WHY DOES ADAPTIVE BATCHING HELP LLM PRETRAINING? A PERSPECTIVE FROM UNBOUNDED VARIANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Increasing the batch size during training is a common practice in large language model (LLM) pretraining, yet the theoretical justification behind its success is not well understood. Analyses of stochastic optimization often assume uniformly bounded stochastic gradient variance, yet recent evidence suggests that this assumption fails in many practical nonconvex problems. The Blum–Gladyshev (BG-0) noise model relaxes this assumption by allowing the variance to grow quadratically with the distance from initialization, suggesting that batch size schedulers can help by controlling the variance growth during training. However, this growth can be overly conservative in practice. We empirically investigate variance growth in LLM pretraining and observe that a generalized BG model with a tunable growth exponent provides a tighter description of practical noise behavior. Motivated by this observation, we introduce the generalized BG-a noise model, which interpolates between bounded variance ($a = 0$) and BG-0 noise ($a = 2$). Under L-smoothness, we derive an information-theoretic lower bound with growth-dependent oracle complexity $\Omega(\epsilon^{-(4+a)})$ and establish a matching upper bound in ϵ-dependence by increasing the batch size as the iterates move away from initialization. Finally, we propose an adaptive batch scheduler that controls variance growth through dynamic batch size adjustments during training. In pretraining OLMo2 models of up to 1B parameters on C4, our scheduler achieves a lower validation loss than both small and large batch training under matched token budgets, while using less than 10% of the iterations of small batch training.

1 INTRODUCTION

With the growing scale of training datasets, particularly in language model pretraining, large batch training is increasingly important for exploiting data parallelism. However, using a very large batch from initialization can reduce token efficiency, motivating schedules that begin with smaller batches and increase them as training progresses (Merrill et al., 2026). Such schedules are already used in large scale training, e.g., GPT-3 gradually increased its batch size during the initial phase of pretraining (Brown et al., 2020). Existing adaptive batch schedulers commonly guide batch selection through gradient noise scale (GNS) estimates (McCandlish et al., 2018; Naganuma et al., 2026). However, Merrill et al. (2026) show that this connection relies on assumptions about the optimizer and curvature that need not hold in deep learning, and find that GNS can be an unreliable proxy in practice. They instead estimate the critical batch size directly through additional branched training runs at different training stages. Despite the success of these schedules in practice, the theoretical justification for why increasing the batch size during training helps LLM pretraining is not well understood. In this work, we study this question from the perspective of stochastic gradient noise, where the variance grows as the iterates move away from initialization.

Classical optimization theory does not directly account for this behavior, as the convergence analysis of stochastic gradient methods commonly combines L-smoothness of the objective with a uniformly bounded variance assumption on the stochastic gradient oracle (Bach & Moulines, 2011; Shamir & Zhang, 2013; Ghadimi & Lan, 2013). For an unbiased estimator $g(x, \xi)$ of $\nabla f(x)$, this assumption requires $\mathbb{E}_\xi \|g(x, \xi) - \nabla f(x)\|^2 \leq \sigma^2$ for every x, where σ is constant and independent of the iteration. However, this assumption is often unrealistic because smoothness directly controls the changes in the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

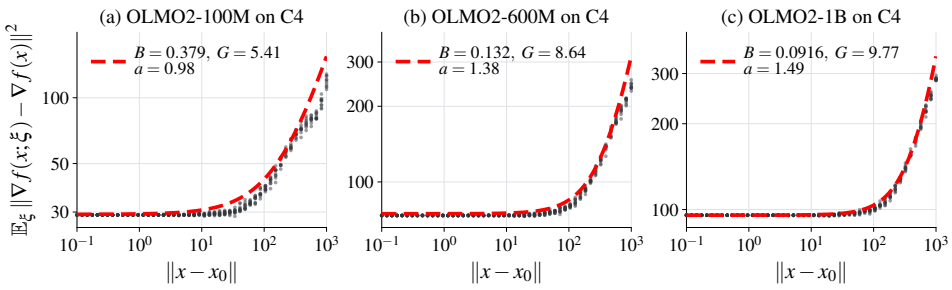


Figure 1: Empirical stochastic-gradient noise of OLMo2 models on C4 as a function of displacement from initialization. For each radius r, we evaluate $x = x_0 + ru$ along ten independently sampled, normalized Gaussian directions and plot the corresponding minibatch variance estimates. The dashed red curve in each panel is the fitted BG-a model, $(B^2 \|x - x_0\|^a + G^2)$. The fitted parameters are: **(a)** OLMo2-100M: $B = 0.379, G = 5.41, a = 0.98$; **(b)** OLMo2-600M: $B = 0.132, G = 8.64, a = 1.38$; and **(c)** OLMo2-1B: $B = 0.0916, G = 9.77, a = 1.49$. Across all three model sizes, the fitted curve captures both the initial noise floor governed by (G^2) and the distance-dependent growth governed by $B^2 \|x - x_0\|^a$, with the fitted exponent a increasing with model size. The corresponding results for the vision models are provided in Appendix E.2.1.

