tag{11}$$

where $\alpha_{\min}, \alpha_{\max} > 0$ are two fixed parameters, $s_k = x_{k+1} - x_k$, $y_k = \nabla f_{k+1} - \nabla f_k$, ξ is the given parameter, the chosen parameters $d, \sigma \in (0,1)$, and $l_{\min}$ is the smallest nonnegative integer l satisfying

$$f(x_k + \xi^l \alpha_k \nabla f_k) \leq f(x_k) + \sigma \xi^l \nabla^\top f_k \alpha_k \nabla f_k, \tag{12}$$

where

$$\alpha_k = -\min\left\{\alpha_{\max}, \max\left\{\alpha_{\min}, \frac{\|s_k\|_2^2}{s_k^\top y_k}\right\}\right\}.$$

The approach used in (12) to obtain the value of $l_{\min}$ is inspired by the nonmonotone line search framework developed by Grippo, Lampariello, and Lucidi [13]. We now propose the SABBm as follows:

2.1.1 Convergence of SABBm method

To establish global convergence, we require the following remarks and assumptions.

Remark 1. Armijo line search: let m_k be the smallest integer that satisfies $f(x_k - \delta^m \nabla f_k) \leq f_k - c\delta^m \|\nabla f_k\|^2$ where $\delta \in (0,1)$. Then $\alpha_{k+1}^{\text{SABB}m} = \delta^{m_k}$.

Assumption 1. The set $Z_0 = \{x \in \mathbb{R}^n : f(x) \leq f(x_0) + (1-\gamma)^{-1} T_0 \eta\}$ is bounded and closed.

Lemma 1. The step size $\alpha_k^{\text{SABB}m}$ in the SABBm method satisfies the condition $\alpha_k^{\text{SABB}m} \geq \alpha$, for all $k \geq 0$, where $\alpha \geq 0$ is a constant.

Algorithm 5: SABBm method

1: **Give an initial point** $x_0 \in \mathbb{R}^n$. Let $\epsilon > 0$, $0 < \alpha_{\min} < \alpha_{\max} < \infty$, $\alpha_0 \in (\alpha_{\min}, \alpha_{\max})$, $T_0 > 0, c, \gamma, \xi, d, \sigma \in (0,1), \eta \in \mathbb{Z}_+$. Set $k := 0$ and $\alpha_0^{\text{SABBm}} = \exp(d)\xi^{l_{\min}}\alpha_0$

2: If $\|\nabla f_k\| < \epsilon$, then stop

3: Compute $z_k = x_k - \alpha_k^{\text{SABB}m}\nabla f_k$
and $\Delta f_k = f(z_k) - (f_k - c\alpha_k^{\text{SABBm}}\|\nabla f_k\|^2)$

4: Let $p_k = e^{-\frac{\Delta f_k}{T_k}}$ and pick a random number $r_k \in \left(e^{-\eta}, e^{-\frac{1}{\eta}}\right)$.

5: If $p_k \geq r_k$, let $x_{k+1} = z_k$ and go to step 7.

6: Otherwise, let $\alpha_k^{\text{SABB}m}$ be a step size determined by the Armijo line search and $x_{k+1} = x_k - \alpha_k^{\text{SABB}m}\nabla f_k$.

7: Compute $\alpha_{k+1}^{\text{SABB}m}$ using (11).

8: Let $T_{k+1} = \gamma T_k; k = k+1$ and go to step 2.

Proof. There are two cases for the step size of the SABBm method: one is by the Armijo line search, while the other is by (11). In the case of the Armijo line search method from [19], by the property of termination in finite steps, there exists an integer $N > 0$ such that $\alpha_k^{\text{SABB}m} > \delta^N$, for all $k \geq 0$. In the second case, from (11), it is clear that $\alpha_k^{SABB}m > \alpha_{\min}$. Let $\alpha = \min\{\delta^N, \alpha_{\min}\}$. The result is obtained, and this concludes the proof. □

Lemma 2. If Assumption 1 holds, then the SABBm method is well defined, and the sequence generated by the SABBm method is $\{x_k\} \subset Z_0$.

Proof. In Algorithm 5, we consider $p_k = e^{-\frac{\Delta f_k}{T_k}}$, and if $p_k \geq r_k$, then $\Delta f_k \leq -T_k \ln r_k$, which implies

$$f(z_k) \leq f_k - c\alpha_k^{\text{SABB}m}\|\nabla f_k\|^2 - T_k \ln r_k.$$

Let

$$\mu_k = \begin{cases} 1 & \text{if } p_k \geq r_k, \\ 0 & \text{otherwise.} \end{cases}$$

Then, the functional value sequence $\{f_k\}$ generated by SABBm satisfies

$$f_{k+1} \leq f_k - c\alpha_k^{\text{SABB}m}\|\nabla f_k\|^2 - \mu_k T_k \ln r_k. \tag{13}$$

