Advances in Nonlinear Variational Inequalities
ISSN: 1092-910X
Vol 28 No. 2 (2025)

Techniques for Solving Bound-Constrained Convex Optimization Using Spectral Projected Conjugate Gradient Based on Barzilai-Borwein Step Length

Sudaba A. Mohammed [1]*, Ivan S. Latif [2], Norjan H. Jumaa [3]
[1] Department of Mathematics, College of Education for Pure Science, University of Kirkuk, Kirkuk, Iraq.
sudabamohammed@gmail.com
[2] Department of Mathematics, College of Education, Salahaddin University, Kurdistan region, Erbil, Iraq.
[3] Department of Mathematics, College of Education for Pure Science, University of Kirkuk, Kirkuk, Iraq.
*Corresponding author. https://orcid.org/0000-0002-9344-2638

Article History:

Received: *31-07-2024*

Revised: *19-09-2024*

Accepted: *28-09-2024*

Abstract

This work aims to create an effective spectral conjugate gradient for non-linear optimization problems and to project the solution into a bounded convex set for largescale optimization problems. We do this by combining the classical spectral conjugated gradient direction with the projected Barzilai and Borwein step lengths. The efficacy and convergence of the new method are illustrated through a series of problems.

Keywords: Conjugate gradient; Spectral; Bound-onstrained; Convex problems; Barzilai-Borwein

1:Introduction

Large-scale bounded constrained optimization problems arising from infinite dimensional space that are discretized employing the method of finite elements are the focus of this work [14]. The study's methodology is based on projected Barzilai and Borwein step lengths in conjunction with the spectral conjugate gradient method [7]. We use the proposed method to address the bounded constrained optimization problems (the same examples studied in [14]), but without using any sophisticated line search. These methods were initially used to address unconstrained nonlinear optimization problems, so it was necessary to ascertain how they affected bounded convex optimization problems. The primary benefit of the technique under study is that, for large-scale convex optimization problems, only gradient directions are employed; in contrast, the non-monotone method guarantees adequate convergence.

Several recent studies [24,9,15,11,21] used spectral conjugate gradient approaches to solve problems related to unconstrained optimization. However, spectral conjugate gradient methods were employed by other researchers [2,23] to solve systems of nonlinear equations. Using projected gradient techniques, constrained optimization problems can be

successfully tackled, as demonstrated by the groundbreaking paper [4, 7]. The MMA approach for topology optimization uses the preconditioned spectral projected gradient [22]. Unconstrained optimization problems, such as [1, 19], are also resolved using the spectral second-order methods. We can consult [10] for sufficient details regarding convergence rates for unconstrained optimization. In contrast, [18] establishes the global convergence and analysis of the strictly convex quadratic case to any number of variables using the BB method. Many related types of research, such as [8,12], are conducted after that to improve and extend Barzilai and Borwien step length. From the numerical calculations, many researchers concluded that the conjugate gradient and spectral ideas could be combined to get the most precise methods for solving unconstrained optimization problems [3, 17, 18].

Furthermore, applying Barzilai-Berwein and spectral conjugate gradient methods to bounded nonlinear convex optimization problems is the main objective of this work. The following is the format of the paper: In Section 2, the basic concept of the proposed problem, the algorithms, and the spectral conjugate gradient method have been introduced. Numerical experiments are covered in Section 3; we pay particular attention to the dependence of the convergence rate, the maximum number of iterations and the CPU time needed. Finally, Section 4 introduces the conclusion.

2: Spectral Project Conjugate Gradient Method For Convex Optimization.

2.1 The Problem

Generally, the main goal of the research is to solve the nonlinear bound-constrained problem [14].

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

Subject to

$$\varphi_i \le x_i \le \psi_i,\ i = 1, \dots, n$$

Given a nonlinear function f that is continuously differentiable and convex, $\varphi_i < \psi_i$ for all i. In order to ensure that a solution exists, It is assumed that either the function f is coercive or the feasible set $\mathscr{F}$ is bounded.

$$\mathscr{F} = \{x \in \mathbb{R}^n \mid \varphi_i \le x_i \le \psi_i, i = 1, \dots, n\}$$

We consider the problem to be one of large-scale optimization, and the suggested problems are discretized using the finite element method; eq. (1) is a bound-box quadratic problem, let us assume.

$$\min_{x\in\mathbb{R}^n} f(x) = \frac{1}{2}x^2 Ax - f^T x \tag{2}$$

$$\varphi_i \le x_i \le \psi_i,$$

The symmetric and positive define matrix $A \in R^{n\times n}$ has a large size n, and $f \in R^n$. In order to solve equation (2), the set of feasible solutions is provided as

$$\Omega = \{x \in R^n \mid \varphi_i \le x_i \le \psi_i\} \tag{3}$$

In this work, we take into account projected Barzilai-Borwien step lengths provided by [7] in conjunction with the spectral conjugate gradient line search direction from [3]. The method then is called the spectral projected conjugate gradient method.

2.2 The Spectral Projected Conjugate Gradient Method

Finding a search direction $d_k \in R^n$ is necessary at each iteration in order to compute a new point x_{i+1} for eq. (1).

$$x_{i+1} = x_i - \alpha_i d_i \tag{4}$$

In this case, d_{i+1} is the new direction given by

$$d_{i+1} = \theta_i g_{i+1} - \beta_i s_i \tag{5}$$

where $x_o \in R^n$ is arbitrary, g_i represents $\nabla f(x_i)$, and

$$d_o = \theta_o g_o$$

such that the two successive approximations, x_i and x_{i+1}, are defined by

$$s_i = x_{i+1} - x_i = \alpha_i d_i$$

and

$$y_i = g_{i+1} - g_i$$

When eq. (2) is used without constraints, the exact minimize x_* satisfies

$$x_* = x_{i+1} + d_k$$

where

$$Ad_k = g_{i+1}, \tag{6}$$

By multiplying the latter by s_i^T, we obtain

$$s_i^T A d_k = s_i^T g_{i+1},$$

hence

$$y_i^T d_k = s_i^T g_{i+1}.$$

Consequently, the hyper-plane

$$\mathscr{H}_i = \{d \in R^n \mid y_i^T d_k = s_i^T g_{i+1}\}$$

retain the ideal direction d_*, which results in

$$x_* = x_{i+1} + d_k$$

It is evident that $\mathscr{H}$ contains the null direction $d = 0$ only if $s_i^T g_{i+1} = 0$. According to Perry (1978), the previous discussion significantly influences the search direction d_{i+1} to belong to the hyper-plane $\mathscr{H}$.

