Iranian Journal of Numerical Analysis and Optimization
Vol. 16, No. 3, 2026, pp 790–814
https://doi.org/10.22067/ijnao.2026.96402.1775
https://ijnao.um.ac.ir/

Research Article

Adaptive-batch stochastic gradient descent for constrained optimization based on relaxed barrier functions

A. Fillali*, S. Addoune and R. Zeghdane

Abstract

Stochastic gradient descent (SGD) is the cornerstone of large-scale optimization; however, its application to problems with a vast number of constraints remains a significant challenge. Methods based on relaxed logarithmic barrier functions have enabled the use of SGD by sampling constraints, yet the reliance on single samples in these methods often leads to high variance and oscillations that hinder progress near the optimal solution.

*Corresponding author

Received 9 November 2025; revised 17 February 2026; accepted 17 February 2026

Abdelouakil Fillali
Laboratoire d'Analyse Mathématiques et ses Applications (LAMA), Department of Mathematics, University Mohamed El Bachir El Ibrahimi, El Anasser, Bordj Bou Arreridj, 34030, Algeria. e-mail: abdelouakil.fillali@univ-bba.dz

Smail Addoune
Laboratoire d'Analyse Mathématiques et ses Applications (LAMA), Department of Mathematics, University Mohamed El Bachir El Ibrahimi, El Anasser, Bordj Bou Arreridj, 34030, Algeria. e-mail: smail.addoune@univ-bba.dz

Rebiha Zeghdane
Laboratoire d'Analyse Mathématiques et ses Applications (LAMA), Department of Mathematics, University Mohamed El Bachir El Ibrahimi, El Anasser, Bordj Bou Arreridj, 34030, Algeria. e-mail: rebiha.zeghdane@univ-bba.dz

How to cite this article
Fillali, A., Addoune, S. and Zeghdane, R., Adaptive-batch stochastic gradient descent for constrained optimization based on relaxed barrier functions. *Iran. J. Numer. Anal. Optim.*, 2026; 16(3): 790-814. https://doi.org/10.22067/ijnao.2026.96402.1775

In this paper, we propose a novel hybrid algorithm, AdaBS-RBSGD, which integrates the relaxed barrier framework with an adaptive batch size strategy. Our algorithm dynamically adjusts the batch size B_k to be inversely proportional to the estimated optimization loss, systematically reducing gradient variance as the iterates approach the solution.

We provide a rigorous mathematical proof establishing the almost-sure convergence of this hybrid algorithm. Furthermore, our theoretical analysis of error bounds demonstrates that active variance management allows the error term associated with stochastic noise to vanish more rapidly compared to the baseline method. Empirically, results on a benchmark problem show that the proposed algorithm significantly outperforms its single-sample counterpart, successfully breaking the "noise floor," achieving higher stability, and reaching a more accurate final solution. This work presents a practical and theoretically sound methodology for accelerating and improving the precision of stochastic constrained optimization.

AMS subject classifications (2020): Primary 90C15, 90C25; Secondary 49M30, 65K05, 68T05.

Keywords: Constrained optimization, Stochastic gradient descent (SGD), Relaxed logarithmic barrier functions, Adaptive batch sizes, Variance reduction, Large-scale optimization.

1 Introduction

Stochastic gradient descent (SGD) and its variants have become cornerstone algorithms in modern large-scale optimization, driving progress in fields ranging from machine learning to networked systems [4, 14, 25]. These methods are fundamentally rooted in the pioneering work on stochastic approximation by Robbins and Monro [16] and later acceleration techniques by Polyak and Juditsky [15]. While SGD is exceptionally efficient for unconstrained problems [10], its effective application to constrained optimization remains an active area of research [19].

Many classical approaches require full knowledge of the constraint set at each iteration, often necessitating computationally expensive projection steps that are prohibitive when the number of constraints is very large [22]. To overcome this challenge, recent methods have been developed that rely on sampling only a subset of constraints [12, 23, 24]. A particularly promising direction is the adaptive relaxed barrier SGD algorithm proposed by Dimitrieski, Cao, and Ebenbauer [7]. This approach avoids expensive projections by using a carefully adapted relaxed barrier function [8, 9], allowing for sampling both the objective function and the constraints.

While the baseline algorithm guarantees almost-sure convergence [7], it suffers from a critical limitation: the inherent high variance of single-sample gradients causes significant oscillations and stagnation near the optimal solution. To overcome this fundamental bottleneck, we introduce AdaBS-RBSGD (Adaptive-batch relaxed barrier SGD), a novel hybrid algorithm that explicitly

targets this variance. Our key contribution lies in the seamless integration of relaxed barrier functions with a dynamic batch-size adaptation strategy, building upon the foundations laid by [1, 2, 3, 20, 21]. Specifically, unlike traditional methods, our approach adopts the logic from [20] to actively suppress gradient noise by inversely scaling the batch size B_k with the optimization loss —a technique shown to be a powerful alternative to decaying the learning rate [21]. This allows the algorithm to break the "noise floor" and enable high-precision convergence without the computational burden of full constraints. We provide a rigorous theoretical analysis establishing the almost-sure convergence of this hybrid method and demonstrate its clear theoretical and empirical advantages in variance reduction.

2 Problem statement and hybrid algorithm

2.1 Problem statement

In this paper, we consider the constrained optimization problem of finding a minimizer for a finite sum objective function, subject to a large number of inequality constraints. The problem is formally stated as:

$$\min_{x\in\mathbb{R}^d} f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x) \quad \text{s.t.} \quad g_j(x) \leq 0, \quad j \in \{1,\dots,m\}, \tag{1}$$

where we assume that the objective functions f_i and constraint functions g_j are continuously differentiable ($\mathcal{C}^1$) functions. We adopt the following standard assumptions [5, 6, 18]:

Assumption 1 (Objective function properties)**.** The objective function f and its components f_i are assumed to possess the following properties for constants $L \geq \mu > 0$ and for any $x, y \in \mathbb{R}^d$:

1. **L-Smoothness:** Each component $f_i(x)$ is L-smooth, meaning its gradient is Lipschitz continuous with constant L:

$$f_i(y) \leq f_i(x) + \nabla f_i(x)^T(y-x) + \frac{L}{2}||y-x||^2$$

2. **convexity:** Each component $f_i(x)$ is convex:

$$f_i(y) \geq f_i(x) + \nabla f_i(x)^T(y-x)$$

3. **Strong Convexity:** The total function $f(x)$ is μ-strongly convex:

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{\mu}{2} ||y - x||^2.$$

These assumptions, although restrictive, are not uncommon in convergence proofs for gradient-based optimization algorithms [5, 7, 14].

Assumption 2 (Constraint functions)**.** The constraint functions $g_j(x)$ are affine, defined as:

$$g_j(x) := a_j^T x + b_j,$$

where $a_j \in \mathbb{R}^d$ and $b_j \in \mathbb{R}$. The feasible set $\mathbb{X} := \{x \in \mathbb{R}^d : g_j(x) \leq 0, \forall j\}$ is assumed to have a nonempty interior, which is a fundamental condition for applying barrier-based methods [5, 9]

Assumption 3 (Stochastic batch sampling)**.** At each iteration k, given a batch size $B_k \geq 1$, we sample:

- A batch of objective indices: $\mathcal{I}_k = \{i_s\}_{s=1}^{B_k}$, where each i_s is drawn independently and uniformly at random from $\mathcal{U}\{1, \ldots, n\}$.
- A batch of constraint indices: $\mathcal{J}_k = \{j_s\}_{s=1}^{B_k}$, where each j_s is drawn independently and uniformly at random from $\mathcal{U}\{1, \ldots, m\}$.