Since $e^{-\eta} < r_k < e^{-\frac{1}{\eta}}$ and $\left(e^{-\eta}, e^{-\frac{1}{\eta}}\right) \subset (0,1)$, we have $\ln r_k < -\frac{1}{\eta} < 0$. Again since, $0 \leq \mu_k \in \{0,1\}$ and $T_k > 0$, we have $-\mu_k T_k \ln r_k \geq 0$. Hence, if the step size $\alpha_k^{\text{SABB}m}$ is determined by the Armijo line search method, we get

$$f_{k+1} \leq f_k - c\alpha_k^{\text{SABB}m} \|\nabla f_k\|^2. \tag{14}$$

Thus, one can conclude that (14) implies (13). Therefore, based on the termination property within a finite number of steps of the Armijo line search [19], the method is well defined.
By (13), we get

$$\begin{aligned}
f_{k+1} &\leq f_k - c\alpha_k^{\text{SABB}m} \|\nabla f_k\|^2 - \mu_k T_k \ln r_k \\
&\leq f_k - \mu_k T_k \ln r_k \\
&\leq f_k - T_k \ln r_k (\text{as } \mu_k = 1 \text{ if } p_k \geq r_k; \text{ otherwise } \mu_k = 0) \\
&= f_0 - \sum_{i=1}^{k} T_i \ln r_i \\
&= f_0 - \sum_{i=1}^{k} \gamma^i T_0 \ln r_i \\
&\leq f_0 - T_0 \eta \sum_{i=1}^{k} \gamma^i \\
&< f_0 + (1-\gamma)^{-1} T_0 \eta
\end{aligned}$$

for all $k \geq 0$. Therefore, $\{x_k\} \subset Z_0$. □

Here is the proof of the global convergence of the SABBm method.

Theorem 1. Under Assumption 1, $\lim_{k\to\infty} \|\nabla f(x_k)\| = 0$, where the sequence $\{x_k\}$ is generated by the SABBm method.

Proof. By Lemma 2, we have $\{x_k\} \subset Z_0$. Since Z_0 is compact, the sequence $\{x_k\}$ is convergent. By (13), for $k \geq 0$

$$\begin{aligned}
f_{k+1} &\leq f_k - c\alpha_k^{\text{SABB}m} \|\nabla f_k\|^2 - \mu_k T_k \ln r_k \\
&\leq f_k - c\alpha_k^{\text{SABB}m} \|\nabla f_k\|^2 - T_k \ln r_k,
\end{aligned}$$

where $-T_k \ln r_k > 0$ and $\mu_k \in \{0,1\}$. Combining this with Lemma 1, we get

$$c\alpha\|\nabla f_k\|^2 \leq c\alpha_k^{\mathrm{SABB}m}\|\nabla f_k\|^2 \leq f_k - f_{k+1} - T_k \ln r_k. \tag{15}$$

Summarizing (15) from $k = 0$ to K, we get

$$c\alpha \sum_{k=0}^{K}\|\nabla f_k\|^2 \leq f_0 - f_{K+1} - \sum_{k=0}^{K} T_k \ln r_k. \tag{16}$$

By Assumption 1 and the continuity of $f(x)$ on Z_0, $f(x) \geq \beta$ holds for all $x \in Z_0$, where $\beta < \infty$ is a real number. Therefore, by taking limits on (16) as $K \to \infty$, we obtain

$$\begin{aligned}
\sum_{k=0}^{\infty}\|\nabla f_k\|^2 &= \sum_{k\geq 0}\|\nabla f_k\|^2 \\
&\leq \frac{f_0 - f_{K+1} - \sum_{k\geq 0} T_k \ln r_k}{c\alpha} \\
&= \frac{f_0 - f_{K+1} - \sum_{k\geq 0} \gamma^k T_0 \ln r_k}{c\alpha} \\
&\leq \frac{f_0 - f_{K+1} + T_0\eta \sum_{k\geq 0} \gamma^k}{c\alpha} \\
&\leq \frac{f_0 - f_{K+1} + (1-\gamma)^{-1}\eta T_0}{c\alpha} \\
&\leq (f_0 - \beta + (1-\gamma)^{-1}\eta T_0)/c\alpha.
\end{aligned}$$

□

2.2 Stabilized SABB (SABB*stab*) method

The method proposed here combines the SABB method from [10] with the BB*stab* method from [5], resulting in a modified version of the BB*stab* method [5]. Two-step size values, namely α_k^{SABB} and α_k^{stab}, are computed using (7) and (10), respectively, and their minimum value is used as the required step size $\alpha_k^{\mathrm{SABB}stab}$. Note that it utilizes two distinct initial values x_0 and x_1 instead of that of SABB and SABBm. We now present the SABB*stab* algorithm.