Therefore, by (5)

$$\beta_i = \frac{(\theta y_i - s_i)^T g_{i+1}}{s_i^T y_i} \tag{7}$$

[3] suggests that if $s_i^T g_{i+1} = 0, i = 0,1,2, \dots, k$ then this was first introduced by [17] with $\theta = 1$.

$$\beta_i = \frac{\theta_i y_i^T g_{i+1}}{\alpha_i \theta_{i-1} g_i^T g_i} \tag{8}$$

This study compares the use of β_i (7) with the classical choice $\theta = 1$ first, and then with the spectral gradient choice.

$$\theta = \frac{s_i^T s_i}{s_i^T y_i} \tag{9}$$

The definition of the directed search d is precisely as it appears in Step 3 of Algorithm SCG [3]. Moreover, the Algorithm has to be restarted to use $d = \theta g_{i+1}$ instead of $\theta_i g_i - \beta_i s_i$ in the case that the angle between d and g_{i+1} is not sufficiently acute. Here we assume that, instead of using a sophisticated line search, the coefficient θ_i is always positive and welldefined due to the conditions applied in the process of selecting the Barzilai-Berwein step lengths. We employ Barzilai-Borwein step lengths and project the solution into the convex set [6] . Additionally, we employ the step lengths in two ways: first, we use them in the alternate Barzilai-Borwein ABB method as described in [7].

$$\alpha^{ABB} = \begin{cases} 1, & k = 1; \\ \frac{s_{k-1}^T s_{k-1}^T}{s_{k-1}^T y_{k-1}}, & \text{for odd } k; \\ \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}, & \text{for even } k. \end{cases} \tag{10}$$

The second application involves converting the concept of alternation into a comparison using the sign of $s_{k-1}^T y_{k-1}$ in the projected Barzilai-Borwein [4]; PBB is used to indicate this.

$$\alpha^{PBB} = \begin{cases} 1, & k = 1; \\ \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}, & \text{otherwise}, \end{cases} \tag{11}$$

Additionally, the Barzilai-Borwien step lengths that were presented in [20] are a good fit for the spectral conjugate gradient line search that is being studied. Nevertheless, the modified Barzilai-Berwein step sizes [20] may result in a slower choice due to the costly determination of stiffness matrix A and A^2. Setting upper and lower bounds for the step sizes is one of the minor adjustments that can be made to make use of that. The following algorithm is determined and structured as a base algorithm to illustrate the behavior of the projected Barzilai-Berwein method for solving bound-constrained optimization problems without using spectral terms.

2.3 Algorithm 1: Barzilai-Borwein Projected Conjugated Gradient Algorithm without Spectral Term

Suppose that $x_0 \in \mathbb{R}^n, d_0 = g_0, g_0 = \nabla f(x_0), \alpha_0 = 1, \epsilon > 0$ and set $i = 0$.
Step 1: If $\langle g_i \cdot (x_i - \varphi_i) \cdot (\psi_i - x_i) \rangle \le \epsilon$, stop.
Step 2: Compute α_i as following

$$\alpha^a := \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}$$

$$\alpha^b := \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}$$

To determine the ABB step length, use 10 to calculate α_i. To determine the PBB step length in the other scenario, use the following commands.

If $s_{i-1}^T y_{i-1} <= 0$, then $\alpha_i = \max\{\alpha^a, \alpha^b\}$, else
$\alpha_i = \min\{\alpha^a, \alpha^b\}$
If $\alpha_i <= 10^{-10}$ or $\alpha_i >= 10^{10}$, let $\alpha = 1$

Step 4: compute β_i

$$\beta_i = \frac{\|y_i\|}{d_{i-1}^T A g_{i-1}}.$$

$$d_i = \begin{cases} g_i, & \text{Steepest descent}; \\ g_i - \beta_i s_i, & \text{Conjugate gradient}, \end{cases} \tag{12}$$

$x_{i+1} = x_i - \alpha_i d_i$ and project it into the feasible set using

$$x_{i+1} = \max\{\varphi_i, x_{i+1}\}; x_{i+1} = \min\{\psi_i, x_{i+1}\},$$

here, the symbols φ_i and ψ_i, respectively, clearly indicate the lower and upper bounds for the Problems.
set $i = i + 1$ and go to step 2 .

Algorithm 1 is only effective in solving bound-quadratic problems up to refinement 8 . That is limited to the conjugate gradient method and only functions with projected Barzilai-Berwein step length PBB. The algorithm also fails for different applications of step length or search direction, as shown in Tables [1].
Consequently, Algorithm 1 is modified by adding the spectral term θ and using some conditions to restrict the step length α, which is changed by including β_i and θ_i from [3] in order to solve more general nonlinear optimization problems.

2.4 Algorithm 2: Barzilai-Borwein Projected Conjugated Gradient Algorithm Using the Spectral Term $\boldsymbol{\theta}$

Assume that $x_0 \in \mathbb{R}^n, d_0 = g_0$ and $\epsilon > 0$.
Step 1: If $\langle g_i \cdot (x_i - \varphi_i) \cdot (\psi_i - x_i) \rangle \leq \epsilon$, stop.
Step 2: compute α_i as same as the step 2 in Algorithm 1.
Step 3: Select the spectral term θ_i for $i = 1,2,3, \ldots$.

$$\theta_i = \begin{cases} 1, & \text{or} \\ \frac{s_{i-1}^T s_{i-1}}{s_{i-1}^T y_{i-1}}, \end{cases} \tag{13}$$

Calculate β_i

$$\beta_i = \frac{(\theta_i y_i - s_i)^T g_i}{s_i^T y_i} \tag{14}$$

Step 4: Determine the search direction d_i, [3]

$$d_i = \begin{cases} \theta_i g_i, & \text{if } d_i^T g_i \geq -0.001 \|d_i\| \| g_i\| \\ \theta_i g_i - \beta_i s_i, & \text{otherwise} \end{cases} \tag{15}$$

Step 5: Calculate a new point x_{i+1}
$x_{i+1} = (x_i - \alpha_i d_i)$ and project it into the feasible set using

$$x_{i+1} = \max\{\varphi_i, x_{i+1}\}; x_{i+1} = \min\{\psi_i, x_{i+1}\},$$

where the symbols φ_i and ψ_i, respectively, represent the problems' lower and upper bounds.

Go to step 2 after setting $i = i + 1$.

The primary contribution to the method's convergence is made by the algorithm, which computes α using projected Barzilai-Berwein step lengths.

The idea of adjusting the Barzilai-Borwein step length selection according to the sign of the projected directional derivative of f in the direction, $-g(x)$, of the steepest descent is simply substituted for step two in the consequent Algorithm 3. At trial point $x - \alpha g(x)$, [14]. Recall that the following defines the set of active indices for a given feasible point x.

$$\mathcal{A}(x) = \{i \mid x_i = \varphi_i \text{ or } x_i = \psi_i\}.$$

Additionally, for this x, a component-wise defined operator $[\cdot]_{\mathcal{A}(x)}: \mathbb{R}^n \to \mathbb{R}^n$ is introduced by

$$\left([h]_{\mathcal{A}(x)}\right)_i = \begin{cases} 0 \text{ if } i \in \mathcal{A}(x) \\ h_i \text{ otherwise.} \end{cases}$$

Let $[\cdot]_\Omega$ finally stand for the feasible set projection, which in this case is trivial.