This uniform distribution is chosen to ensure that the stochastic batch average gradients are unbiased estimators of the true gradients, thereby satisfying the standard assumptions for SGD algorithms [16]. This is demonstrated by the following function:

1. For the objective function: The expected value of the sampled stochastic batch gradient is equal to the true gradient of the full function [7, 16]:

$$\mathbb{E}\left[\frac{1}{B_k}\sum_{s=1}^{B_k} \nabla f_{i_s}(x)\right] = \frac{1}{B_k}\sum_{s=1}^{B_k} \mathbb{E}[\nabla f_{i_s}(x)] = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x) = \nabla f(x).$$

2. For the barrier function: The expected value of the sampled stochastic batch gradient of the barrier function equals the average effect of the gradients over all constraints [7]:

$$\begin{aligned}\mathbb{E}\left[\frac{1}{B_k}\sum_{s=1}^{B_k} \nabla B(g_{j_s}(x), \delta)\right] &= \frac{1}{B_k}\sum_{s=1}^{B_k} \mathbb{E}[\nabla B(g_{j_s}(x), \delta)] \\ &= \frac{1}{m}\sum_{j=1}^{m} \nabla B(g_j(x), \delta).\end{aligned}$$

To handle the constraints stochastically, we use the same relaxed logarithmic barrier function $B(z, \delta)$ as defined in [7]. The core idea is to solve a sequence of unconstrained problems:

$$\min_{x \in \mathbb{R}^d} \Phi_k(x) := f(x) + \frac{1}{m} \sum_{j=1}^{m} B(g_j(x), \delta_k),$$

where the barrier parameter $\delta_k = \delta_\infty + \epsilon_k$ is adapted at each iteration k.

For completeness, we state the formulas for $B(z, \delta)$ and its gradient here, as adopted from [7]. The function is defined piecewise as:

$$B(z, \delta) = \begin{cases} -\delta \ \log(-z) & \text{if } z < -\delta, \\ \frac{1}{2}\left(\frac{(z+2\delta)^2}{\delta} - \delta\right) - \delta \ \log(\delta) & \text{if } z \geq -\delta. \end{cases}$$

Its gradient with respect to z, $\nabla_z B(z, \delta)$, used in the SGD updates, is:

$$\nabla_z B(z, \delta) = \begin{cases} -\delta/z & \text{if } z < -\delta, \\ (z + 2\delta)/\delta & \text{if } z \geq -\delta. \end{cases}$$

2.2 The baseline algorithm

The baseline algorithm proposed by Dimitrieski, Cao, and Ebenbauer [7] is a single-sample SGD method. This corresponds to a special case of Assumption 3 where the batch size is fixed to $B_k = 1$ for all k. At each iteration k, it samples one objective index $i_k \sim \mathcal{U}\{1, \ldots, n\}$ and one constraint index $j_k \sim \mathcal{U}\{1, \ldots, m\}$, and updates x_k using the gradient G_k^{base}:

$$G_k^{base} = \nabla f_{i_k}(x_k) + \nabla B(g_{j_k}(x_k), \delta_k), \tag{2}$$

$$x_{k+1} = x_k - \gamma_k G_k^{base}. \tag{3}$$

As shown in our results, the high variance of G_k^{base} hinders the optimization process, limiting the algorithm to the standard SGD convergence rate of $\mathcal{O}(1/\sqrt{k})$ and causing significant oscillations near the optimal solution.

2.3 The proposed hybrid algorithm

To address the high variance of the baseline, we propose a hybrid algorithm, AdaBS-RBSGD, which integrates the relaxed barrier framework with an adaptive batching mechanism [20]. Instead of a single sample, our method draws a batch of B_k indices: $\mathcal{I}_k = \{i_1, \ldots, i_{B_k}\}$ and $\mathcal{J}_k = \{j_1, \ldots, j_{B_k}\}$. The batch gradient G_k is the average:

$$G_k = \frac{1}{B_k} \sum_{i \in \mathcal{I}_k} \nabla f_i(x_k) + \frac{1}{B_k} \sum_{j \in \mathcal{J}_k} \nabla B(g_j(x_k).\delta_k) \tag{4}$$

The batch size B_k is adapted based on the practical batch loss, L_k^{batch}, which is defined as follows:

$$L_k^{batch} = \frac{1}{B_k} \sum_{i \in \mathcal{I}_k} f_i(x_k) + \frac{1}{B_k} \sum_{j \in \mathcal{J}_k} B(g_j(x_k), \delta_k). \tag{5}$$

We follow the RADADAMP logic from [20] to update the batch size using this loss. The process involves tracking a "noise" term t_k and its exponential moving average $\hat{d}_k$.

First, t_k is calculated as a proxy for the 'optimization noise', using the batch loss L_k^{batch} and batch gradient G_k:

$$t_k = L_k^{batch} + \lambda ||G_k||^2. \tag{6}$$

Next, this term is smoothed using an exponential moving average $\hat{d}_k$:

$$\hat{d}_k = \rho \hat{d}_{k-1} + (1 - \rho) t_k. \tag{7}$$

Finally, the batch size for the next iteration, B_{k+1}, is computed to be inversely proportional to this average $\hat{d}_k$.

The intuition is that as the algorithm approaches the optimum, both the batch loss L_k^{batch} and the gradient norm $||G_k||^2$ will decrease, leading to a smaller t_k and consequently a smaller $\hat{d}_k$. This smaller $\hat{d}_k$ will, in turn, lead to a larger batch size B_{k+1}, actively reducing variance as the solution is approached. The conceptual feedback loop of this process is illustrated in Figure 1. The full algorithm is detailed in Algorithm 1.

⋆ Parameters ρ and λ are adopted from the RADADAMP logic [20]. ρ is the 'memory' or smoothing factor for the exponential moving average $\hat{d}_k$ (7), while λ is a balancing coefficient for the 'noise' term t_k (6) which combines the loss L_k^{batch} and the gradient norm $||G_k||^2$.

Algorithm 1 AdaBS-RBSGD (Hybrid adaptive-batch relaxed barrier SGD)

Require: Initial point $x_0 \in \mathbb{R}^d$, total iterations $K \in \mathbb{N}$, initial batch size $B_0 \in \mathbb{N}_{\geq 1}$, max batch size $B_{\max} \in \mathbb{N}_{\geq B_0}$, step-size schedule $\{\gamma_k\}_{k=0}^{K-1} \subset \mathbb{R}^+$, barrier schedule $\{\epsilon_k\}_{k=0}^{K-1} \subset \mathbb{R}^+$, adaptive parameters $\rho \in (0,1)$, $\lambda \in \mathbb{R}^+$, $\delta_\infty \in \mathbb{R}^+$.*

Initialize: $B \leftarrow B_0$, $\hat{d} \leftarrow 0.0$, $d_0 \leftarrow -1.0$

for $k = 0$ to $K-1$ **do**

// Phase 1: Setup Batch Size and Learning Rate

$B_k \leftarrow \lceil B \rceil$

$\gamma'_k \leftarrow \gamma_k$

if $B_k \geq B_{\max}$ **then**

$\gamma'_k \leftarrow \gamma_k \cdot (B_{\max}/B_k)$

$B_k \leftarrow B_{\max}$

end if

Set barrier parameter $\delta_k \leftarrow \delta_\infty + \epsilon_k$

// Phase 2: Compute Stochastic Batch Gradient and Loss

Sample $\mathcal{I}_k = \{i_1, \ldots, i_{B_k}\} \sim \mathcal{U}\{1:n\}^{B_k}$

Sample $\mathcal{J}_k = \{j_1, \ldots, j_{B_k}\} \sim \mathcal{U}\{1:m\}^{B_k}$

$G_k \leftarrow \frac{1}{B_k} \sum_{i \in \mathcal{I}_k} \nabla f_i(x_k) + \frac{1}{B_k} \sum_{j \in \mathcal{J}_k} \nabla B(g_j(x_k), \delta_k)$