Algorithm 6: SABB*stab* method

1: Give two initial points $x_0, x_1 \in \mathbb{R}^n$ such that $x_0 \neq x_1$ and a scalar $\Delta > 0$. Let $\epsilon > 0, c \in (0,1), 0 < \alpha_{\min} < \alpha_{\max} < \infty$, $\alpha_1 \in [\alpha_{\min}, \alpha_{\max}], T_0 > 0, \gamma \in (0,1), \eta \in \mathbb{Z}_+$. Set $k := 1$

2: If $\|\nabla f_k\| < \epsilon$, then stop

3: Compute $z_k = x_k - \alpha_k \nabla f_k$ and $\Delta f_k = f(z_k) - (f_k - c\alpha_k \|\nabla f_k\|^2)$

4: Let $p_k = e^{-\frac{\Delta f_k}{T_k}}$ and pick a random number $r_k \in \left(e^{-\eta}, e^{-\frac{1}{\eta}}\right)$.

5: If $p_k \geq r_k$ let $x_{k+1} = z_k$ and go to Step 6. Otherwise, let α_k^{BB} be a step size determined by the Armijo line search and $x_{k+1} = x_k - \alpha_k^{BB} \nabla f_k$.

6: Compute $\alpha_{k+1}^{\mathrm{BB}} = \max\left\{\alpha_{\min}, \min\left\{\alpha_{\max}, \frac{s_k^\top s_k}{s_k^\top y^k}\right\}\right\}$, where $s_k = x_{k+1} - x_k, y_k = \nabla f_{k+1} - \nabla f_k$ and Compute $\alpha_{k+1}^{\mathrm{stab}} = \frac{\Delta}{\|\nabla f_k\|}$.

7: Compute $\alpha_{k+1} = \min\{\alpha_{k+1}^{\mathrm{BB}}, \alpha_{k+1}^{\mathrm{stab}}\}$

8: Let $T_{k+1} = \gamma T_k; k = k+1$ and go to first Step 3.

2.2.1 Convergence of SABB*stab* Method

Assumption 2. There exist positive constants $\Lambda_1 \leq \Lambda_2$, and the function $f : \mathbb{R}^n \to \mathbb{R}^1$ is twice continuously differentiable such that

$$\Lambda_1 \|\nabla v\|^2 \leq \ v^\top \nabla^2 f(x) v \leq \Lambda_2 \|v\|^2 \ \text{ for all } \ x, v \in \mathbb{R}^n.$$

Assumption 3. For some $\rho > 0$ and $L \geq 0$, the following property holds:

$$\left\|\nabla^2 f(x) - \nabla^2 f(x^*)\right\| \leq \ L\|x - x^*\| \ \text{ for all } \ x \in \ B_\rho(x^*),$$

where L is a Lipschitz constant, ρ is a radius, and $B_\rho(x^*) = \{x \in \mathbb{R}^n : \|x - x^*\| \leq \rho\}$.

If Assumptions 1, 2, and 3 hold, then our proposed method SABB*stab* is well defined by the Lemma 2. Note that in Algorithm 6, if $\alpha_{k+1} = \alpha_{k+1}^{stab}$, then it is convergent, according to [5, Theorem 3.2] method, and if $\alpha_{k+1} = \alpha_{k+1}^{\mathrm{BB}}$, then it is convergent, according to [10, Theorem 1].

2.3 Modified stabilized SABB (SABB*mstab*) method

In this section, we present a stabilized version of the SABBm method termed SABB*mstab*, which is our main proposal. Similar to the previous method, this approach also utilizes two different initial values, x_0 and x_1. In this method, we propose to choose

$$\alpha_k = \min\{\alpha_k^{SABB}m, \alpha_k^{stab}\}.$$

We now present the SABB*mstab* algorithm below.

Algorithm 7: SABB*mstab* method

1: **Give initial points** $x_0, x_1 \in \mathbb{R}^n$ such that $x_0 \neq x_1$ and a scalar $\Delta > 0$. Let $\epsilon > 0, c \in (0,1), 0 < \alpha_{\min} < \alpha_{\max} < \infty$, $\alpha_1 \in [\alpha_{\min}, \alpha_{\max}], T_0 > 0, \gamma \in (0,1), \eta \in \mathbb{Z}_+$. Set $k := 1$

2: If $\|\nabla f_k\| < \epsilon$, then stop

3: Compute $z_k = x_k - \alpha_k \nabla f_k$ and $\Delta f_k = f(z_k) - (f_k - c\alpha_k \|\nabla f_k\|^2)$

4: Let $p_k = e^{-\frac{\Delta f_k}{T_k}}$ and pick a random number $r_k \in \left(e^{-\eta}, e^{-\frac{1}{\eta}}\right)$.

5: If $p_k \geq r_k$ let $x_{k+1} = z_k$ and go to step 6. Otherwise, let α_k^{BB} be a step size determined by the Armijo line search and $x_{k+1} = x_k - \alpha_k \nabla f_k$.

6: Compute $\alpha_{k+1} = \min\{\alpha_{k+1}^{SABBm}, \alpha_{k+1}^{stab}\}$, where α_{k+1}^{SABBm} is computed from (11) and $\alpha_{k+1}^{\text{stab}} = \frac{\Delta}{\|\nabla f_k\|}$.