2.5 Algorithm 3: Spectral Conjugate Gradient Algorithm Using Projected Barzilai-Borwein Step-Length Based on Gradient Directional Derivative

step 2: Calculate $x^+$and γ as follow

$$x^+ = [x - \alpha g(x)]_\Omega, \tag{16}$$

$$\gamma = -(g(x))^T [g(x^+)]_{\mathcal{H}(x^+)} \tag{17}$$

$\alpha_1 = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}, \alpha_2 = \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}$

$$\alpha = \begin{cases} \max\{\alpha_1, \alpha_2\}, & \text{if } \gamma < 0; \\ \min\{\alpha_1, \alpha_2\}, & \text{if } \gamma > 0. \end{cases}$$

3 Convergence analysis

Our experiments show that using the combination of Barzilai-Borwein step lengths and spectral conjugate gradient line search has a significant impact on solving bounded nonlinear optimization problems on convex sets. Furthermore, the results of using Algorithm 3 and the PBB method are nearly identical. This suggests a similarity between the sign of the projected directional derivative of f in the steepest descent direction $-g(x)$ at the trial point $x - \alpha g(x)$, [14], and the sign of $s_{k-1}^T y_{k-1}$ in the projected Barzilai-Borwein [4]. We could therefore forego introducing the numerical experiments from the Algorithm 3. As a result, we present the next two lemmas from the literature. Ω 's convexity leads to the simple convergence properties of the algorithms that are presented. Additionally, allow us to state the following literature-based lemma.

Lemma 3.1 For all $x \in \Omega, \alpha \in (0, \alpha_{\max})$,

1. $\langle g(x), x^+ - x\rangle \leq -\frac{1}{\alpha} \|x^+ - x\|_2^2 \leq -\frac{1}{\alpha_{\max}} \|x^+ - x\|_2^2$,
2. The vector $(x^+ - x_{\text{new}})$ vanishes if and only if x_{new} is a constrained stationary point.
3. $f(x_{\text{new}}) < f(x)$ and $\gamma^+ < -(\nabla f(x))^T \nabla f(x_{\text{new}}) < 0$.

Proof. The proof of the first two theorems is given in Lemma 2.1 in [4]. While the third theorem's proof is outlined in [14].

4 Numerical experiments

We use problems from the literature [13,5,14] to report the results of our numerical experiments in this section. We will investigate the effects of using spectral conjugate gradient line search on projected Barzilai-Borwin methods, starting with a quadratic problem.

We will provide, for each problem, the asymptotic rate of convergence for a range of numbers of refinement meshes (3,5,7,8, and 9 levels of discretization). The same table includes the number of iterations, function and gradient calls, and CPU complexity time on the discretized level. We will also present an asymptotic rate of convergence and comparison (in the function/gradient calls) with projected gradient, based on the standard projected gradient method with backtracking Armijo line search [14]. To avoid duplication, the final Table 5 presents the outcomes of all problems using the GP-Armijo, standard projected gradient method with backtracking Armijo line search [14].

In the infinite-dimensional setting, each problem is defined on the $\Omega = (0,1)^2$ square. Next, we discretize square finite elements using bilinear basis functions on regular meshes. The first coarsest mesh has four elements (refinement level 0). $j = 8$ uniform refinement steps are used to obtain nine embedded finite element meshes. The finest mesh has 262144 finite elements and 261121 interior nodes. Consequently, at refinement level $k, k = 0,1,2, \ldots$, we have 4^{k+1} finite elements and $\big(2^{k+1} -$

$1)^2$ interior nodes. Under certain conditions, we restrict the steplength using the same step lengths as studied in the projected Barzilai and Borwein [7]. These are combined with spectral conjugate gradient line search [3]. Additionally, we set the maximum number of iterations to 200 and employ the following stopping criteria in all of our tests:

The procedure was terminated at $\|x - x^*\| \leq 10^{-5}$. The L-BFGS-B code [16] was utilized to determine the precise solution x^*. The asymptotic rate of convergence was calculated based on these final iterations.

Two approaches are taken to examine the initial value for the step length: using $1/\|g_1\|_\infty$ [4] and using 1. Regardless of whether it was feasible or not, the starting point for every experiment was a zero vector.
Every algorithm was implemented using MATLAB. Stephen Becker [1] created the L-

BFGS-B code interface. For all experiments, we used a laptop with an Intel Core i7-3570 CPU M 620 at 2.67 GHz and 4GB RAM. On 64-bit Windows 10, MATLAB version 8.0.0 (2012b) was running.

4.1 Obstacle problem in quadratic form

We first begin our numerical experiments with the "Spiral problem" from [13]. This is a quadratic optimization problem derived from the Laplace equation $\Omega \subset \mathbb{R}^2$:

$$\min_{u \in H_0^1(\Omega)} \mathcal{J}(u) := \frac{1}{2}\int_\Omega \| \nabla u \|^2 \, dx - \int_\Omega Fudx$$

Subject to

$$\varphi \leq u \leq \psi, \text{ a.e. in } \Omega,$$

where $F \in L^2(\Omega)$. The spiral obstacle, as suggested in [13, \$7.1.1], will be employed>

$$\varphi(x(r,\phi)) = \sin(2\pi/r + \pi/2 - \phi) + \frac{r(r+1)}{r-2} - 3r + 3.6, \; r \neq 0,$$

and polar coordinates $x(r,\phi) = re^{i\phi}$ with $\varphi(0) = 3.6$. The upper bound function, ψ, is set to infinity, and the function on the right, F, to zero. The solution for 9 refinement discretization is shown in Figure 1, which can be obtained with all recommended convergent methods from Table 2. This is achieved using Algorithm 2.

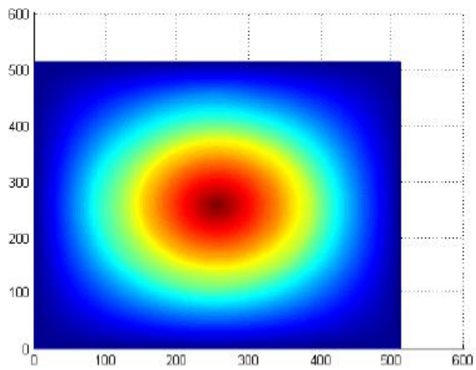

Figure 1: Problem 4.1, 9th refinement level, SCG-ABB solution. $\alpha = 1, \theta = 1$.

Table 2 shows the numerical results by the ways involved in Algorithm 2, Problem 4.1. It displays the total number of evaluations of gradients and objective functions on the discretized level, along with the convergence's asymptotic rate, which is an average over the last three to five iterations. Figures 2 and 3 show the convergence rate vs. number of iterations for the final refinement grids (9th) for all different ways studied.