$L_k^{batch} \leftarrow \frac{1}{B_k} \sum_{i \in \mathcal{I}_k} f_i(x_k) + \frac{1}{B_k} \sum_{j \in \mathcal{J}_k} B(g_j(x_k), \delta_k)$

// Phase 3: Update Primary Iterate

$x_{k+1} \leftarrow x_k - \gamma'_k G_k$

// Phase 4: Update Batch Size Logic (from [20])

$t_k \leftarrow L_k^{batch} + \lambda \|G_k\|^2$

if $k = 0$ **then**

$\hat{d} \leftarrow t_k$

$d_0 \leftarrow t_k$

else

$\hat{d} \leftarrow \rho \hat{d} + (1-\rho) t_k$

if $\hat{d} > 10^{-8}$ **and** $d_0 > 0$ **then**

$B \leftarrow B_0 \cdot (d_0/\hat{d})$

else

$B \leftarrow B_{\max}$

end if

end if

end for

return x_K

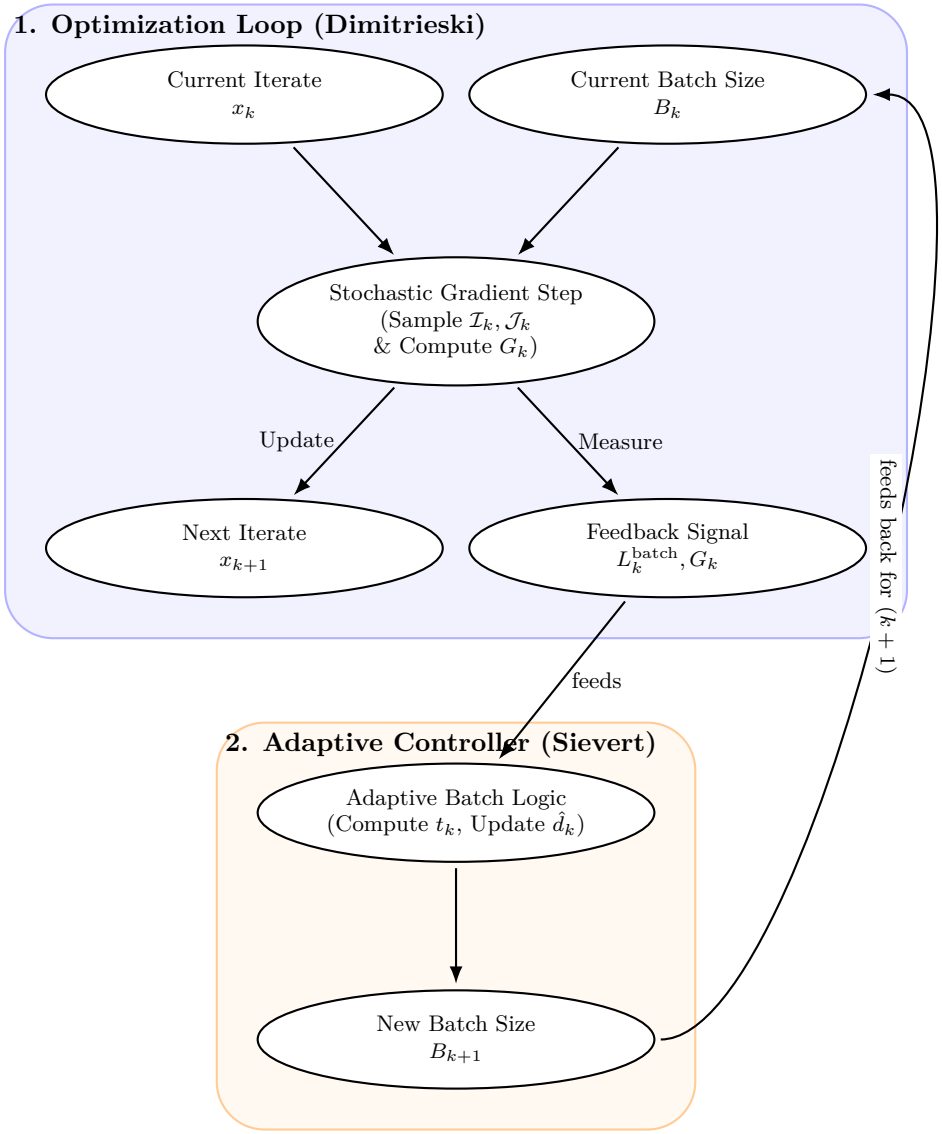

Figure 1: Conceptual diagram of the proposed `AdaBS-RBSGD` algorithm. The system is a feedback loop: (1) The **Optimization Loop** (blue, from [7]) performs the stochastic update and measures the current loss. (2) The **Adaptive Controller** (orange, from [20]) receives this feedback to compute the batch size B_{k+1} for the next iteration, actively managing the gradient variance.

2.4 Design rationale

The proposed AdaBS-RBSGD algorithm is a nontrivial hybrid designed to address the core weaknesses of its parent methods. The relaxed barrier component [7] provides a theoretically-sound mechanism to handle constraints stochastically. However, its primary weakness is the high variance from single-sample gradients. The adaptive batching component [20] directly targets this weakness. By dynamically increasing B_k as the loss decreases, the algorithm actively suppresses the gradient variance, allowing for more stable and precise steps near the optimum.

Noise Handling and Adaptation Speed: The algorithm manages stochastic noise by continuously monitoring the noise proxy t_k (6), which acts as a real-time signal of the variance in the optimization landscape. The responsiveness of the algorithm to changing noise levels is explicitly controlled by the memory factor ρ in the exponential moving average (7). A lower ρ enables rapid adaptation to sudden noise shifts but may lead to instability, whereas a higher ρ (such as our chosen $\rho = 0.999$) ensures a smoother, more gradual adaptation. This conservative approach effectively filters out transient noise spikes, allowing the batch size to grow steadily only when a persistent trend of variance reduction is detected, thus maintaining algorithmic stability.

Implicit Determination of Batch Growth Rate: Unlike heuristic schedules that enforce a fixed growth rate, the rate of increase for B_k in AdaBS-RBSGD is not predetermined but is fully data-driven. It is governed implicitly by the inverse relationship with the smoothed noise metric $\hat{d}_k$ (7). As the optimization loss decreases, $\hat{d}_k$ diminishes, naturally driving B_k upward. Consequently, the "optimal" rate emerges automatically from the landscape geometry: regions of significant progress induce rapid batch scaling, while stagnation leads to slower growth. This mechanism aligns with the practical guideline established by Smith et al. [21], which posits that increasing the batch size is theoretically equivalent to decaying the learning rate γ_k. Thus, practically, our algorithm decouples the manual tuning of batch schedules from the learning rate, allowing the batch size to adjust automatically based on the solver's progress.

3 Convergence analysis

In this section, we provide the formal statement of the convergence theorem for the proposed AdaBS-RBSGD algorithm (Algorithm 1). The rigorous mathematical proof, structured into logical steps, follows in Section 4.

Theorem 1 (Almost sure convergence of AdaBS-RBSGD)**.** Consider the constrained optimization problem (1) and assume Assumptions 1, 2, and 3 hold. Let $\{X_k\}_{k\geq 0}$ be the sequence of

iterates generated by Algorithm 1. Let $\{\gamma_k\}_{k\geq 0}$ and $\{\epsilon_k\}_{k\geq 0}$ be positive sequences such that $\delta_k = \delta_\infty + \epsilon_k$ (with $\delta_\infty > 0$) satisfying

(a) $\sum_{k=0}^{\infty} \gamma_k = \infty$,

(b) $\sum_{k=0}^{\infty} \gamma_k^2 < \infty$,

(c) $\sum_{k=0}^{\infty} \gamma_k \epsilon_k < \infty$.