7: Let $T_{k+1} = \gamma T_k; k = k+1$ and go to Step 2.

Note that in the SABB*mstab* method, we only replace the step size of the SABB*stab* with the step given in (11). Since SABB*stab* is convergent, SABB*mstab* method is also convergent.

3 Numerical experiments

In this section, we perform numerical experiments to showcase the effectiveness of all variants of the SABB method. In our entire study, we discuss two types of algorithms. The performances of the proposed methods SABBm, SABB*stab*, and SABB*mstab* are compared with the performance of the SABB method described in the research [10]. For all our experiments, we imple-

mented the algorithms using RStudio. The computations were performed on a laptop equipped with an Intel(R) Core(TM) i3 CPU at 3.20 GHz and 4GB of memory. We focus on demonstrating the performance differences between stabilized and nonstabilized variants of SABB. The algorithms SABB and SABBm are terminated when the number of iterations exceeds 3000, or $\|\nabla f_k\| \leq 10^{-6}$. In the stabilized version, we compute two step sizes in each iteration and choose their minimum. The step-size computation becomes a time-consuming task; therefore, we terminate the algorithms SABB$stab$ and SABB$mstab$ when the iterations exceed 10^5, or $\|\nabla f_k\| \leq 10^{-6}$. In all four methods, if the above termination criteria fail, then we claim that the method fails, and we reflect it in Table 2 as "F".

We observe that the selection of α_0, $\alpha_{\min}$, and $\alpha_{\max}$ plays a crucial role in the computation. Similarly, for the SABB$stab$ and SABB$mstab$ methods, along with these three values, the selection of the second initial value x_1 plays a crucial role in the convergence of these methods. For these two methods, we need the value of Δ also, which we compute using the formula $\Delta = \|x_k - x_{k-1}\|$ for $k = 1, 2, \ldots, n$. To identify suitable parameters for our methods, we solved instances of different dimensions available in the CUTEr library and observed that the accuracy in our computation is much better when we set $2^{-30} \leq \alpha_{\min} \leq \alpha_{\max} \leq 2^{20}$, $\alpha_0 = 2^{-10}(\alpha_{\min} + \alpha_{\max}), \gamma = 0.99, T_0 = 1000, c = 10^{-4}$, and $\eta = 20$.

The test functions reflected in Table 1 are taken from the CUTEr library [2, 12]. Table 2 presents the numerical results, where $IP(n)$ denotes the problem serial number with the dimension of the problem. We have taken the dimension from 2 to 500. In the columns under the methods such as SABB, SABBm, SABB$stab$, and SABB$mstab$, three computations are mentioned, such as *feval*, *iter*, and *cput*, which represent the number of function evaluations, the number of iterations, and CPU time (measured in seconds), respectively.

Table 1: Test functions

IP	Function Name	IP	Function Name
1	Extended Freudenstein and Roth Function:	40	Extended BD1 function (Block Diagonal):
2	Extended Trigonometric Function:	41	Extended Maratos Function:
3	Extended Rosenbrock Function:	42	Perturbed quadratic diagonal Function:
4	Generalized Rosenbrock Function:	43	Extended Wood Function:
5	Extended White and Holst function:	44	Quadratic QF1 Function:
6	TRIDIA function (CUTE):	45	Extended quadratic penalty QP1 Function:
7	Extended Beale function:	46	Extended quadratic penalty QP2 Function:
8	Extended Penalty function:	47	Quadratic QF2 Function:
9	ARGLINB function (CUTE):	48	Extended quadratic exponential EP1 function:
10	FLETCHCR function (CUTE):	49	POWER function (CUTE):
11	ARWHEAD function (CUTE):	50	ENGVAL1 function (CUTE):
12	EG2 function (CUTE):	51	EDENSCH function (CUTE):
13	Partial Perturbed Quadratic function (CUTE):	52	CUBE function (CUTE):
14	Almost Perturbed Quadratic function:	53	Extended quadratic exponential EP1 function:
15	NONDIA function (CUTE):	54	Perturbed Tridiagonal Quadratic function:
16	Staircase 1 function:	55	Staircase 2 function:
17	LIARWHD function (CUTE):	56	DQDRTC function (CUTE):
18	Extended Freudenstein and Roth Function:	57	Perturbed Tridiagonal Quadratic function:
19	BDQRTIC function (CUTE):	58	BIGGSB1 Function (CUTE):
20	NONDQUAR function (CUTE):	59	Extended DENSCHNB Function (CUTE):
21	Broyden Tridiagonal function (CUTE):	60	Generalized Quartic Function:
22	ARGLINC function (CUTE):	61	Diagonal 8 Function:
23	BDEXP function (CUTE):	62	Full Hessian FH3 Function:
24	NONSCOMP function (CUTE):	63	SINCOS Function:
25	QUARTC function (CUTE):	64	HIMMELH Function (CUTE):
26	Extended DENSCHNF Function (CUTE):	65	Raydan 1 Function:
27	DIXON3DQ Function (CUTE):	66	Raydan 2 Function:
28	COSINE Function (CUTE):	67	DIXMAANA Function:
29	Diagonal 7 Function:	68	DIXMAANB Function:
30	Diagonal 9 Function:	69	DIXMAANC Function:
31	HIMMELBG Function (CUTE):	70	DIXMAAND Function:
32	Diagonal 1 Function:	71	DIXMAANF Function:
33	Diagonal 2 Function:	72	DIXMAANH Function:
34	Diagonal 3 Function:	73	Extended TET Function (Three Exponential Terms):
35	Generalized Tridiagonal 1 Function:	74	Diagonal 4 Function:
36	Extended Tridiagonal 1 Function:	75	Diagonal 5 Function:
37	Generalized PSC1 Function:	76	Extended Himmelblau Function:
38	Extended PSC1 Function:	77	Generalized White and Holst Function:
39	Full Hessian FH2 Function:		