The results from Table 2 indicate that the alternating Barzilai-Berwein ABB overcomes projected Barzilai-Berwein PBB even though both methods ABB and PBB with the choice $\theta = 1$ converge rapidly with a good rate and low function evaluations. These include both uses of initial step lengths α_1. However, Table 1 presents the numerical results from algorithm 1 which can fail by excluding the spectral constant θ in its procedures unlike Algorithms 2 and 3 .

As for Figures 2 and 3, Figures 2 confirms the nonmonotonic performance for ABB especially when $\theta = \frac{s_k^T s_k}{s_k y_k}$ but that can be reduced if it combined with initial step length $\alpha_1 = 1/\|g_1\|_\infty$. Furthermore, it affects decreasing the number of iteration and function

evaluations as well. On the other hand, Figure 3 presents the more monotonic behavior for PBB with all different choices in terms of initial step lengths and the spectral constants with little differences in their results.

Table 1: Problem 4.1, Algorithm 1 computes the complexity time "CPU" for refinement levels 3-7, the number of descretized level function evaluations "feval", the number of iterations "it", and the asymptotic rate of convergence "rate". ABB step length with projected steepest descent is denoted by ABB − SD, and conjugate gradient line searches is denoted by ABB-CG. PBB-SD and PBB-CG indicate the two PBB-based approaches, respectively.

Algorithm 2	ABB-SD				ABB-CG			
Levels	rate	Feval	it	CPU	rate	feval	it	CPU
4	-	-	-	-	0.04	44	9	0.005 s
5	-	-	-	-	0.13	90	13	0.01 s
7	-	-	-	-	-	-	-	-
8	-	-	-	-	-	-	-	-
9	-	-	-	-	-	-	-	-
Algorithm 2	PBB-SD				PBB-CG			
Levels	rate	feval	it	CPU	rate	feval	it	CPU
4	-	-	-	-	0.07	54	10	0.01 s
5	-	-	-	-	0.13	135	16	0.02 s
7	-	-	-	-	0.35	434	29	0.62 s
8	-	-	-	-	0.43	989	44	8.37 s
9	-	-	-	-	-	-	-	-

Advances in Nonlinear Variational Inequalities
ISSN: 1092-910X
Vol 28 No. 2 (2025)

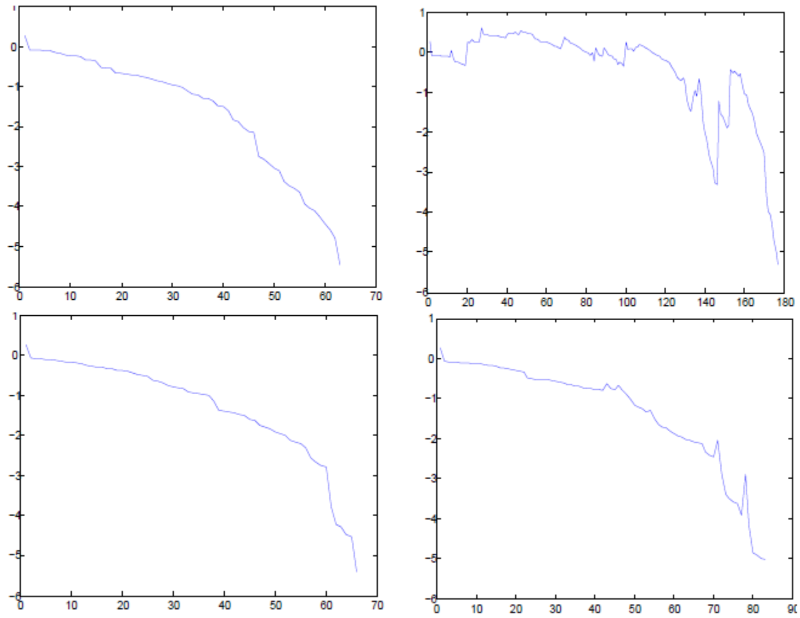

Figure 2: Problem 4.1, 9th refinement level, from top, ABB error (left) with $\theta = 1, \alpha_1 = 1$ and ABB error (right), $\theta = \frac{s_k^T s_k}{s_k y_k}, \alpha_1 = 1$. From down, ABB error (left) with $\theta = 1, \alpha_1 = \frac{1}{\|g_1\|_\infty}$ and ABB error (right), $\theta = \frac{s_k^T s_k}{s_k^T y_k}, \alpha_1 = \frac{1}{\|g_1\|_\infty}$.

Table 2: Problem 4.1, Algorithm 2, convergence's asymptotic rate and number of evaluations of top-level functions for 4-9 discretized levels. ABB denotes alternating BarzilaiBerwein step length alongside spectral conjugate gradient direction with $\theta = 1, \theta = \frac{s_k^T s_k}{s_k^T y_k}$ and the tolerance criteria is less than 10^{-5}.

Algorithm 2, $\alpha_1 = 1$	ABB, $\theta = 1$				ABB, $\theta = \frac{s_k^T s_k}{s_k^T y_k}$			
Levels	rate	feval	it	CPU	rate	feval	it	CPU
4	0.06	35	8	0.02 s	0.097	77	12	0.02s
5	0.13	65	11	0.02 s	0.096	152	17	0.04 s
7	0.15	275	23	0.67 s	0.36	1484	54	3.91 s
8	0.45	702	37	9.69 s	0.53	11780	152	190.94
9	0.46	2015	63	91.34 s	0.48	15752	176	717.38
Algorithm 2, $\alpha_1 = \frac{1}{\|\|_1\|_\infty}$	ABB, $\theta = 1$				ABB, $\theta = \frac{s_k^T s_k}{s_k^T y_k}$			
Levels	rate	feval	it	CPU	rate	feval	it	CPU
4	0.04	35	8	0.02 s	0.20	65	11	0.03 s

5	0.13	65	11	0.02 s	0.05	152	17	0.04 s
7	0.31	377	27	1.03 s	0.17	1127	47	2.84 s
8	0.39	902	42	12.18 s	0.35	1769	59	25.96
9	0.43	2210	67	98.24 s	0.88	3485	83	165.21 s
Algorithm 2, $\alpha_1 = 1$	PBB, $\theta = 1$				PBB, $\theta = \frac{s_k^T x_1}{s_k^T x_k}$			
Levels	rate	feval	it	CPU	rate	feval	it	CPU
4	0.07	44	9	0.02 s	0.08	77	12	0.02 s
5	0.16	104	14	0.04 s	0.08	|135	16	0.02 s
7	0.50	702	37	1.64 s	0.42	665	36	2.25 =
8	0.68	2144	65	25.85 s	0.70	3002	77	41.93 s
9	0.75	6327	112	275.86 s	0.71	11780	152	598.72
Algorithm 2, $\alpha_1 = \frac{1}{\|g_1\|_\infty}$	PBB, $\theta = 1$				PBB, $\theta = \frac{s_k^2 s_k}{s_k^2 k_k}$			
Levels	rate	feval	it	CPU	rate	feval	it	CPU
4	0.09	44	9	0.01 s	0.16	54	10	0.02s
5	0.19	104	14	0.03 s	0.17	152	17	0.04 s
7	0.50	779	39	1.73 s	0.33	945	43	2.22 s
8	0.70	2144	65	24.57 s	0.57	3485	83	42.22 s
9	0.78	6785	116	277.75 s	0.79	13529	164	567.28 s