Then, the sequence of iterates $\{X_k\}_{k\geq 0}$ converges almost surely (a.s.) to the unique minimizer of the relaxed problem, $x^*(\delta_\infty)$. That is:

$$\lim_{k\to\infty} X_k = x^*(\delta_\infty) \quad \text{a.s.}$$

Remark 1 (Convergence conditions and failure modes)**.** The almost-sure convergence established in Theorem 1 relies strictly on Assumptions 1-3. Consequently, the method guarantees convergence primarily for strongly convex objectives subject to affine constraints, provided the step-size conditions in Theorem 1 are met. The algorithm may fail or exhibit instability under the following specific conditions:

1. **Violation of convexity:** Applying the method to nonconvex objectives violates Assumption 1. Without strong convexity, the drift analysis in the proof (specifically the bound on $\langle \nabla\Phi(X_k), X_k\rangle$) no longer holds, risking entrapment in local minima or divergence.

2. **Nonaffine constraints:** Assumption 2 requires affine constraints to ensure the barrier gradient variance remains quadratically bounded (as shown in (11) of the analysis). With general nonlinear constraints, this bound may be violated, potentially leading to explosive gradient variance.

3. **Incompatible parameter decay:** Convergence requires the specific interplay between step-size and barrier parameters ($\sum \gamma_k \epsilon_k < \infty$). If ϵ_k decays too rapidly relative to γ_k, the bias error term will not vanish, preventing convergence to the exact solution.

4 Proof of almost-sure convergence

In this section, we provide a rigorous, step-by-step convergence analysis of the proposed AdaBS-RBSGD algorithm. We rely on the Robbins-Siegmund supermartingale convergence lemma [17] and the techniques for biased stochastic approximation [11] to establish that the iterate sequence $\{X_k\}$ converges almost surely to the optimal solution x^*.

Proof of Theorem 1. We define the Lyapunov function R_k as the squared Euclidean distance between the current iterate X_k and the unique minimizer $x^* := x^*(\delta_\infty)$ of the limit problem:

$$R_k := \|X_k - x^*\|^2.$$

Let $\mathcal{F}_k$ denote the σ-algebra generated by the random variables $\{X_0, \dots, X_k\}$. Our goal is to analyze the conditional expectation $\mathbb{E}[R_{k+1}|\mathcal{F}_k]$.

Step 1: Expansion of the iteration

Recall the update rule: $X_{k+1} = X_k - \gamma_k G_k$. We expand the squared norm:

$$\begin{aligned} R_{k+1} &= \|X_k - \gamma_k G_k - x^*\|^2 \\ &= \|(X_k - x^*) - \gamma_k G_k\|^2 \\ &= \|X_k - x^*\|^2 - 2\gamma_k \langle G_k, X_k - x^* \rangle + \gamma_k^2 \|G_k\|^2. \end{aligned}$$

Taking the conditional expectation with respect to $\mathcal{F}_k$:

$$\mathbb{E}[R_{k+1}|\mathcal{F}_k] = R_k - 2\gamma_k \langle \mathbb{E}[G_k|\mathcal{F}_k], X_k - x^* \rangle + \gamma_k^2 \mathbb{E}[\|G_k\|^2|\mathcal{F}_k]. \tag{8}$$

Measurability note: Before proceeding, it is important to note that since the current iterate X_k is $\mathcal{F}_k$-measurable (fully determined by the history up to step k), the terms $\nabla\Phi(X_k)$ and $\nabla C_k(X_k)$ are deterministic constants with respect to the conditional expectation $\mathbb{E}[\cdot|\mathcal{F}_k]$. Consequently, terms such as $\|\nabla\Phi(X_k)\|$ satisfy $\mathbb{E}[\|\nabla\Phi(X_k)\| \mid \mathcal{F}_k] = \|\nabla\Phi(X_k)\|$, as the expectation operator acts solely on the random batch indices $\mathcal{I}_k$ and $\mathcal{J}_k$ drawn at the current iteration.

Step 2: Analysis of the expected gradient (Drift Term)

We analyze the expected gradient term $\mathbb{E}[G_k|\mathcal{F}_k]$. The batch gradient G_k is the average of B_k independent samples. Let $\nabla\psi_{i,j}(x) = \nabla f_i(x) + \nabla B(g_j(x), \delta_k)$ be a single sample gradient.

$$\mathbb{E}[G_k|\mathcal{F}_k] = \mathbb{E}\left[\frac{1}{B_k}\sum_{s=1}^{B_k} \nabla\psi_{i_s,j_s}(X_k)\middle|\mathcal{F}_k\right] = \mathbb{E}[\nabla\psi_{i,j}(X_k)|\mathcal{F}_k].$$

We decompose this expectation into the target gradient $\nabla\Phi(X_k)$ and the bias term $\nabla C_k(X_k)$. Following the notation in [7], we define the optimality gap of the potential function as:

$$\Phi_0(X_k) := \Phi(X_k) - \Phi(x^*) \geq 0. \tag{9}$$

Substituting the decomposition back into the inner product term in (8):

$$-\langle \mathbb{E}[G_k|\mathcal{F}_k], X_k - x^* \rangle = -\langle \nabla\Phi(X_k), X_k - x^* \rangle - \langle \nabla C_k(X_k), X_k - x^* \rangle.$$

By the strong convexity of $\Phi(x)$**, we have:

$$-\langle \nabla\Phi(X_k), X_k - x^* \rangle \leq -(\Phi_0(X_k) + \frac{\mu}{2}\|X_k - x^*\|^2).$$

Thus, applying the Cauchy-Schwarz inequality, the drift term is bounded by:

$$-2\gamma_k \langle \mathbb{E}[G_k], \tilde{X}_k \rangle \leq -\gamma_k(2\Phi_0(X_k) + \mu R_k) + 2\gamma_k \|\nabla C_k(X_k)\| \sqrt{R_k}. \tag{10}$$

Step 3: Analysis of the variance (Noise Term)

We analyze $\mathbb{E}[\|G_k\|^2|\mathcal{F}_k]$. Since samples are i.i.d., the variance of the average decreases with B_k:

$$\begin{aligned}\mathbb{E}[\|G_k\|^2|\mathcal{F}_k] &= \|\mathbb{E}[G_k]\|^2 + \mathrm{Var}(G_k|\mathcal{F}_k) \\ &= \|\nabla\Phi(X_k) + \nabla C_k(X_k)\|^2 + \frac{1}{B_k}\mathrm{Var}(\nabla\psi_{i,j}(X_k)).\end{aligned}$$

Using the inequality $\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$ and the bound on single-sample variance:

$$\gamma_k^2 \mathbb{E}[\|G_k\|^2|\mathcal{F}_k] \leq \gamma_k^2 \left(2\|\nabla C_k(X_k)\|^2 + 2\|\nabla\Phi(X_k)\|^2 + \frac{\sigma^2}{B_k}\right). \tag{11}$$

Crucially, the stochastic noise term $\frac{\sigma^2}{B_k}$ is explicitly reduced by the batch size.

Step 4: Synthesis and convergence

Substituting the bounds from (10) and (11) into (8), and utilizing the upper bounds for $\|\nabla C_k(X_k)\|$ and $\|\nabla\Phi(X_k)\|^2$ established in [7] (specifically in [7, Eqs. (8)–(10)]), we derive the following supermartingale inequality:

$$\mathbb{E}[R_{k+1}|\mathcal{F}_k] \leq (1 - \gamma_k\mu + \gamma_k\xi_k)R_k$$

** The function $\Phi(x)$ is strongly convex as it is the sum of a strongly convex function $f(x)$ (Assumption 1) and a sum of convex functions. See [7], Appendix V-D, for the formal proof.

$$- 2\gamma_k(1 - 4\gamma_k\hat{L})\Phi_0(X_k) + \underbrace{\left(\beta_k(\epsilon_k) + \frac{\gamma_k^2\sigma^2}{B_k}\right)}_{Q_k}, \tag{12}$$

where $\hat{L}$ is the modified smoothness constant, and Q_k collects the error terms[***]. We verify the conditions for Robbins-Siegmund:

1. The sequence $\gamma_k\xi_k$ is summable since $\sum\gamma_k\epsilon_k < \infty$.
2. The noise term Q_k is summable since $\sum\gamma_k^2 < \infty$ and $B_k \geq 1$.
3. For sufficiently large k, $(1 - 4\gamma_k\hat{L}) > 0$.