population gradient, not the variance of its stochastic estimates. As a result, the bounded variance assumption need not hold even for unconstrained least squares problems (Alacaoglu et al., 2025; Fazla et al., 2026; Upadhyay et al., 2026), where differences between sample gradients and the population gradient can grow linearly with the parameters, causing variance to grow quadratically. As a remedy, researchers have analyzed the Blum–Gladyshev (BG-0) noise model (Blum, 1954; Gladyshev, 1965) which accommodates this behavior by allowing $\mathbb{E}_\xi \|g(x,\xi) - \nabla f(x)\|^2 \le B^2 \|x - x_0\|^2 + G^2$, where x_0 is a fixed reference point, and B, G are problem dependent constants. For smooth nonconvex optimization under this unbounded variance oracle, Fazla et al. (2026) establish an $\Omega(\epsilon^{-6})$ stochastic first-order oracle lower bound for finding an ϵ-stationary point and provide matching upper bounds for stochastic gradient descent (SGD) with dynamic batching. Their results show that distance-dependent noise incurs an intrinsic complexity penalty relative to the classical bounded variance rate of $\Theta(\epsilon^{-4})$ (Arjevani et al., 2023).

Although BG-0 relaxes the bounded variance assumption, its quadratic dependence on distance can overestimate the variance growth encountered in practice. As illustrated in Figure 1, we find that the empirical variance growth in LLM pretraining is subquadratic, with fitted exponents $a \in [0.98, 1.49]$ that increase with model size, and we observe a similar behavior for the vision models we study. Motivated by these observations, we introduce the generalized BG-a noise model, $\mathbb{E}_\xi \|g(x,\xi) - \nabla f(x)\|^2 \le B^2 \|x - x_0\|^a + G^2$, which interpolates between uniformly bounded variance at $a = 0$ and classical quadratic BG noise ($a = 2$, denoted as BG-0 in prior works (Fazla et al., 2026; Upadhyay et al., 2026)). For L-smooth nonconvex objectives, we establish matching lower and upper bounds with $\Theta(\epsilon^{-(4+a)})$ dependence for finding an ϵ-stationary point for $a \in [0, 2]$. Thus, the rate at which stochastic gradient noise grows with distance directly governs the oracle complexity, while recovering the bounded variance and BG-0 rates as the two endpoints.

One direct way to control distance-dependent noise is to increase the batch size as the iterates move far from initialization. Accordingly, Fazla et al. (2026) use dynamic batching to control the minibatch variance under BG-0 noise, which we extend to the BG-a model, where the corresponding batch size requirement scales with $B^2 \|x_t - x_0\|^a + G^2$. We show that this dynamic batching strategy under the BG-a model yields a matching $\mathcal{O}(\epsilon^{-(4+a)})$ oracle complexity for SGD. Thus, we provide a theoretical explanation for increasing batch size during training: small batches are sufficient near initialization, where noise is small, while larger batches are needed to control variance growth as iterates move away from initialization. However, directly scaling batch sizes with noise level can lead to large per-iteration batch requirements as displacement grows. To alleviate this issue and make the method more flexible and practical, we extend the dynamically batched SGD analysis to SGD with momentum (SGDM) under BG-a noise. Our analysis shows that momentum provides an additional form of averaging over time, reducing the sufficient per-iteration batch size at the cost of additional

iterations while preserving the optimal $\mathcal{O}(\epsilon^{-(4+a)})$ oracle complexity. This establishes a theoretical trade-off between batch size and iteration count under distance-dependent noise.

Together, these results offer a theoretical perspective on why increasing the batch size during pretraining works, complementary to empirical approaches based on GNS and critical batch size estimation. In particular, our BG-a analysis connects the growth of the batch size to the growth of stochastic gradient noise, prescribing batch sizes that control variance growth as the iterates move away from initialization. To examine this perspective in practice, we translate the resulting variance control condition into an adaptive batch scheduler. In our LLM pretraining experiments, our scheduler achieves a lower validation loss than both small and large batch training under matched token budgets. We also provide experiments on image classification tasks, and discuss the role of adaptive batching in offline, multi-epoch training in Appendix E.2.