Table 2: Test results

IP(n)	SABB			SABB*m*			SABB*stab*			SABB*mstab*		
	feval	iter	cput	feval	iter	cput	feval	iter	cput	feval	iter	cput
1(2)	573	142	0.1159	53	12	0.0131	545	136	0.1844	48	12	0.0153
2(2)	873	217	0.3042	97	23	0.0291	681	170	0.4070	72	18	0.0565
3(2)	12005	3000	4.4121	3001	749	0.9078	38297	9574	25.9366	5172	1293	3.4904
4(2)	12005	3000	5.3538	3001	749	1.3600	38297	9574	25.1429	5172	1293	3.1331
5(2)	12005	3000	5.9533	3653	912	1.7136	39965	9991	28.1861	5396	1349	3.7478
6(2)	641	159	0.1028	521	129	0.0807	665	166	0.1515	540	135	0.1224
7(2)	3489	871	0.9035	3229	806	0.7592	400001	100000	5.4008	400000	100000	2.1504
8(2)	1157	288	0.2649	941	234	0.2008	1189	297	0.9345	972	243	0.3556
9(2)	20	4	0.0168	24	5	0.0161	33	8	0.0491	32	8	0.0130
10(10)	1906	503	3.2912	1886	498	3.0315	5	1	0.0181	4	1	0.0150
11(10)	104	25	0.2370	116	28	0.2753	385	96	2.1929	308	77	0.9040
12(10)	12005	3000	10.8701	12005	3000	10.1288	53605	13401	1.6342	43408	10852	1.3426
13(10)	249	61	0.9032	201	49	0.6771	213	53	1.0810	168	42	0.8606
14(10)	529	131	0.5164	429	106	0.3814	441	110	0.6078	356	89	0.5343
15(10)	6789	1696	4.3888	6209	1551	3.9080	51641	12910	2.5466	50620	12655	1.8658
16(10)	2244	560	2.7465	2503	625	3.2653	4813	1203	7.0123	56936	14234	1.7390
17(10)	1593	397	0.7971	1561	389	0.8241	3641	910	2.1399	3568	892	2.0244
18(40)	383	96	1.6555	445	112	2.2936	F	F	F	F	F	F
19(40)	606	152	3.4532	797	200	4.4500	4001	1000	5.0125	1700	425	1.5056
20(40)	12001	3000	55.9570	11999	3000	1.2298	8001	2000	3.1065	8000	2000	3.4526
21(40)	171	42	1.8844	167	41	1.8661	F	F	F	F	F	F
22(40)	1324	332	56.2828	11999	3000	10.4524	1957	489	35.0812	1840	460	23.7643
23(40)	12005	3000	2.2816	12005	3000	2.3259	32001	8000	15.1335	32000	8000	13.5988
24(40)	11972	3000	20.2500	11982	3000	21.5313	8001	2000	2.2935	8000	2000	2.0169
25(40)	1377	343	3.5877	1373	342	3.2536	8001	2000	2.1647	8000	2000	1.7061
26(40)	104	25	0.3547	108	26	0.3713	4001	1000	2.4536	264	66	10.2070
27(40)	1467	368	3.5020	1398	351	3.0541	5	1	0.0238	4	1	0.0369
28(40)	225	55	0.6246	165	40	0.4936	4001	1000	1.8497	4000	1000	1.9239
29(40)	44	10	0.1541	44	10	0.1378	4001	1000	1.5144	1628	407	41.7341
30(40)	11997	3000	29.1681	11982	3000	28.9967	4001	1000	1.2257	4000	1000	1.1827
31(40)	12005	3000	32.1075	12005	3000	32.4967	32001	8000	8.7434	32000	8000	8.7530
32(40)	12003	3000	6.6654	340	84	11.8590	F	F	F	F	F	F
33(40)	224	55	8.7581	236	58	3.8310	4001	1000	3.0450	4000	1000	3.0237
34(40)	12003	3000	2.8101	228	56	2.8166	621	155	30.4833	820	205	38.2410
35(40)	113	27	3.0366	109	26	2.4956	F	F	F	F	F	F
36(40)	1209	301	20.4789	1425	355	23.4677	40001	10000	11.3231	40000	10000	12.7731
37(40)	12003	3000	8.3764	12003	3000	8.2833	40001	10000	25.0045	40000	10000	25.0573
38(40)	57	13	1.4227	57	13	1.4582	1249	312	1.7140	1248	312	1.7120
39(40)	953	240	12.6938	1377	346	18.4661	F	F	F	F	F	F
40(40)	300	74	5.5401	304	75	5.8676	F	F	F	F	F	F
41(40)	64	15	0.9178	64	15	0.9510	F	F	F	F	F	F
42(40)	196	48	2.1041	288	71	2.6848	7201	1800	31.1102	7188	1797	27.9218
43(40)	3456	885	51.5988	6748	1731	1.6669	4001	1000	3.2047	4000	1000	2.1038
44(40)	247	61	1.9867	291	72	1.9010	817	204	8.7129	812	203	8.4563
45(40)	112	27	1.3176	96	23	1.0800	1469	367	32.8439	1464	366	33.5307
46(40)	247	62	3.4999	477	121	6.5672	5	1	0.1461	4	1	0.1123
47(40)	334	83	3.0507	335	83	2.9854	4001	1000	49.9540	4000	1000	49.8391
48(40)	313	77	5.7917	325	80	5.8702	7085	1771	28.6868	7072	1768	27.9670
49(50)	6503	1630	12.5130	7159	1796	11.1532	5	1	0.0179	4	1	0.0170
50(50)	2461	614	7.5193	1125	280	3.3699	797	199	3.6244	780	195	3.6969
51(50)	433	107	1.8298	421	104	1.7742	749	187	4.4861	736	184	4.4655
52(50)	11996	3000	28.3663	11998	3000	28.9950	5	1	0.0229	4	1	0.0181
53(50)	41	9	1.8297	41	9	1.0646	37	9	1.9401	36	9	1.2722
54(50)	436	108	24.9522	412	102	27.8671	1981	495	4.1695	1976	494	1.6517
55(50)	2346	588	6.9273	2375	595	7.4707	5	1	1.0590	4	1	0.7714
56(50)	158	39	10.1396	169	42	6.9630	9013	2253	2.7833	9012	2253	3.6513
57(50)	327	81	11.9068	327	81	7.0950	1341	335	1.1654	1340	335	1.2965
58(50)	1286	322	40.9906	1223	306	25.9469	5	1	0.0379	4	1	0.0302
59(100)	657	163	5.8164	533	132	4.3743	657	164	8.2231	532	133	6.4424
60(100)	561	139	7.0155	453	112	4.9969	397	99	7.3207	320	80	6.3334