Therefore, R_k converges almost surely to a random variable R_∞. Furthermore, we have $\sum_{k=0}^{\infty}\gamma_k\Phi_0(X_k) < \infty$. Given that $\sum\gamma_k = \infty$, this implies $\liminf_{k\to\infty}\Phi_0(X_k) = 0$. Since Φ_0 is continuous and nonnegative, and R_k converges, we conclude $X_k \to x^*(\delta_\infty)$ almost surely. □

Remark 2 (Theoretical advantage)**.** The derivation in Step 3 highlights the theoretical advantage of the hybrid approach over the single-sample baseline proposed in [7]. Specifically, the variance term for the hybrid algorithm behaves as:

$$Q_{var}^{hybrid} \sim \mathcal{O}\left(\frac{\gamma_k^2(\sigma_\Phi^2 + \sigma_C^2(\epsilon_k))}{B_k}\right).$$

In contrast, the baseline method (where $B_k = 1$) has a variance term of order $\mathcal{O}(\gamma_k^2(\sigma_\Phi^2 + \sigma_C^2(\epsilon_k)))$ (see [7, Eq. 20]). As B_k increases adaptively, the noise term in AdaBS-RBSGD vanishes at a faster rate, allowing the algorithm to achieve lower error floors compared to the baseline.

5 Numerical experiments

In this section, we present a comprehensive empirical investigation to validate the theoretical claims of our proposed AdaBS-RBSGD algorithm. The experiments are designed to demonstrate its practical advantages over the baseline single-sample method from Dimitrieski, Cao, and Ebenbauer [7].

[***] In the baseline proof [7], the total error term is $\gamma_k\epsilon_k r_k + 4\gamma_k^2\sigma_\Phi^2$. Our Q_k term modifies the variance component to account for $B_k > 1$, while preserving the structural terms ξ_k and $\hat{L}$ derived from the barrier relaxation.

5.1 Experimental setup

To ensure a fair and reproducible evaluation, the experiments were conducted on the Google Colab platform with the following specifications:

- **CPU:** Intel(R) Xeon(R) CPU @ 2.20GHz
- **RAM:** 13.61 GB
- **GPU:** NVIDIA Tesla T4 (15 GB VRAM)
- **OS:** Linux

Test problem generation: We utilize the large-scale quadratic programming benchmark framework introduced in [7]. The problem is constructed to ensure that the unconstrained minimizer lies outside the feasible region, thereby enforcing active constraints at the optimal solution. The objective function $f(x)$ is defined as:

$$f(x) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{d}\left(\alpha_{i,j}x_j + \log\left(1+e^{-\alpha_{i,j}x_j}\right) + (x_j-\beta)^2\right),$$

where the dimension is set to $d = 50$ and the number of components is $n = 10$. The coefficients $\alpha_{i,j}$ are sampled uniformly from $\mathcal{U}(0.1, 1.1)$. The shift parameter is set to $\beta \approx 2.365$, such that the Euclidean norm of the unconstrained minimizer is approximately $\|x^*_{unc}\|_2 \approx 15$, ensuring it violates the constraints.

The feasible set is defined by $m = 10,000$ affine inequality constraints, generated to form an outer approximation of an ellipsoidal set $\mathcal{E} := \{x \in \mathbb{R}^d : x^\top Qx \leq 100\}$. Here, Q is a positive definite diagonal matrix with entries $q_i \sim \mathcal{U}(1, 1.5)$, and the constraint normal vectors are randomly sampled from the boundary of $\mathcal{E}$. This construction guarantees a compact feasible set satisfying the nonempty interior assumption.

The optimal solution x^* is pre-computed using the L-BFGS-B algorithm on the deterministic barrier objective. Crucially, to ensure statistical robustness, the results presented in this section are averaged over $R = 1000$ **independent simulation trajectories**, providing a significantly more rigorous validation compared to standard evaluations.

5.2 Parameter robustness analysis

To assess the stability of our proposed hybrid algorithm, we conducted a comprehensive grid search over the two key adaptive parameters from the RADADAMP logic [20]: the memory factor ρ (RHO) and the regularization coefficient λ (REG LAMBDA). We evaluated the Mean Final Error $||x_k - x^*||_2$ at $K = 5000$ across a grid of $\rho \in \{0.9, 0.99, 0.999\}$ and $\lambda \in \{10^{-4}, 10^{-3}, 10^{-2}\}$.

Contrarily to findings in previous studies which noted performance degradation at lower memory factors, our implementation demonstrated remarkable robustness. As illustrated in Figure 2, the algorithm maintained a consistently low final error norm (approximately 6.6×10^{-3} to 6.8×10^{-3}) across the entire parameter grid. The uniform distribution of low error values indicates that the adaptive batching mechanism effectively suppresses gradient noise regardless of minor variations in hyperparameter settings.

Based on these results, we proceeded with the conservative settings of $B_0 = 2$, $B_{\max} = 8$, $\rho = 0.999$, and $\lambda = 10^{-3}$ for the main experiments, although our analysis suggests that the algorithm remains stable and precise over a much wider range of configurations.

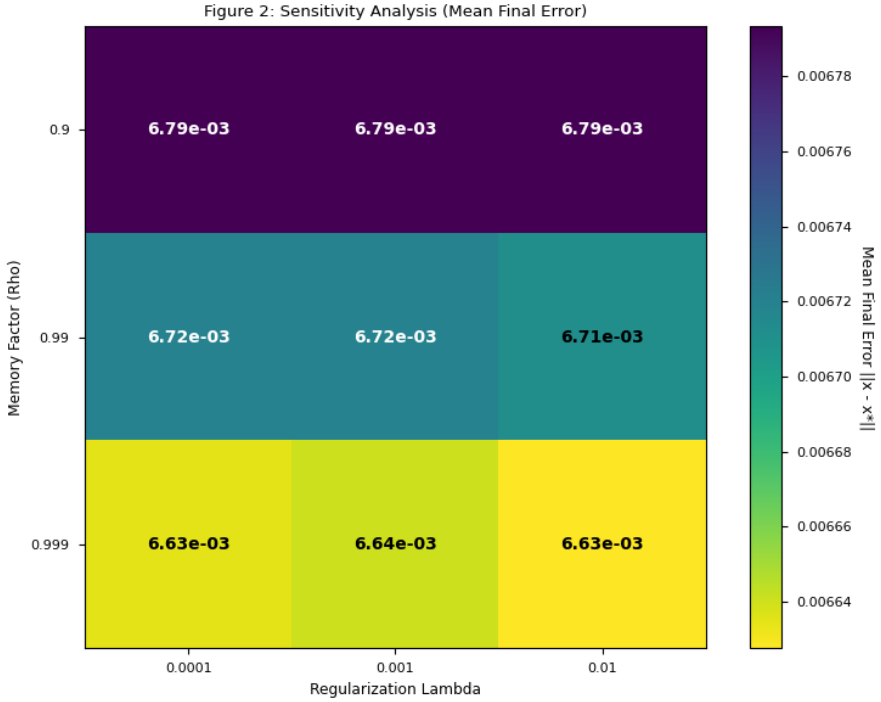

Figure 2: **Robustness Analysis:** Mean Final Error $||x_k - x^*||_2$ at $K = 5000$ for varying memory factors (ρ) and regularization coefficients (λ). The uniform color distribution highlights the stability of the AdaBS-RBSGD algorithm against parameter variations.