We summarize our contributions as follows:

- We empirically study the stochastic gradient noise in LLM pretraining and demonstrate that its variance grows subquadratically with the distance from initialization. We formalize this phenomenon by introducing the BG-a oracle, which interpolates between bounded variance and BG-0 noise.
- For L-smooth nonconvex optimization, we establish an $\Omega(\epsilon^{-(4+a)})$ lower bound on the stochastic first-order oracle complexity for $a \in [0, 2]$, characterizing how distance-dependent noise growth degrades the sample complexity of finding an ϵ-stationary point.
- We establish matching $\mathcal{O}(\epsilon^{-(4+a)})$ oracle complexity bounds for both SGD and SGDM under dynamic batching schedules. We identify momentum as a mechanism to reduce the required batch size at the expense of additional iterations, all while maintaining the optimal dependence on ϵ.
- We instantiate our theoretical variance control analysis into a practical adaptive batch scheduler that scales batch sizes according to displacement from initialization and can be readily used in practice. In OLMo2 pretraining on C4, our scheduler achieves a lower validation loss than both small and large batch training under matched token budgets, while using less than 10% of the iterations of small batch training.

2 Gradient Noise Grows with Distance in LLM Pretraining

Before developing our theoretical results, we first examine the behavior of stochastic gradient noise in LLM pretraining. Specifically, we study how the gradient variance grows as the model parameters move away from initialization and assess how tightly the quadratic distance dependence assumed by the BG-0 oracle captures this growth. We then investigate whether allowing a tunable growth exponent yields a tighter characterization of the observed variance.

We study language modeling with OLMo2-100M, OLMo2-600M, and OLMo2-1B models (Team OLMo et al., 2024) on C4 (Raffel et al., 2020). To study the effect of displacement on noise growth without following an optimizer generated trajectory, we evaluate the model at randomly perturbed parameter states, without performing training. Starting from a randomly initialized anchor point x_0, we construct evaluation points as

$$x = x_0 + ru, \qquad u = \frac{Dz}{\|Dz\|}, \qquad z \sim \mathcal{N}(0, I),$$

where $r > 0$ is the distance from initialization and D is a diagonal matrix specifying the relative perturbation scales. We set the diagonal entries of D to 2 for parameters of the LM head and 1 for all remaining parameters. Thus, we double the Gaussian perturbation scale of the LM head before globally normalizing the direction, preserving $\|x - x_0\| = r$. For each model, we consider 50 logarithmically spaced radii and sample 10 independent directions at each radius.

This modeling choice is motivated by our observation that the LM head exhibits the largest weight changes among all parameter groups in our pretraining runs. This observation agrees with prior work that identifies a gradient bottleneck at the LM head (Godey & Artzi, 2026) and highlights heterogeneous output layer gradients and, hence, larger effective learning rates for the LM head weights (Glentis et al., 2026). We use a factor of two as a simple way to reflect this behavior, so that the perturbation directions more closely resemble how the parameters move during training.

At each evaluation point x, we estimate the variance of stochastic gradients around a fixed large batch reference gradient $\nabla \tilde{f}(x)$, which is computed before the evaluation process and used as

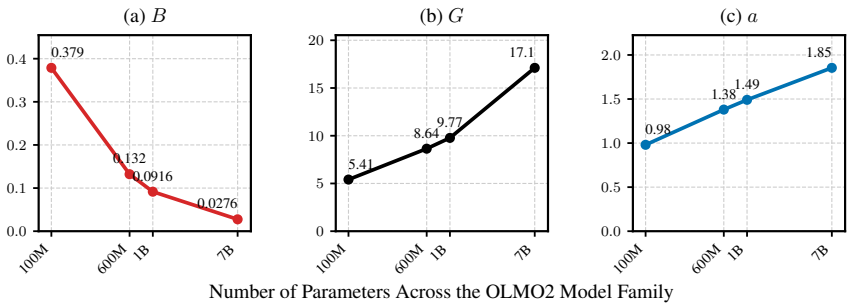


Figure 2: Estimated BG-a noise-model parameters across OLMo2 model sizes. We observe that as model size increases, (a) B tends to decrease, while (b) G and (c) a tend to increase.

an approximation to the population gradient. Specifically, we approximate the stochastic gradient variance $\mathbb{E}_\xi \|\nabla f(x;\xi) - \nabla f(x)\|^2$ using $\frac{1}{M}\sum_{i=1}^{M} \left\| g_b^{(i)}(x) - \nabla \tilde{f}(x) \right\|^2$, where $g_b^{(i)}(x) = \frac{1}{b}\sum_{j=1}^{b} \nabla f(x;\xi_{i,j})$. For all OLMo2 models, we use a reference batch of 512 sequences and $M = 32$ stochastic batches, each containing 4 sequences, where sequence length is set as 2048.