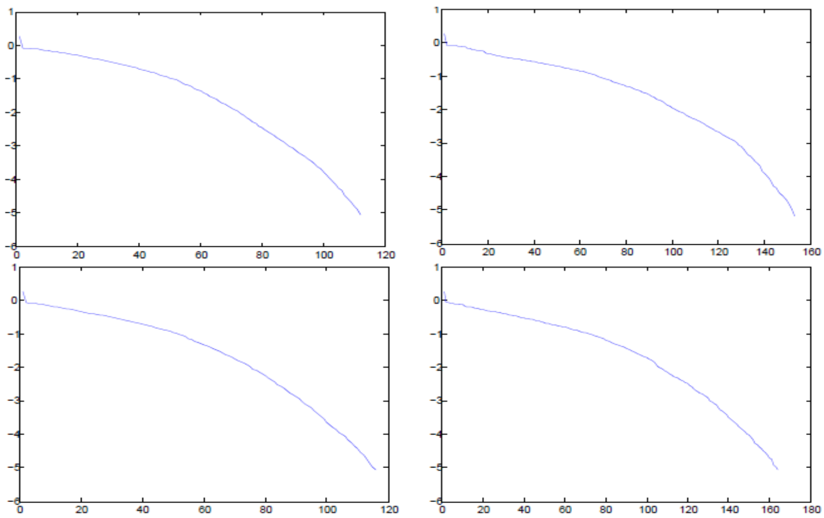

Figure 3: Problem 4.1, 9th refinement level, from the top, PBB error (left) with $\theta = 1, \alpha_1 = 1$ and PBB error (right), $\theta = \frac{s_k^T s_k}{s_k^T y_k}, \alpha_1 = 1$. From down, ABB error (left) with $\theta = 1, \alpha_1 = \frac{1}{\|g_1\|_\infty}$ and PBB error (right), $\theta = \frac{s_k^T s_k}{s_k^T y_k}, \alpha_1 = \frac{1}{\|g_1\|_\infty}$.

4.2 Bounded non-quadratic problem

Take into account the optimization problem in $\Omega \subset \mathbb{R}^2$ that follows:

$$\min_{u \in H_0^1(\Omega)} \mathcal{G}(u) := \frac{1}{2}\int_\Omega \| \nabla u \|^2 - (ue^u - e^u)dx - \int_\Omega Fudx$$

Subject to

$$\varphi \le u \le \psi, \text{ a.e. in } \Omega,$$

here

$$\varphi(x_1, x_2) = -8(x_1 - 7/16)^2 - 8(x_2 - 7/16)^2 + 0.2,\ \psi = 0.5$$

and

$$F(x_1, x_2) = \left(9\pi^2 + e^{(x_1^2 - x_1^3)\sin(3\pi x_2)}(x_1^2 - x_1^3) + 6x_1 - 2\right)\sin(3\pi x_1).$$

A nonlinear PDE was examined in [5, p.105] for the unconstrained version of the problem. Using convergent Algorithm 2, the solution to the proposed constrained problem on the 9 th refinement level is displayed in Figure 4.

Similar to the previous discussion about Table 2, here we have results for non-quadratic obstacle problem 4.2 from the Algorithm 2 process in MATLAB. It can be noticed that the ABB with $\alpha_1 = 1$ and $\theta = 1$ is the winner in terms of rapid convergence rate, low number evaluations of functions, and iterations as well. However, it shows the worst nonmonotonic convergence with maximum number of function evaluations and iterations for the second choice of α_1 and θ. While PBB is the best in its results with $\alpha_1 = 1$ and $\theta = \frac{s_k^T s_k}{s_k^T y_k}$ compared to its use with different initial step length and both choices of θ. Furthermore, the PBB method reduces the monotonous behavior of the error especially when $\alpha_1 = 1$ and $\theta = 1$, as can be seen clearly in the 9th refinement level Figures 5 and 6.

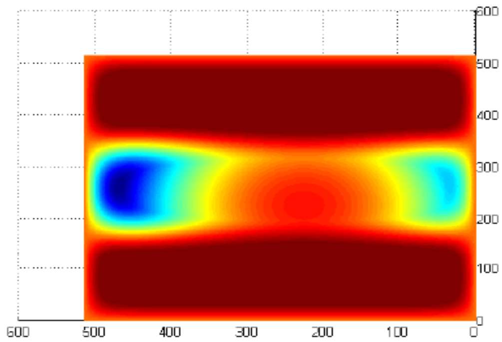

Figure 4: Problem 4.2, solution of bounded non-quadratic problem.

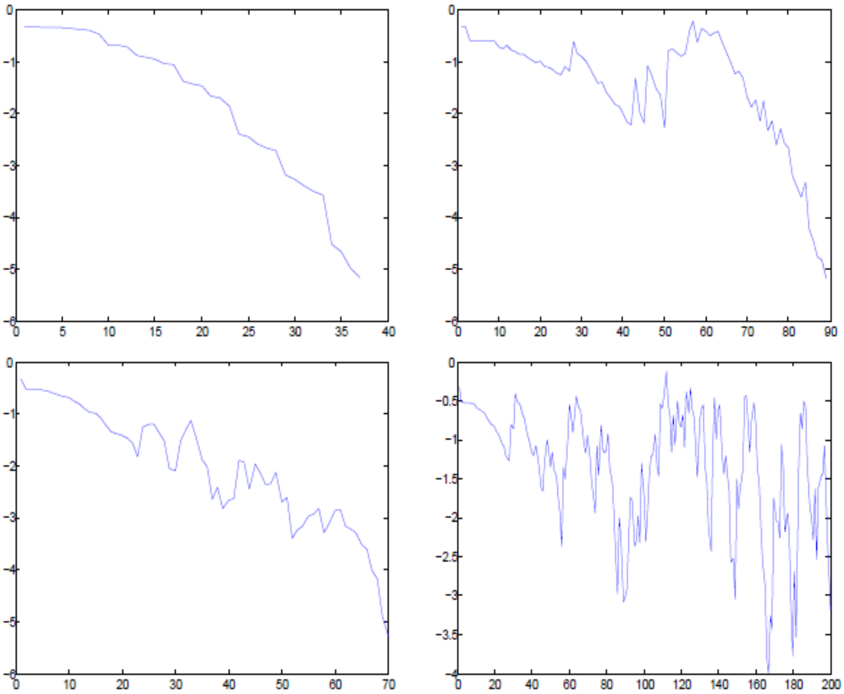

Figure 5: Problem 4.2, 9th refinement level, from top, ABB error (left) with $\theta = 1, \alpha_1 = 1$ and ABB error (right), $\theta = \frac{s_k^T - t}{s_k^T y_k}, \alpha_1 = 1$. From down, ABB error (left) with $\theta = 1, \alpha_1 = \frac{1}{\|g_1\|_\infty}$ and ABB error (right), $\theta = \frac{5_k s_k}{5_k y_k}, \alpha_1 = \frac{1}{\|k_1\|_\infty}$.