5.3 Algorithms for comparison

We compare the performance of two algorithms:

- **Base SGD (Baseline):** The adaptive relaxed barrier SGD algorithm from [7], which corresponds to our Algorithm 1 with a fixed batch size of $B_k = 1$.
- **AdaBS-RBSGD (Ours):** Our proposed hybrid algorithm using the adaptive parameters defined in Section 5.2.

5.4 Performance evaluation

We now evaluate the algorithms based on convergence speed, final solution accuracy, and variance. The results are averaged over $R = 1000$ independent simulation trajectories.

5.4.1 Error vs. iterations

Figure 3 plots the primary performance metric, the Euclidean distance to the optimum $||x_k - x^*||_2$, against the number of iterations (k). The results clearly validate our central hypothesis.

- **Baseline (Red):** The baseline algorithm converges quickly but stagnates around a high-variance "noise floor" (approx. 1.4×10^{-2}). The shaded red area remains large, preventing further progress.
- **AdaBS-RBSGD (Green):** Our hybrid algorithm initially tracks the baseline. However, as it approaches the optimum, the adaptive batching mechanism activates. The algorithm breaks away from the baseline's noise floor, actively suppressing variance. By iteration 10,000, it achieves a final solution that is approximately 64% more accurate ($\approx 4.9 \times 10^{-3}$) than the baseline.

Robustness to Ill-Conditioning: The superior stability observed in Figure 3 also suggests resilience against ill-conditioning. In ill-conditioned landscapes, where high curvature in specific directions causes single-sample SGD to oscillate violently (zig-zagging), the adaptive increase in batch size B_k plays a stabilizing role. By averaging out the stochastic noise components, the algorithm computes a more accurate approximation of the true descent direction. This effectively dampens the erratic oscillations typical of ill-conditioned problems, allowing AdaBS-RBSGD to

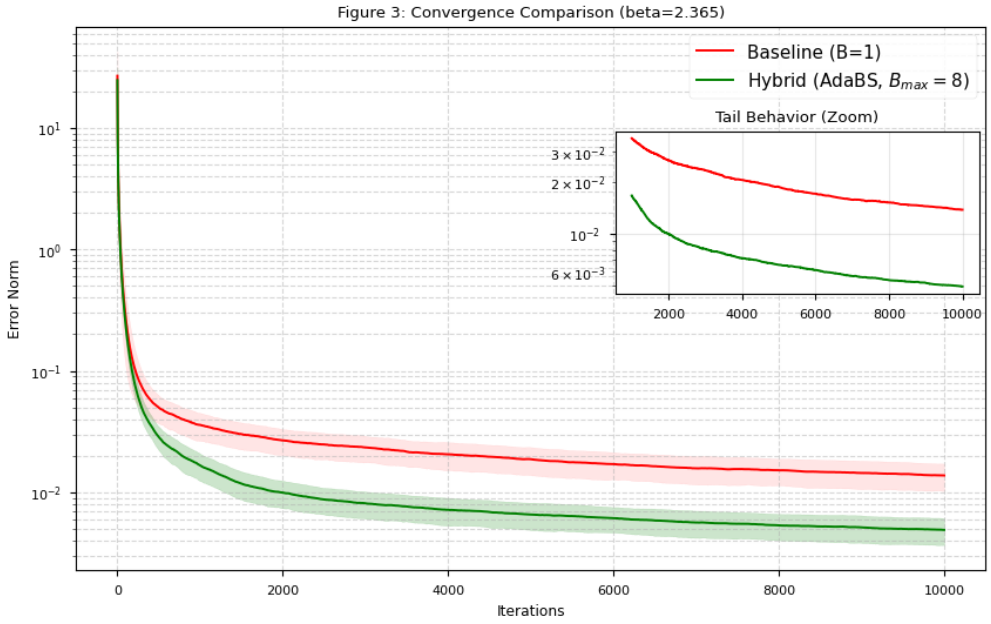

Figure 3: Algorithm Performance Comparison ($n = 10, m = 10000$). The plot shows the mean $||x_k - x^*||_2$ (solid line) and ± 1 standard deviation (shaded area) over $R = 1000$ runs.

maintain steady progress where the baseline stagnates. Regarding nonconvexity, as noted in Remark 1, while the current theory assumes convexity, this variance reduction mechanism is expected to be empirically beneficial in navigating nonconvex landscapes, a subject reserved for future investigation.

5.4.2 Adaptive batching mechanism

Figure 4 explains *how* the hybrid algorithm achieves this variance suppression. It plots the behavior of the mean batch size B_k over the $R = 1000$ trajectories. B_k starts at $B_0 = 2$ allowing fast initial exploration. As the optimization loss decreases, B_k rapidly increases until it hits the cap $B_{\max} = 8$. This transition from a low-batch to a controlled high-batch regime is precisely what actively suppresses the gradient variance and enables continued convergence.

5.4.3 Computational efficiency

Figure 5 provides a critical analysis of computational efficiency by plotting the error against the total number of gradient computations.

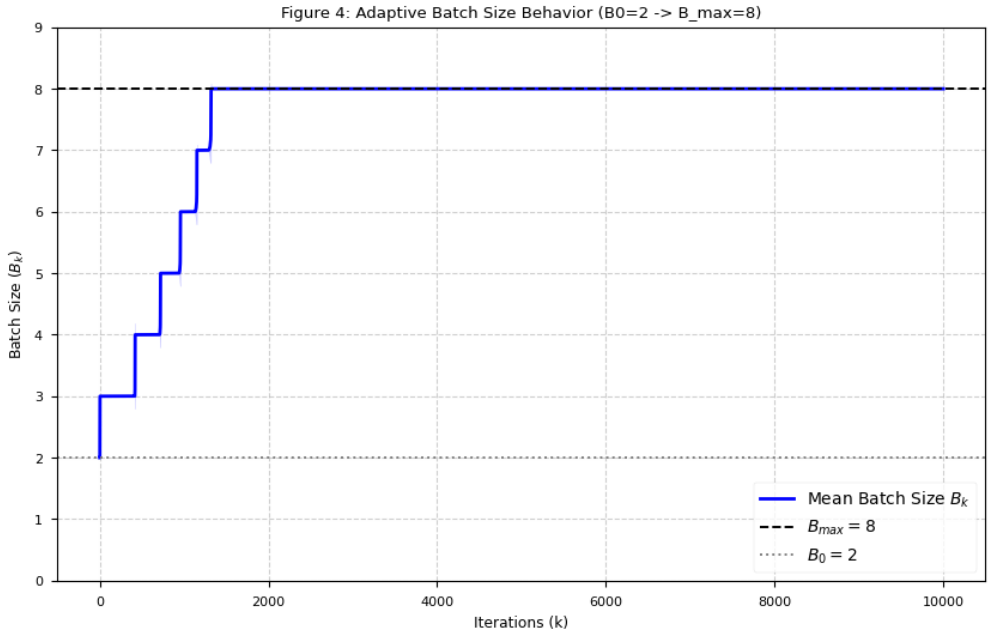

Figure 4: Adaptive batch size behavior (Hybrid algorithm). The plot shows the mean B_k increasing from $B_0 = 2$ to $B_{\max} = 8$ as the algorithm converges.

- **Baseline (Red):** Efficient for low-accuracy solutions but hits a precision ceiling at $\approx 1.4 \times 10^{-2}$, regardless of the computational budget.

- **AdaBS-RBSGD (Green):** While requiring slightly more gradient computations initially, the hybrid method successfully continues to converge past the baseline's ceiling. It trades a marginal increase in computation for a higher precision solution that is impossible for the baseline to achieve.