Table 2 Continued...

61(200)	357	88	17.6759	289	71	10.9239	405	101	32.2711	328	82	22.8033
62(200)	25	5	1.5282	80	19	4.6503	21	5	1.7963	48	12	3.9969
63(200)	569	141	33.5531	401	99	22.7348	1413	353	2.1739	1156	289	1.6964
64(200)	621	154	24.6155	505	125	20.2527	645	161	39.3083	524	131	31.9428
65(200)	12005	3000	5.3382	12005	3000	5.5383	2533	633	1.6214	2100	525	1.1722
66(200)	1449	361	39.0737	1057	263	28.5491	893	223	35.2179	732	183	28.7968
67(300)	45	10	16.0129	45	10	16.3452	261	65	19.8307	260	65	21.2720
68(300)	37	8	12.1318	37	8	12.1640	493	123	43.9275	492	123	1.1385
69(300)	37	8	11.2262	37	8	11.6436	929	232	7.1460	928	232	7.2529
70(300)	72	17	23.1260	72	17	23.8315	1881	470	15.0397	1876	469	15.0572
71(300)	968	241	35.4163	981	244	35.2606	4001	1000	41.7712	4000	1000	39.0339
72(300)	589	147	24.1729	683	171	27.5352	F	F	F	F	F	F
73(500)	109	26	1.0931	121	29	49.6553	77	19	46.0448	76	19	35.9698
74(500)	33	7	4.4168	56	13	4.8235	5	1	0.7420	4	1	0.5555
75(500)	40	9	13.7751	40	9	8.7212	5	1	1.0040	4	1	0.6785
76(500)	136	33	26.7670	136	33	17.6005	2893	723	12.9185	2884	721	9.4026
77(500)	61	14	1.0407	61	14	52.2529	4525	1131	1.0296	4516	1129	24.5784

In Table 3, we present the number of problems for which the method achieves the least iter, the least cput, and the least feval, respectively. The observations from Table 3 lead to the conclusion that the SABBm method exhibits superior performance compared to SABB, and SABB$mstab$ outperforms SABB$stab$ in the stabilized version. We also observe that for problems $10, 14, 27, 46, 49, 52, 55, 58, 74$, and 75, the stabilized versions SABB$mstab$ and SABB$stab$ exhibit superior performance in terms of iterations, function evaluations, and time.