Figure 1 shows the distribution of individual variance measurements across the three OLMo2 models, together with the best fitting curves of the form $B^2\|x - x_0\|^a + G^2$, subject to $B, G \geq 0$ and $a \in [0, 2]$. The fitted exponents are approximately $a = 0.98$, 1.38, and 1.49 for OLMo2-100M, OLMo2-600M, and OLMo2-1B, respectively. These results indicate that the distance from initialization is a useful predictor of gradient noise growth. Across the sampled points, the proposed BG-a condition closely captures the observed increase in variance as the iterates move farther from initialization. Moreover, allowing the exponent a to adapt to the observed noise-growth behavior provides a better fit than the BG-0 condition of Alacaoglu et al. (2025); Fazla et al. (2026); Upadhyay et al. (2026), which fixes the corresponding exponent to 2. Figure 2 further shows how the fitted parameters change with model size: as the model grows, the growth exponent a and the noise floor G increase, while B decreases. We observe a similar subquadratic noise growth for ResNet50 on image classification tasks, which we report in Appendix E.2.1.

These observations motivate the BG-a noise model, which we formalize and analyze in the next section, and an adaptive batch size scheduler that increases the batch size as the iterates move farther from initialization, thereby controlling variance growth throughout training.

3 Theoretical Analysis under the BG-a Oracle

3.1 Preliminaries

For a given random variable X, we write $\mathbb{E}[X]$ to denote the expectation with respect to all randomness introduced by the algorithm. $\|\cdot\|$ denotes the Euclidean norm, and $\langle\cdot,\cdot\rangle$ denotes the Euclidean inner product. The notation $\mathcal{O}(\cdot)$ suppresses numerical constants that are independent of the target accuracy ϵ. We use $\mathcal{F}_t$ to denote the history of the algorithm up to iteration t, which contains the initialization and all samples used at iterations $0, \ldots, t-1$. We denote the corresponding conditional expectation by $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot \mid \mathcal{F}_t]$. In assumptions involving an oracle sample, $\mathbb{E}_\xi$ denotes expectation only with respect to the sample ξ, with the query points held fixed.

We consider the stochastic optimization problem $\min_{x\in\mathbb{R}^d} f(x),\ f(x) := \mathbb{E}_\xi\big[f(x;\xi)\big]$. Here, $f : \mathbb{R}^d \to \mathbb{R}$ is differentiable, nonconvex, and bounded below by $f^{\inf} := \inf_x f(x) > -\infty$. We denote the initialization query point with x_0 and use $\Delta = f(x_0) - f^{\inf} \geq 0$.

The convergence criterion that we consider is the average expected squared gradient norm along the iterates $x_0, \ldots, x_{T-1}$,

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\,\|\nabla f(x_t)\|^2 \leq \epsilon^2. \tag{1}$$

Equivalently, returning x_τ with τ drawn uniformly from $\{0, \dots, T-1\}$ yields $\mathbb{E}\|\nabla f(x_\tau)\|^2 \le \epsilon^2$. Given a query point x, the stochastic first-order oracle (SFO) returns a stochastic gradient $\nabla f(x;\xi)$. We measure the computational complexity by the total number of SFO calls. At iteration t, the algorithms we consider query the oracle b_t times at x_t and form the minibatch gradient

$$g_t = \frac{1}{b_t}\sum_{i=1}^{b_t} \nabla f(x_t;\xi_{t,i}), \tag{2}$$

where the batch size b_t is chosen based on $\mathcal{F}_t$ before sampling, and the samples $\xi_{t,1}, \dots, \xi_{t,b_t}$ are drawn independently of each other and of $\mathcal{F}_t$. Hence, iteration t uses b_t SFO calls. We next state the assumptions used in our work.

3.2 Assumptions

Assumption 1 (L-smoothness of f)**.** *The objective function f is globally L-smooth with $L > 0$ such that:*

$$\|\nabla f(x) - \nabla f(y)\| \le L\|x - y\| \qquad \text{for all } x, y \in \mathbb{R}^d.$$

Assumption 2 (Unbiased BG-a stochastic oracle)**.** *Given $a \ge 0$, for every $x \in \mathbb{R}^d$, the stochastic gradient oracle is unbiased, and there exist constants $B, G \ge 0$ satisfying:*

$$\mathbb{E}_\xi\big[\nabla f(x;\xi)\big] = \nabla f(x), \qquad \mathbb{E}_\xi \left\|\nabla f(x;\xi) - \nabla f(x)\right\|^2 \le B^2 \left\|x - x_0\right\|^a + G^2.$$

3.3 Convergence Results

We now summarize our three main theoretical results. Complete statements, constants, and proofs are given in the Appendix.