Scalability with large number of constraints: A key advantage of the adaptive sampling strategy is its scalability. The computational cost per iteration is dominated by the gradient evaluations of the sampled batch, scaling as $\mathcal{O}(B_k \cdot d)$, where d is the dimension of the decision variable. Crucially, this cost is independent of the total number of constraints m. Even as m grows extremely large (high-dimensional constraint space), the batch size remains capped at $B_{\max} \ll m$. Consequently, AdaBS-RBSGD maintains high computational efficiency and is well-suited for large-scale problems where evaluating all constraints is prohibitive.

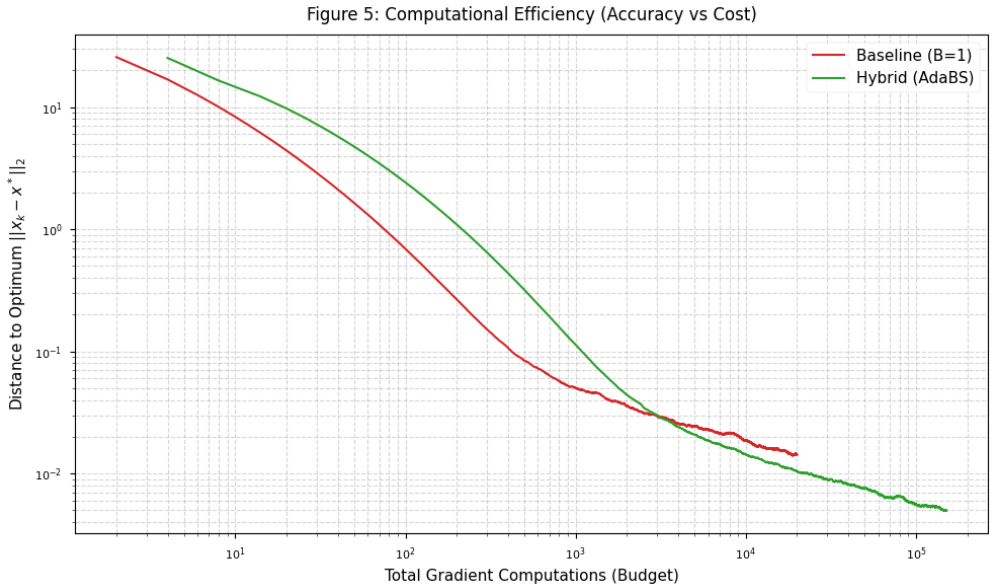

Figure 5: Computational Efficiency. The plot shows the mean $||x_k - x^*||_2$ against the total cumulative gradient computations (log-log scale).

5.4.4 Final solution quality and computational cost

Finally, we analyze the final iterate x_K after $K = 10,000$ iterations. Table 1 summarizes the mean objective value $f(x_K)$, maximum constraint violation, and total wall-clock time.

Table 1: Comparison Results (Mean ± Std. Dev. over $R = 1000$ runs)

Algorithm	**Final Error** $||x - x^*||_2$	**Max Violation**	**Mean Objective**	**Total Time (s)**
Dimitrieski (Baseline)	$1.38 \times 10^{-2} (\pm 3.48 \times 10^{-3})$	0.00(±0.00)	81.7414(±0.0001)	1.79
AdaBS-RBSGD (Ours)	$\mathbf{4.91 \times 10^{-3}} (\pm \mathbf{1.22 \times 10^{-3}})$	0.00(±0.00)	81.7412(±0.0000)	2.53

The table provides two key insights:

1. **Accuracy stability:** Both algorithms find feasible solutions. However, the hybrid algorithm achieves a significantly lower final error norm. More importantly, the standard deviation of our method ($\pm 1.22 \times 10^{-3}$) is approximately a factor of 3 lower than the baseline ($\pm 3.48 \times 10^{-3}$), indicating superior algorithmic stability.

2. **Computational Trade-offs:** The transition from a fixed single-sample batch to an adaptive batch size introduces a fundamental trade-off. By dynamically increasing B_k, the per-iteration computational cost rises (since B_k gradient evaluations are required instead of one). This explains why the hybrid algorithm's total wall-clock time (2.53s)

is higher than the baseline (1.79s). However, this additional computational investment is strictly necessary to suppress the stochastic noise that limits the baseline's precision. Effectively, we trade a controlled increase in runtime for a significant gain in solution accuracy and stability. Practically, this trade-off is kept efficient by the cap $B_{\max} = 8$, ensuring the algorithm remains computationally feasible while achieving a precision level that the fixed-batch baseline cannot reach.

5.5 Sensitivity to batch capacity limit ($B_{\max}$)

To rigorously justify our choice of the batch capacity limit $B_{\max} = 8$, we conducted an ablation study analyzing the trade-off between solution accuracy and computational cost. We evaluated the hybrid algorithm across a range of capacity limits $B_{\max} \in \{2, 4, 8, 16, 32, 64\}$, averaging the results over independent trials to ensure statistical significance.

The results, illustrated in Figure 6, reveal a distinct trade-off point at $B_{\max} = 8$.

- **Significant accuracy gain:** Increasing the limit from $B_{\max} = 2$ to 8 results in a sharp reduction in the final error norm $||x_K - x^*||$ (from $\approx 9.5 \times 10^{-3}$ to $\approx 4.9 \times 10^{-3}$), effectively halving the error with only a minimal increase in execution time.

- **Computational stability regime:** For $B_{\max} \leq 8$, the wall-clock time (dashed red line) remains remarkably stable, showing only a slight linear increase. This indicates that the adaptive batching is efficiently utilizing the available computational resources without overhead.

- **Cost-accuracy Trade-off:** Beyond $B_{\max} = 8$, while the error continues to decrease, the computational cost exhibits a steep, super-linear increase. For instance, achieving the marginal accuracy gain at $B_{\max} = 64$ requires nearly doubling the execution time compared to $B_{\max} = 8$.

Consequently, $B_{\max} = 8$ represents the optimal balance on the Pareto frontier, maximizing solution precision while maintaining the computational efficiency that characterizes our proposed method.

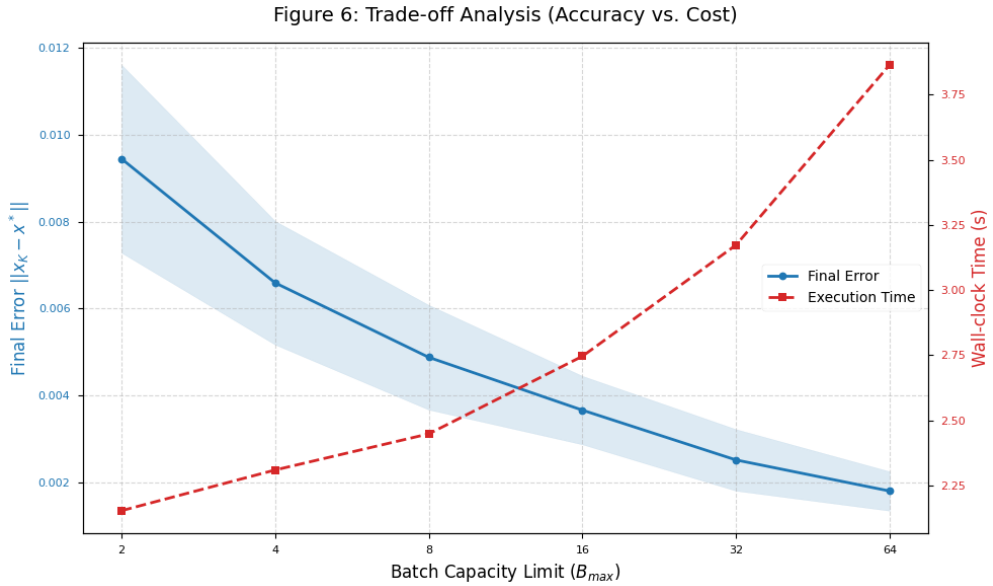

Figure 6: Trade-off Analysis: Final error (solid blue) vs. Wall-clock time (dashed red) for varying batch capacity limits ($B_{\max}$). The plot demonstrates that $B_{\max} = 8$ offers the optimal balance, avoiding the steep computational penalty observed at higher capacities.