Table 3: Least table

Metric	**SABB**	**SABB**m	**SABB**$stab$	**SABB**$mstab$
feval	31	**33**	1	23
iter	34	**36**	14	19
cput	17	**30**	8	22

We employed the performance profile of Dolan and Moré [9] to compare the performance of the proposed methods. We construct the performance profile graphs for three key metrics: function evaluations (feval), number of iterations (iter), and CPU time (cput). Figure 1 illustrates the function evaluations performance profile. Figure 2 depicts the performance profile for number of iterations, and Figure 3 shows the performance profile of CPU time. Across all three metrics (Figures 1–3), SABBm is almost the top curve, indicating its superior efficiency compared to the other methods.

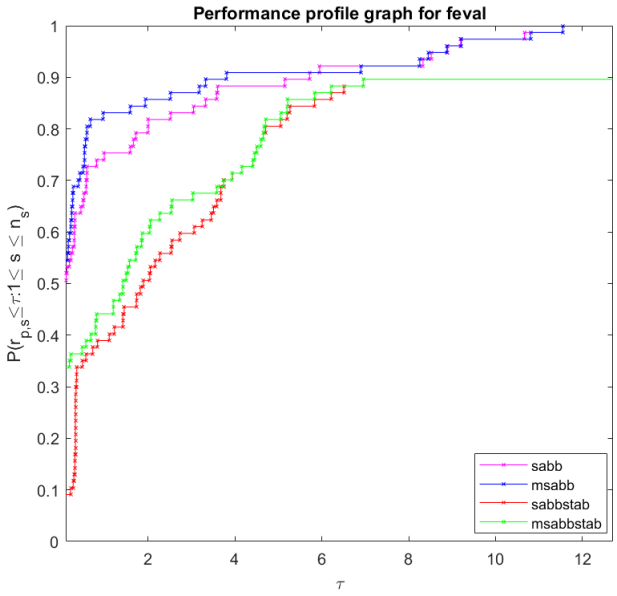

Figure 1: Performance profile for functional evaluations.

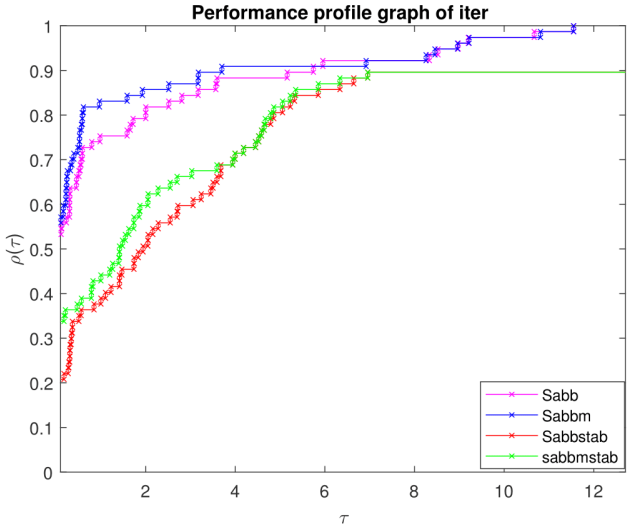

Figure 2: Performance profile for number of iterations.

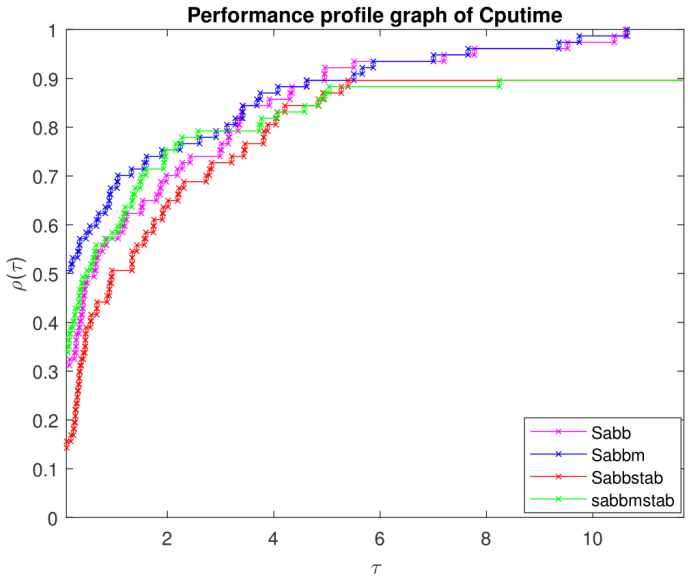

Figure 3: Performance profile for CPU time.

4 Conclusion

This study proposes three novel variants of the SABB algorithm for solving unconstrained optimization problems. These variants hybridize the BBStab and SABB approaches. The performance of the proposed methods is evaluated on a set of 77 benchmark problems from the CUTEr library (details in Table 2). The results reveal that SABBm emerges as the most efficient algorithm in terms of function evaluations, iterations, and CPU time. However, SABB$stab$ and SABB$mstab$ outperform SABBm in a small subset of problems regarding the number of iterations and computational time.