Theorem 1 (Lower bound under BG-a noise)**.** *Fix any finite $a \ge 0$. Under Assumption 1, for all $\epsilon > 0$ satisfying $\epsilon^2 \le c\,L\Delta$, where $c > 0$ is an absolute constant specified in Appendix B, every algorithm that returns a point $\widehat{x}$ satisfying $\mathbb{E}\left\|\nabla f(\widehat{x})\right\|^2 \le \epsilon^2$ for every unbiased BG-a oracle must incur worst-case expected SFO complexity*

$$\Omega_a\left(\frac{L\Delta}{\epsilon^2} + \frac{G^2 L\Delta}{\epsilon^4} + \frac{B^2 L\Delta^{a+1}}{\epsilon^{4+a}}\right). \tag{3}$$

For Theorem 1, the oracle may be any unbiased vector estimator satisfying the BG-a variance bound. See Appendix B for the complete statement and proof.

Theorem 2 (SGD Upper Bound with Dynamic Batching)**.** *Suppose Assumptions 1 and 2 hold with $0 \le a \le 2$. Fix $\sigma^2 > 0$ and let $0 < \epsilon^2 \le \min\{2\sigma^2, L\Delta\}$. At iteration t, use*

$$b_t = \max\left\{1, \left\lceil \frac{B^2\left\|x_t - x_0\right\|^a + G^2}{\sigma^2} \right\rceil\right\}, \qquad \eta = \frac{\epsilon^2}{2L\sigma^2}, \qquad T = \left\lceil \frac{4\Delta}{\eta\epsilon^2} \right\rceil, \tag{4}$$

and update the iterates using $x_{t+1} = x_t - \eta g_t$, where g_t is the minibatch gradient in equation 2. Then equation 1 holds with the expected SFO cost given as

$$\mathbb{E}\left[\sum_{t=0}^{T-1} b_t\right] = \mathcal{O}\Big(L\Delta(\sigma^2 + G^2)\epsilon^{-4} + B^2 L\Delta^{a+1}\epsilon^{-(4+a)}\Big). \tag{5}$$

In particular, for fixed σ^2, the expected SFO complexity is $\mathcal{O}(\epsilon^{-(4+a)})$. The complete theorem statement and proof are provided in Appendix C.

Theorem 3 (SGDM Upper Bound with Dynamic Batching)**.** *Suppose Assumptions 1 and 2 hold with $0 \le a \le 2$. Fix $\beta \in [0, 1)$ and $\sigma^2 > 0$, and let $c_\beta = \frac{1-\beta}{1+\beta}$ and $0 < \epsilon^2 \le \min\{8\sigma^2(1-\beta^2), L\Delta\}$. At iteration t, use*

$$\begin{aligned} b_t &= \max\left\{1, \left\lceil \frac{c_\beta(B^2\left\|x_t - x_0\right\|^a + G^2)}{\sigma^2} \right\rceil\right\}, \\ \eta &= \frac{c_\beta\epsilon^2}{16L\sigma^2}, \qquad T = \left\lceil \frac{16\Delta}{\eta\epsilon^2} \right\rceil, \end{aligned} \tag{6}$$

and, with g_t as in equation 2, update the iterates using

$$m_{-1} = 0, \qquad m_t = \beta m_{t-1} + (1-\beta) g_t, \qquad x_{t+1} = x_t - \eta m_t. \tag{7}$$

Then equation 1 holds, and

$$\mathbb{E}\left[\sum_{t=0}^{T-1} b_t\right] = \mathcal{O}\left(L\Delta\left(\frac{\sigma^2}{c_\beta} + G^2\right)\epsilon^{-4} + B^2 L \Delta^{a+1} \epsilon^{-(4+a)}\right). \tag{8}$$

Thus, for fixed β and σ^2, SGDM also uses $\mathcal{O}(\epsilon^{-(4+a)})$ expected SFO calls. The complete theorem statement and proof are provided in Appendix D.

When $B > 0$, the upper bounds match the growth-dominated lower-bound exponent: ϵ^{-4} at $a = 0$, $\epsilon^{-(4+a)}$ for $0 < a < 2$, and ϵ^{-6} at $a = 2$. When $B = 0$ and $G > 0$, both stochastic rates reduce to ϵ^{-4}. We emphasize that the restriction $a \leq 2$ in the upper bounds is needed to control the a-th displacement moment using the second moment supplied by the variance assumption. We discuss the obstruction that occurs when $a > 2$ in Appendix C.