Table 3: Problem 4.2, Algorithm 2 asymptotic convergence rate, rate, total number of evaluations of top-level functions,feval, total number of iterations, it, and CPU time for 4-9 discretized levels. ABB denotes alternating Barzilai-Berwein step length alongside spectral conjugate gradient direction with $\theta = 1$ and $\theta = \frac{s_k s_k}{s_k s_k}$ and the tolerance criteria is less than 10^{-5}.

Algorithm 2, $\alpha_1 = 1$	ABB, $\theta = 1$				ABB, $\theta = \frac{s_k^T s_k}{s_k^3 y_k}$			
Levels	rate	feval	it	CPU	rate	feval	it	CPU
4	0.15	54	11	0.07 s	0.38	119	15	0.03 s
5	0.27	104	15	0.03 s	0.17	275	24	0.15 s
7	0.28	209	21	0.91 s	0.72	860	42	4.32 s
8	0.42	377	28	7.52 s	0.50	1952	63	41.53 s
9	0.47	702	38	52.16 s	0.42	4004	90	288.40 s
Algorithm 2, $\alpha_1 = \frac{1}{\|s\|_\infty}$	ABB, $\theta = 1$				ABB, $\theta = \frac{s_k^T s_k}{s_k^2 y_k}$			
Levels	rate	feval	it	CPU	rate	feval	it	CPU

4	0.20	87	11	0.03 s	0.27	104	14	0.04 s
5	0.21	119	15	0.06 s	0.48	275	24	0.14 s
7	0.40	464	31	2.28 s	0.39	1377	53	6.36 s
8	0.31	1080	47	22.58 s	0.18	7625	124	168.39 s
9	0.44	2484	70	176.45 s	0.46	20532	200	1540.90 s
Algorithm 2, $\alpha_1 = 1$	PBB, $\theta = 1$				PBB, $\theta = \frac{s_k^s s_k}{s_k y_k}$			
Levels	rate	feval	it	CPU	rate	feval	it	CPU
4	0.22	77	13	0.02 s	0.32	119	16	0.03 s
5	0.38	189	20	0.05 s	0.2	2	21	0.06 s
7	0.58	594	35	2.64 s	0.45	433	30	2.11 s
8	0.69	1377	53	32.75 s	0.55	1377	53	18.61 s
9	0.73	3002	78	153.33 s	0.56	2849	78	132.50 s
Algorithm 2, $\alpha_1 = \frac{1}{\|k_1\|_\infty}$	PBB, $\theta = 1$				PBB, $\theta = \frac{s_k s_k}{s_k y_k}$			
Levels	rate	feval	it	CPU	rate	feval	it	CPU
4	0.17	54	10	0.03 s	0.33	135	16	0.03 s
5	0.22	119	15	0.06 s	0.35	299	25	0.16 s
7	0.42	665	37	3.56 s	0.51	1175	49	6.29 s
8	0.46	1485	55	36.19 s	0.53	2484	71	60.21 s
9	0.68	4185	92	340.97 s	0.48	9590	139	804.63 s

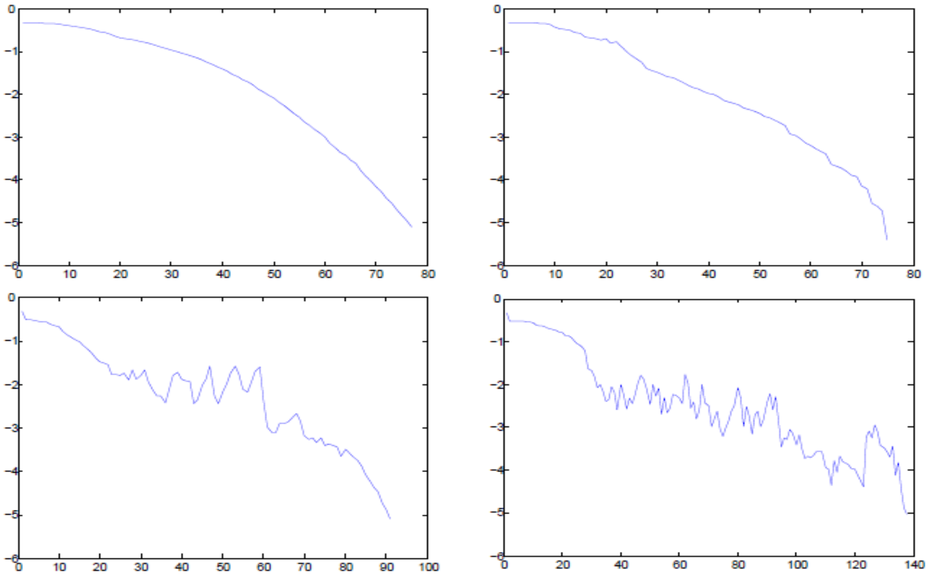

Figure 6: Problem 4.2, 9th refinement level, from the top, PBB error (left) with $\theta = 1, \alpha_1 = 1$ and PBB error (right), $\theta = \frac{s_k^T * k}{s_k^T \mu_k}, \alpha_1 = 1$. From down, ABB error (left) with $\theta = 1, \alpha_1 = \frac{1}{\|g_1\|_\infty}$ and PBB error (right), $\theta = \frac{s_k^T s_k}{s_k^T y_k}, \alpha_1 = \frac{1}{\|s_1\|_\infty}$.

4.3 Obstacle problem

Let's look at one last problem that has both an additional equality constraint and an obstacle. The nonlinear PDE is the source of the issue

$$\begin{aligned} -\Delta u - u^2 &= f(x) && \text{in } \Omega \\ u &= 0 && \text{on } \partial\Omega \end{aligned}$$

and can be demonstrated as the subsequent optimization problem

$$\min_{u \in H_0^1(\Omega)} \mathcal{Y}(u) := \frac{1}{2}\int_\Omega \left(\| \nabla u \|^2 - \frac{1}{3} u^3 \right) dx - \int_\Omega Fudx$$

Subject to

$$\varphi \leq u \text{ a.e. in } \Omega$$

$$\int_\Omega udx = 1,$$

with $F \equiv 0$ and

$$\varphi(x_1, x_2) = -32(x_1 - 0.5)^2 - 32(x_2 - 0.5)^2 + 2.5.$$

Figure 7 displays the solution on the 9th refinement using PSCGBB with 262144 elements and 261121 variables of Problem 4.3.