Compliance with Ethical Standards

- This article does not contain any studies involving animals performed by any of the authors.

- This article does not contain any studies involving human participants performed by any of the authors.

Acknowledgments

This research work is fully supported by the National Board for Higher Mathematics (NBHM), Department of Atomic Energy, Govt. of India (Grant no. 02011/22/2021 NBHM (R.P.)/R&D II/900 Date 06/08/2021).

Conflict of interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

[1] Akaike, H. *On a successive transformation of probability distribution and its application to the analysis of the optimum gradient method*, Annals of the Institute of Statistical Mathematics 11(1) (1959), 1–16.

[2] Andrei, N. *An unconstrained optimization test functions collection*, Adv. Model. Optim. 10(1) (2008), 147–161.

[3] Barzilai, J. and Borwein, J.M. *Two-point step size gradient methods*, IMA J. Numer. Anal. 8(1) (1988), 141–148.

[4] Birgin, E.G., Martínez, J.M. and Raydan, M. *Nonmonotone spectral projected gradient methods on convex sets*, SIAM J. Optim. 10(4) (2000), 1196–1211.

[5] Burdakov, Y., Dai, O. and Huang, N. *Stabilized Barzilai–Borwein method*, J. Comput. Math. 37(6) (2019), 916–936.

[6] Cauchy, A. *Méthode générale pour la résolution des systemes d'équations simultanées*, Comp. Rend. Sci. Paris 25 (1847), 536–538.

[7] Dai, Y.-H. and Liao, L.-Z. *R-linear convergence of the barzilai and borwein gradient method*, IMA J. Numer. Anal. 22(1) (2002), 1–10.

[8] Dai, Y.-H. and Zhang, H. *Adaptive two-point stepsize gradient algorithm*, Numer. Algorithms 27 (2001), 377–385.

[9] Dolan, E.D. and Moré, J.J. *Benchmarking optimization software with performance profiles*, Math. Program. 91(2) (2002), 201–213.

[10] Dong, W.-L., Li, X. and Peng, Z. *A simulated annealing-based Barzilai–Borwein gradient method for unconstrained optimization problems*, Asia-Pac. J. Oper. Res. 36(04) (2019), 1950017.

[11] Fletcher, R. *Low storage methods for unconstrained optimization*, Dundee Department of Mathematics and Computer Science, University of Dundee, 1988.

[12] Gould, N.I.M., Orban, D. and Toint, P.L. *Cutest: a constrained and unconstrained testing environment with safe threads for mathematical optimization*, Comput. Optim. Appl. 60(3) (2015), 545–557.

[13] Grippo, L., Lampariello, F. and Lucidi, S. *A nonmonotone line search technique for newton's method*, SIAM J. Numer. Anal. 23(4) (1986) 707–716.

[14] Han, J. and Liu, G. *Global convergence analysis of a new nonmonotone BFGS algorithm on convex objective functions*, Comput. Optim. Appl. 7 (1997), 277–289.

[15] Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P. *Optimization by simulated annealing*, Science, 220(4598) (1983), 671–680.

[16] Liu, G.H. and Peng, J.M. *The convergence properties of a nonmonotonic algorithm*, J. Comput. Math. 1 (1992), 65–71.

[17] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. and Teller, E. *Equation of state calculations by fast computing machines*, J. Comput. Math. 21(6) (1953), 1087–1092.

[18] Mu, X. and Liu, W. *An augmented lagrangian method for binary quadratic programming based on a class of continuous functions*, Optim. Lett. 10(3) (2016), 485–497.

[19] Nocedal, J. and Wright, S.J. *Numerical optimization*, Springer, 1999.

[20] Raydan, M. *On the barzilai and borwein choice of steplength for the gradient method*, IMA J. Numer. Anal. 13(3) (1993), 321–326.

[21] Raydan, M. *The barzilai and borwein gradient method for the large scale unconstrained minimization problem*, SIAM J. Optim. 7(1) (1997), 26–33.

[22] Toint, P.L. *An assessment of nonmonotone linesearch techniques for unconstrained optimization*, SIAM J. Sci. Comput. 17(3) (1996), 725–739.

[23] Wang, C., Liu, Q. and Yang, X. *Convergence properties of nonmonotone spectral projected gradient methods*, J. Comput. Appl. Math. 182(1) (2005), 51–66.

[24] Zhang, H. and Hager, W.W. *A nonmonotone line search technique and its application to unconstrained optimization*, SIAM J. Optim. 14(4) (2004), 1043–1056.

[25] Zhensheng, Yu. *Solving bound constrained optimization via a new nonmonotone spectral projected gradient method*, Appl. Numer. Math. 58(9) (2008), 1340–1348.

[26] Zhou, J.L. and Tits, A.L. *Nonmonotone line search for minimax problems*, J. Optim. Theory Appl. 76(3) (1993), 455–476.