3.4 Implications for Batch Size Scheduling

Theorems 2 and 3 prescribe a batch size that tracks the BG-a variance bound at the current iterate. For $a > 0$, since x_0 is the initialization, the prescribed batch size starts at $b_0 = \max\{1, \lceil c_\beta G^2/\sigma^2 \rceil\}$, with $c_\beta = 1$ for SGD, and increases only as the iterates move away from initialization, in proportion to $B^2 \|x_t - x_0\|^a$. When $a = 0$, with the convention $\|x - x_0\|^0 = 1$, the variance is uniformly bounded and the prescribed batch size remains constant. In our proofs, the expected squared displacement $\mathbb{E}\|x_t - x_0\|^2$ is bounded by a quantity that increases with t, so the prescribed batch size is small at the start of training and can grow as training progresses. This avoids the drawbacks of the two fixed batch alternatives: a fixed small batch does not control the variance growth as the iterates move away from initialization, which requires a smaller step size and more iterations, whereas a fixed large batch chosen for the variance level late in training spends unnecessary samples early in training, where the noise is small. Moreover, the batch size rule depends on the noise only through the parameters (B, G, a), which we estimate empirically in Section 2, and serves as the basis of our adaptive batch scheduler in Section 4. Finally, momentum reduces the prescribed batch size by a factor of c_β at the same query point, at the cost of a $1/c_\beta$ times larger iteration count, providing a trade-off between batch size and iteration count while preserving the $\mathcal{O}(\epsilon^{-(4+a)})$ oracle complexity.

4 Adaptive Batch Size Scheduler

We introduce an adaptive batch size scheduler that dynamically adjusts the batch size throughout training to account for the BG-a noise growth. Motivated by the empirical evidence in Section 2 and the theoretical analysis in Section 3, our scheduler determines the batch size at each iteration based on the displacement of the current iterate from the initialization x_0. We first describe the batch size rule and then present the optimization scheme built on top of it.

Under the BG-a noise model, the gradient variance at iteration t is bounded by $B^2\|x_t - x_0\|^a + G^2$, where the exponent a governs how the variance grows with the distance from initialization, while B and G control the distance-dependent and constant noise components, respectively. Our scheduler takes these noise model parameters as inputs, which we estimate using the noise measurements in Section 2, and determines the batch size before sampling the stochastic gradients at each iteration. Following Theorem 3, given a target noise scale $\sigma^2 > 0$ and a momentum parameter $\beta \in [0, 1)$ with $c_\beta = (1-\beta)/(1+\beta)$, we define the distance-dependent function

$$v_t := \frac{c_\beta\left(B^2\|x_t - x_0\|^a + G^2\right)}{\sigma^2}. \tag{9}$$

For simplicity, throughout this section we assume $B, G > 0$ and $a \in (0, 2]$. Hence, $v_0 = c_\beta G^2/\sigma^2 > 0$ and $v_t \geq v_0$.

We initialize the batch size as $b_0 \geq v_0$ and restrict subsequent batch sizes to power-of-two multiples of b_0 to reduce the frequency of potentially costly batch size adjustments, while allowing distributed

systems to naturally benefit from data parallelism. At each iteration, we round the ratio v_t/v_0 upward to the smallest power of two that is at least this ratio:

$$k_t := 2^{\lceil \log_2(v_t/v_0) \rceil}, \qquad b_t := k_t b_0. \tag{10}$$

This guarantees $k_t \in \{1, 2, 4, 8, \ldots\}$, $k_0 = 1$, and $v_t/v_0 \le k_t < 2v_t/v_0$. Thus, $b_t \ge v_t$, and conditional independence and the BG-a bound give

$$\mathbb{E}_t \left\| g_t - \nabla f(x_t) \right\|^2 \le \frac{B^2 \left\| x_t - x_0 \right\|^a + G^2}{b_t} = \frac{\sigma^2 v_t}{c_\beta k_t b_0} \le \frac{\sigma^2 v_0}{c_\beta b_0} = \frac{G^2}{b_0} \le \frac{\sigma^2}{c_\beta}. \tag{11}$$

The scheduler therefore keeps the minibatch variance bound at or below its initial level. Since b_t is a positive integer and $b_t \ge v_t$, it satisfies the batch size requirement in Theorem 3, and in Theorem 2 when $\beta = 0$ so that $c_\beta = 1$.

While c_β and σ^2 determine the initial batch size b_0, they cancel out in the ratio v_t/v_0. Hence, the multiplier k_t can be computed directly from (x_t, x_0, B, G, a) as

$$k_t = \mathrm{AdaptiveBatchSize}(x_t, x_0, B, G, a) := 2^{\left\lceil \log_2\left(1 + \frac{B^2}{G^2} \left\| x_t - x_0 \right\|^a \right) \right\rceil}. \tag{12}$$

Note that k_t depends on the noise parameters only through the ratio B^2/G^2 and the exponent a. In particular, rescaling B^2 and G^2 by a common factor, such as the minibatch size used when estimating the variance in Section 2, leaves k_t unchanged.