Table 4 shows the numerical results for Problem 4.3, the methods ABB and PBB converge in general with high accuracy until the 8th refinement mesh. whilst for the 9th refinement, the convergence is either diverge or slow and can just reach the solution with low accuracy maximum 10^{-3}, for example, the PBB method with $\theta = 1, \alpha_1 = 1$ Figure 8 . However, ABB method converges nonmonotocally until 9 th refinement mesh with high accuracy just when $\theta = 1, \alpha_1 = \frac{1}{\|s_1\|_\infty}$ and doesnt converge with spectal term $\theta = \frac{s_k^T s_k}{s_k^T y_k}$ except for lower refinement mesh Figure 8.

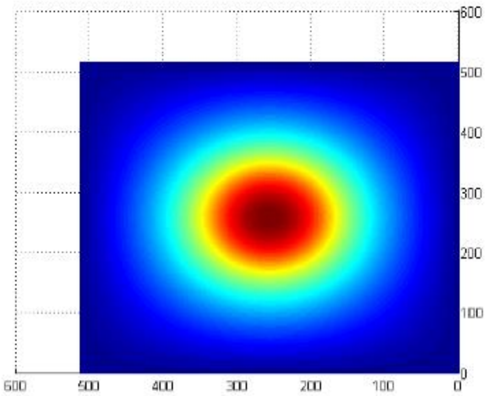

Figure 7: Problem 4.3, solution on the 9th refinement.

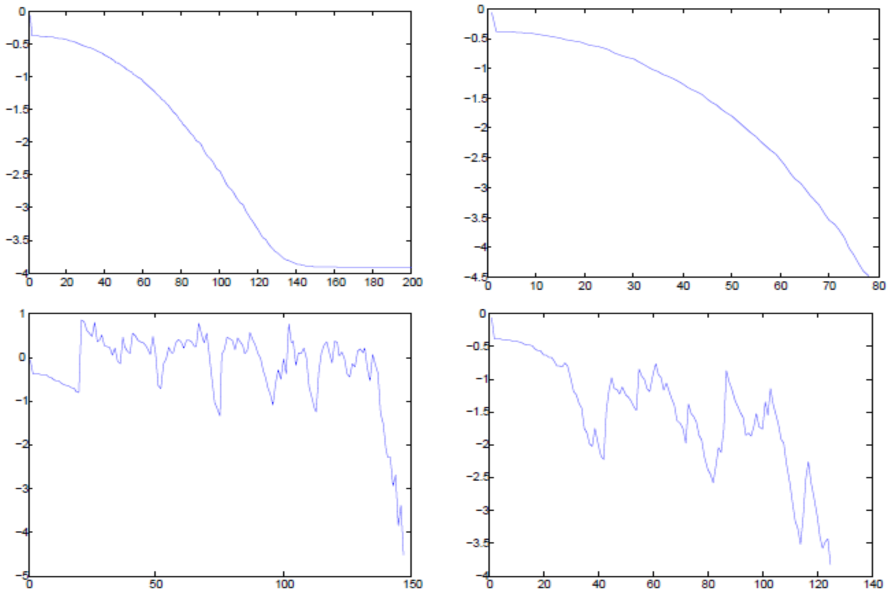

Figure 8: Problem 4.3, from top, PBB error (left) with $\theta = 1, \alpha_1 = 1$ with 9 th refinement level and 8th refinement level (right). From down, ABB error (left) with $\theta = \frac{s_k^T - s_2}{s_k^2 y_k}, \alpha_1 = 1$ on the 9 th refinement level and ABB error (right) with $\theta = 1, \alpha_1 = \frac{1}{1 g_1 1/\sqrt{0}}$ on the 9 th refinement level.

Table 4: Problem 4.3, Algorithm 2 asymptotic convergence rate, rate, number of evaluations of top-level functions, feval, total number of iterations, it, and CPU time for 4-9 discretized levels. ABB stands for alternating projecting Barzilai-Berwein step length alongside spectral conjugate gradient direction with $\theta = 1$ and $\theta = \frac{s_k^T * k}{s_k^T)}, 4e^{-5}$ and PBB stands for projecting Barzilai-Berwein step length.

Algorithm 2, $\alpha_1 = 1$	ABB, $\theta = 1$				ABB, $\theta = \frac{s_k^T s_k}{s_k^T y_k}$			
Levels	rate	feval	it	CPU	rate	feval	it	CPU
4	0.16	43	7	0.01 s	0.17	90	14	0.02 s
5	0.18	65	11	0.02 s	0.21	209	21	0.08 s
7	0.42	377	28	1.85 s	0.36	10991	147	34.34
8	0.68	902	43	17.35 s	-	-	-	-
9	converges	with	low	accuracy	-	-	-	-
Algorithm 2, $\alpha_1 = \frac{1}{\|g_1\|_\infty}$	ABB, $\theta = 1$				ABB, $\theta = \frac{s_k s_k}{s_k^2 y_k}$			
Levels	rate	feval	it	CPU	rate	feval	it	CPU
4	0.15	60	10	0.02 s	0.28	120	14	0.03 s
5	0.26	90	14	0.05 s	0.53	209	20	0.08 s
7	0.48	594	37	2.25 s	-	-	-	-
8	0.59	2099	62	39.57 s	-	-	-	-
9	0.71	8375	126	1698.94 s	-	-	-	-
Algorithm 2, $\alpha_1 = 1$	PBB, $\theta = 1$				PBB, $\theta = \frac{s_k^T s_k}{s_k^T y_k}$			
Levels	rate	feval	it	CPU	rate	feval	it	CPU
4	0.23	54	11	0.02 s	0.28	90	14	0.25 s
5	0.33	135	17	0.05 s	0.30	135	17	0.13 s
7	0.62	1034	46	4.57 s	0.57	819	41	4.31
8	0.74	3289	79	71.55 s	0.82	4559	96	115.58 s
9	0.99	11324	200	271.11 s	0.95	11324	200	12.19 s

Algorithm 2, $\alpha_1 = \frac{1}{\lVert g_1 \rVert_\infty}$	PBB, $\theta = 1$				PBB, $\theta = \frac{s_k^T s_k}{s_k^T y_k}$			
Levels	rate	feval	it	CPU	rate	feval	it	CPU
4	0.17	54	11	0.02 s	0.30	104	15	0.03 s
5	0.26	77	17	0.05 s	0.23	152	18	0.06 s
7	0.42	527	44	2.89 s	0.62	1484	55	4.69 s
8	0.53	2849	75	42.84 s	0.75	6203	109	135.68 s
9	converges	with	low	accuracy	converges	with	low	accuracy

Table 5: Problem 4.1, 4.2 and 4.3 using the standard projected gradient method with backtracking Armijo line search [14], asymptotic convergence rate, rate, number of evaluations of top-level functions, feval, total number of iterations, it, and CPU time for 4-9 discretized levels.