6 Conclusion

In this paper, we introduced AdaBS-RBSGD, a novel and efficient hybrid algorithm for solving stochastic constrained optimization problems. Our work builds upon the modern framework established by Dimitrieski, Cao, and Ebenbauer [7], addressing its critical limitation of high gradient variance which leads to significant oscillation near the optimum. To overcome this, we integrated the adaptive batching strategy (RADADAMP) from Sievert-Shah [20]. Our hybrid algorithm dynamically adjusts the batch size B_k to be inversely proportional to the optimization loss, actively suppressing gradient variance as the iterates approach the solution.

Our primary contribution is twofold. First, we provided a rigorous theoretical analysis establishing the almost-sure convergence of this nontrivial hybrid algorithm (Theorem 1). Second, we presented comprehensive numerical experiments that empirically validate our approach. Our results clearly demonstrate that AdaBS-RBSGD successfully breaks away from the high-variance 'noise floor' that stalls the baseline method, allowing our algorithm to achieve a significantly more accurate and stable final solution.

This superior accuracy involves a manageable computational trade-off. As our experiments with the optimized batch cap ($B_{\max} = 8$) confirm, achieving this high-precision solution requires only a moderate increase in the computational budget (approx. 1.4× time increase) compared

to the baseline method. This work thus presents a practical and theoretically-sound method for accelerating stochastic constrained optimization, offering a valuable trade-off for applications where solution precision and stability are paramount.

Future work

The promising results of our AdaBS-RBSGD algorithm open several interesting avenues for future research:

- **Analysis of convergence rate:** Our current analysis proves almost-sure convergence. A valuable theoretical extension would be to derive the explicit convergence rate of the hybrid AdaBS-RBSGD algorithm, similar to analyses found in [13], formally proving that it achieves a faster rate than the baseline, as suggested by our variance analysis.

- **Interaction with step-size:** This work used a pre-defined decaying step-size γ_k. Future work should investigate the synergy between the adaptive batch size B_k and adaptive learning rates (e.g., AdaGrad-style scaling), as both mechanisms respond to the optimization landscape.

- **Nonaffine constraints:** The current analysis, similar to the baseline work [7], is focused on problems with affine (linear) constraints. Investigating the performance, stability, and theoretical guarantees of the AdaBS-RBSGD approach when applied to problems with general nonlinear constraints would be an important next step.

- **Nonuniform sampling:** Our algorithm currently employs standard uniform sampling for constraints. Future work could explore the synergy between adaptive batching and nonuniform or importance sampling techniques. For instance, adaptively sampling constraints that are more frequently violated might accelerate convergence.

- **Extension to non convex objectives:** While our current theoretical framework relies on the assumption of strong convexity to guarantee almost-sure convergence (Assumption 1), stochastic gradient methods are widely used in nonconvex settings (e.g., neural network training). Investigating the empirical performance and establishing local convergence guarantees for AdaBS-RBSGD in nonconvex optimization landscapes represents a significant and promising direction for future research.

Funding

This research received no external funding.

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] Balles, L., Romero, J. and Hennig, P. *Coupling adaptive batch sizes with learning rates*, In 33rd Conf. Uncertainty in Artif. Intell. (UAI), 675–684, 2017.

[2] Beiser, F., Keith, B., Urbainczyk, S. and Wohlmuth, B. *Adaptive sampling strategies for risk-averse stochastic optimization with constraints,* IMA J. Numer. Anal. 43(6), (2023) 3729–3765.

[3] Bollapragada, R., Byrd, R. and Nocedal, J. *Adaptive sampling strategies for stochastic optimization,* SIAM J. Optim. 28(4), (2018) 3312–3343.

[4] Bottou, L., Curtis, F.E. and Nocedal, J. *Optimization methods for large-scale machine learning,* SIAM Rev. 60(2) (2018), 223–311.

[5] Boyd, S. and Vandenberghe, L. Convex optimization, Cambridge University Press, 2004.

[6] Bubeck, S. *Convex optimization: Algorithms and complexity,* Found. Trends Mach. Learn. 8(3-4) (2015), 231–358.

[7] Dimitrieski, N., Cao, J. and Ebenbauer, C. *Stochastic gradient descent for constrained optimization based on adaptive relaxed barrier functions*, IEEE Control Syst. Lett. 9 (2025) 829–834.

[8] Feller, C. and Ebenbauer, C. *A stabilizing iteration scheme for model predictive control based on relaxed barrier functions,* Automatica 80 (2017), 328–339.

[9] Fiacco, A.V. and McCormick, G.P. *Nonlinear programming: Sequential unconstrained minimization techniques*, John Wiley & Sons, 1968.

[10] Garrigos, G. and Gower, R.M. *Handbook of convergence theorems for (stochastic) gradient methods,* arXiv preprint arXiv:2301.11235 (2023).

[11] Karandikar, R.L. and Vidyasagar, M. *Convergence rates for stochastic approximation: biased noise with unbounded variance, and applications*, J. Optim. Theory Appl. 203(3) (2024) 2412–2450.

[12] Li, M., Grigas, P. and Atamtürk, A. *New penalized stochastic gradient methods for linearly constrained strongly convex optimization,* J. Optim. Theory Appl. 205(2) (2025) 29.

[13] Liu, J. and Yuan, Y. *On almost sure convergence rates of stochastic gradient methods,* In Proc. 35th Annu. Conf. Learn. Theory (COLT), PMLR, 178, 1–21, 2022.

[14] Nocedal, J. and Wright, S.J. *Numerical optimization*, 2nd ed., Springer, 2006.

[15] Polyak, B.T. and Juditsky, A.B. *Acceleration of stochastic approximation by averaging,* SIAM J. Control Optim. 30(4) (1992), 838–855.

[16] Robbins, H. and Monro, S. *A stochastic approximation method,* Ann. Math. Stat. 22(3) (1951), 400-407.

[17] Robbins, H. and Siegmund, D. *A convergence theorem for nonnegative almost supermartingales and some applications*, In Optimizing Methods in Statistics, Academic Press, 233–257, 1971.

[18] Rockafellar, R.T. *Convex analysis*, Princeton University Press, 1970.

[19] Roy, S.K. and Harandi, M. *Constrained stochastic gradient descent: The good practice,* In Int. Conf. Digit. Image Comput. Tech. Appl. (DICTA), IEEE, 1–8, 2017.

[20] Sievert, S. and Shah, S. *Improving the convergence of SGD through adaptive batch sizes,* arXiv preprint arXiv:1910.08222 (2019).

[21] Smith, S.L., Kindermans, P.J. and Le, Q.V. *Don't decay the learning rate, increase the batch size*, In Int. Conf. Learn. Represent. (ICLR), 2018.

[22] Wang, M. and Bertsekas, D.P. *Incremental constraint projection methods for variational inequalities,* Math. Program. 150 (2015), 321–363.

[23] Wang, X., Ma, S. and Yuan, Y. *Penalty methods with stochastic approximation for stochastic nonlinear programming,* Math. Comput. 86(306) (2017) 1793–1820.

[24] Yan, Y. and Xu, Y. *Adaptive primal-dual stochastic gradient method for expectation-constrained convex stochastic programs,* Math. Program. Comput. 14(2) (2022), 319-363.

[25] Zhang, T. *Solving large scale linear prediction problems using stochastic gradient descent algorithms,* In Proc. 21st Int. Conf. Mach. Learn. (ICML), 116, 2004.