Algorithm 1 applies the proposed batch size scheduler to an optimizer of choice, with a base learning rate schedule $\{\eta_t\}_{t=0}^{T-1}$ specified for the initial batch size b_0. We adjust the learning rate to the selected batch size using the optimizer-dependent scaling rules of linear scaling for SGD-based optimizers (Goyal et al., 2017; McCandlish et al., 2018) and square-root scaling for Adam (Malladi et al., 2022; Merrill et al., 2026). Using the multiplier $k_t = b_t/b_0$, we set

$$\bar{\eta}_t = \eta_t f_{\mathrm{OPT}}(k_t), \qquad f_{\mathrm{OPT}}(k) = \begin{cases} k, & \mathrm{OPT} \in \{\mathrm{SGD}, \mathrm{SGDM}\}, \\ \sqrt{k}, & \mathrm{OPT} \in \{\mathrm{Adam}\}. \end{cases} \tag{13}$$

Consequently, doubling the batch size doubles the learning rate for SGD and SGDM and multiplies it by $\sqrt{2}$ for Adam.

Algorithm 1 Optimization with the Adaptive Batch Size Scheduler

Require: Initialization x_0; initial batch size b_0; optimizer OPT with its hyperparameters; base learning rate schedule $\{\eta_t\}_{t=0}^{T-1}$; iteration budget $T \ge 1$; noise parameters $B, G > 0, a \in (0, 2]$

$s_0 \leftarrow \mathrm{OPT.Initialize}()$ ▷ Initialize optimizer state
for $t = 0, 1, \ldots, T-1$ **do**
  $k_t \leftarrow \textsc{AdaptiveBatchSize}(x_t, x_0, B, G, a)$ ▷ Using equation 12
  $b_t \leftarrow k_t b_0$
  $\bar{\eta}_t \leftarrow \eta_t f_{\mathrm{OPT}}(k_t)$ ▷ Using equation 13
  Draw fresh conditionally independent samples $\xi_{t,1}, \ldots, \xi_{t,b_t}$
  $g_t \leftarrow \frac{1}{b_t} \sum_{i=1}^{b_t} \nabla f(x_t; \xi_{t,i})$
  $(x_{t+1}, s_{t+1}) \leftarrow \mathrm{OPT.Step}(x_t, s_t, g_t, \bar{\eta}_t)$
end for
return x_T

In Algorithm 1, s_t corresponds to the optimizer state at iteration t, such as momentum for SGDM, or first and second moment estimates for Adam, whereas SGD requires no states. The scheduler changes the batch size and adjusts the learning rate, while the optimizer follows its usual state update.

Our analysis requires $b_0 \ge v_0 = c_\beta G^2/\sigma^2$. However, in practice, a suitable target noise level σ^2 is not known a priori and can differ across model-dataset pairs. Therefore, in our experiments, we set b_0 equal to the batch size of the small batch baseline, so that BG-a and the small batch baseline start from the same batch size and the same base learning rate. We choose this as the smallest batch

size that trains stably and attains a competitive final loss after tuning the learning rate through a grid search (see Appendix F).

The scheduler follows the batch size rule based on our theoretical analysis. Since $b_t \geq v_t$ at every iteration, the resulting batch sizes satisfy the variance control condition in Theorems 2 and 3, while the power-of-two rounding enables better compatibility with common pretraining frameworks, e.g., OLMo-core (Team OLMo et al., 2024). As discussed in Section 3.4, the scheduler starts training with a small batch, where the noise is small, and increases the batch size as the iterates move away from initialization and the variance grows. The noise parameters (B, G, a) used by the scheduler are estimated through the measurements in Section 2, connecting our empirical observations and theoretical analysis to a practical scheduler. We discuss further practical considerations in Appendix G. Next, we present experiments showing that, under matched token budgets, our scheduler achieves a lower validation loss in LLM pretraining while using far fewer iterations than small batch training.

5 Experiments

We evaluate our adaptive batch size scheduler in LLM pretraining, where large batch training is particularly important for exploiting data parallelism, and where each training token is seen at most once (online training). Our experiments on image classification, where models are trained for multiple epochs (offline training), are provided in Appendix E.2. We pretrain OLMo2-100M, OLMo2-600M, and OLMo2-1B models on C4 using Chinchilla optimal training budgets of approximately 20 tokens per parameter (Hoffmann et al., 2022), corresponding to 2B, 12B, and 20B tokens, respectively. All runs use AdamW with a linear warmup over the first 5% of training followed by a cosine decay. We compare our scheduler, denoted as BG-a, against a small and a large fixed batch baseline under the same token budget. BG-a starts from the small batch size, uses the noise parameters (B, G, a) estimated in Section 2, and scales the learning rate with the square-root rule in equation 13. Further training details are provided in Appendix F.