Armijo	Problem 4.1				Problem 4.2				Problem 4.3			
Levels	rate	feval	it	CPU	rate	feval	it	CPU	rate	feval	it	CPU
4	0.12	152	10	0.02 s	0.36	526	16	0.06 s	0.27	386	15	0.20 s
5	0.27	531	17	0.07 s	0.41	1481	27	0.27 s	0.72	1499	26	0.44 s
7	0.70	7464	58	15.41 s	0.78	6183	49	25.26 s	0.49	10980	67	35.28 s
8	0.84	28572	112	337.09 s	0.68	11206	64	195.48 s	0.64	34715	100	512.26 s
9	0.92	92262	200	1733.05 s	0.75	33461	96	1146.17 s	-			

5 Conclusions

The preceding section's experiments demonstrate that the PBB method is typically monotone and convergent for problems involving quadratic, nonlinear, and even obstacles. This is particularly true when using the first option of initial step length $\alpha_1 = 1$. Applying the PBB method with the second option, $\alpha = 1/\lVert g_1 \rVert_\infty$, yields a better nonmonotone convergent process for all three problem types, but at the cost of more CPU time, iterations, and function evaluations. The optimal nonmonotone convergent, with fewer function evaluations, iterations, and CPU time, is the ABB method for solving quadratic and nonlinear problems. However, this cannot be true for obstacle problems, which have the potential to diverge and fail when dealing with applicants who have $\theta = \frac{s_k^T s_k}{s_k^T y_k}$. In light of this, we used Amijo back-tracking to present the results for the problems from Table 5 , which shows a high number of function evaluations when compared to the ABB and PBB methods [14]. The stopping criteria for all the experiments were set to 10^{-5} and 4×10^{-5} specifically for the obstacle problem 4.3 .

Therefore, This combination of the well-known method spectral conjugate gradient for unconstrained optimization with Barzilai-Borwein step lengths results in a method that can be used efficiently for solving bounded convex optimization problems especially those that arise from infinite-dimensional problems. Thus, the primary benefits of the suggested approach lie in its minimal memory needs, low computational complexity, and reduced number of iterations needed to solve large-scale convex problems.

In contrast to previous studies in the spectral conjugate gradient field, the new approach solves large-scale problems effectively and to good accuracy without the need for complex line searches. Nevertheless, the technique can only be applied to convex problems with bound constraints and still be convergent. However, we think that the presented method, when applicable, is one of the effective methods to solve the wide class of convex optimization problems that are both bound-constrained and unconstrained.

Funding: No funding were received for this work.
Conflict of interest: I hereby confirm that all figures, tables, and measurements in the manuscript are all original. For We used MATLAB version 8.0 .0 (2012b) running in 64-bit Windows 10 on a laptop equipped with an Intel Core i7-3570 CPU M 620 at 2.67 GHz and 4 GB RAM for all of the experiments.

References

[1] M Al-Baali and I S Latif. Combined conjugate gradient and quasi-newton methods for unconstrained optimization. Journal of Iraqi Al-Khwarizmi, 6(2), 2022.

[2] A M Awwal, P Kumam, and A B Abubakar. Spectral modified polak-ribiére-polyak projection conjugate gradient method for solving monotone systems of nonlinear equations. Applied Mathematics and Computation, 362:124514, 2019.

[3] E G Birgin and J M Martínez. A spectral conjugate gradient method for unconstrained optimization. Applied Mathematics and optimization, 43(2):117-128, 2001.

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

[5] W L Briggs, V E Henson, and S F McCormick. A multigrid tutorial. SIAM, 2000.

[6] P H Calamai and J J Moré. Projected gradient methods for linearly constrained problems. Mathematical programming, 39(1):93-116, 1987.

[7] Y H Dai and R Fletcher. Projected Barzilai-Borwein methods for large-scale boxconstrained quadratic programming. Numerische Mathematik, 100(1):21-47, 2005.

[8] Y H Dai and IZ Liao. R-linear convergence of the barzilai and borwein gradient method. IMA Journal of Numerical Analysis, 22(1):1-10, 2002.

[9] YH Dai and H Zhang. Adaptive two-point stepsize gradient algorithm. Numerical Algorithms, 27:377-385, 2001.

[10] J E Dennis and R B Schnabel. Numerical methods for unconstrained optimization and nonlinear equations. SIAM, 1996.

[11] P Faramarzi and K Amini. A modified spectral conjugate gradient method with global convergence. Journal of Optimization Theory and Applications, 182:667-690, 2019.

[12] A Friedlander, J M Martínez, B Molina, and M Raydan. Gradient method with retards and generalizations. SIAM Journal on Numerical Analysis, 36(1):275-289, 1998.

[13] C Graser and R Kornhuber. Multigrid methods for obstacle problems. J. Comput. Math, 27(1):1-44, 2009.

[14] M Kočvara and S Mohammed. A first-order multigrid method for bound-constrained convex optimization. Optimization Methods and Software, 31(3):622-644, 2016.

[15] J K Liu, Y M Feng, and L M Zou. A spectral conjugate gradient method for solving large-scale unconstrained optimization. Computers & Mathematics with Applications, 77(3): 731 − 739,2019.

[16] J L Morales and J Nocedal. Remark on Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization". ACM Transactions on Mathematical Software (TOMS), 38(1):7, 2011.

[17] A Perry. A modified conjugate gradient algorithm. Operations Research, 26(6): 10731078,1978 [18] M Raydan. On the barzilai and borwein choice of steplength for the gradient method. IMA Journal of Numerical Analysis, 13(3):321-326, 1993.

[19] E L Sadraddin and I S Latif. A matrix form of spectral scaling in quasi-newton algorithm. Palestine Journal of Mathematics, 12, 2023.

[20] T Serafini, G Zanghirati, and L Zanni. Gradient projection methods for quadratic programs and applications in training support vector machines. Optimization Methods and Software, 20(2-3):353-378, 2005.

[21] Z Sun, H Li, J Wang, and Y Tian. Two modified spectral conjugate gradient methods and their global corvergence for unconstrained optimization. International Journal of Computer Mathematics, 95(10): 2082-2099, 2018.

[22] R Tavakoli and H Zhang. A nonmonotone spectral projected gradient method for large-scale topology optimization problems. arXiv preprint arXiv:I006.0561, 2010.

[23] Z Wan, J Guo, J Liu, and W Liu. A modified spectral conjugate gradient projection method for signal recovery. Signal, Image and Video Processing, 12: 1455-1462, 2018.

[24] Z Wan, Z Yang, and Y Wang. New spectral prp conjugate gradient method for unconstrained optimization. Applied Mathematics Letters, 24(1):16-22, 2011.