Table 1 and Figure 3 show that BG-a achieves the lowest validation loss across all 3 model sizes. Hence, we observe that the optimization benefit of adaptive batching directly translates into a lower validation loss. We emphasize that BG-a achieves these results while using only 8.76%, 8.10%, and 5.25% of the iterations of small batch training for OLMo2-100M, OLMo2-600M, and OLMo2-1B, respectively. Since the scheduler starts from the small batch size and increases it only as the iterates move away from initialization, it uses more iterations than large batch training, but it remains within the same order of magnitude. The reduction in iterations also translates into a shorter training time, with BG-a reducing the wall-clock time of small batch training by approximately 40% for OLMo2-100M, 28% for OLMo2-600M and 20% for OLMo2-1B (Table 5).

The comparison between the two fixed batch baselines further supports our analysis on the BG-a noise model. For OLMo2-100M, whose fitted exponent is $a = 0.98$, small batch training outperforms large batch training. For OLMo2-600M and OLMo2-1B, where the fitted exponents are larger, small batch training attains the worst validation loss among the three methods. This is consistent with our theory, as a faster growth of the variance with the distance from initialization makes a fixed small batch less effective in controlling the noise later in training. Note that as detailed in Appendix F, we select the base learning rate through a grid search for small batch training, and obtain the learning rates for large batch and adaptive batch training through square-root scaling, without additional tuning. This setup favors small batch training, as it is the only configuration whose learning rate is tuned directly. Similarly, the small batch size itself is selected as the smallest batch size that trains stably and attains a competitive final loss in each setting, as also described in Appendix F. We adopt this protocol to ensure that the improvements of our scheduler do not result from an insufficient learning rate or insufficiently small batch size. We also present downstream accuracy scores on HellaSwag and CommonsenseQA in Table 2 in Appendix E. We provide a detailed discussion in Appendix G.

6 Conclusion

In this work, we studied why increasing the batch size during training is effective in LLM pretraining, a common practice that lacks a clear theoretical justification. We approached this question from the perspective of stochastic gradient noise whose variance grows as the iterates move away from initialization, a behavior that the classical bounded variance assumption does not capture. Motivated

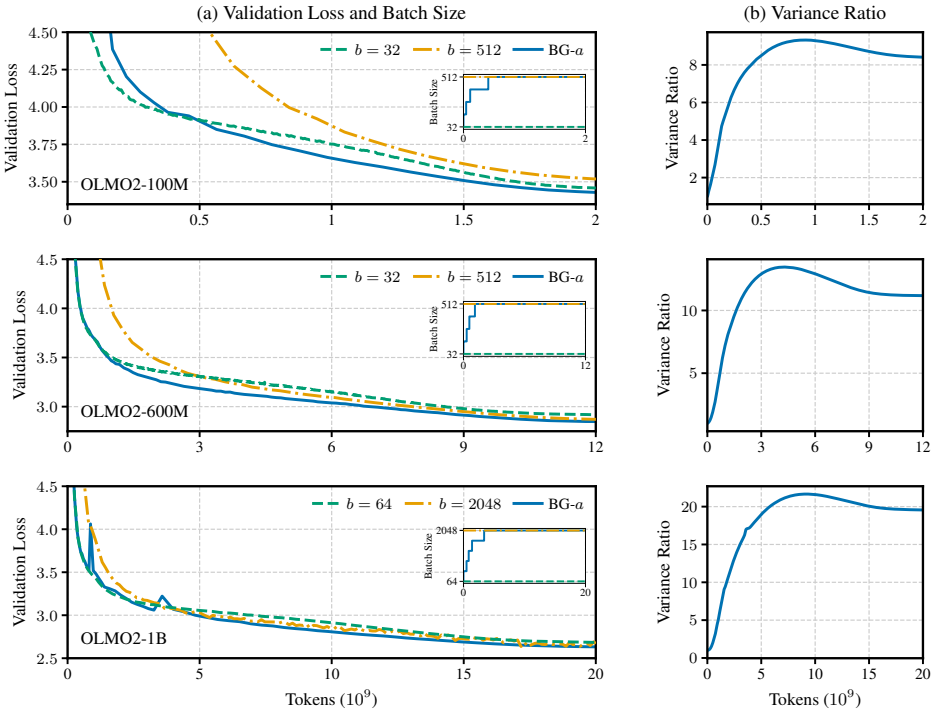


Figure 3: Results for OLMo2-100M (top), OLMo2-600M (middle) and OLMo2-1B (bottom) pretrained on C4. (a) Validation loss for a fixed small batch ($b = 32$ for 100M and 600M, $b = 64$ for 1B), a fixed large batch ($b = 512$ for 100M and 600M, $b = 2048$ for 1B), and BG-a, with batch size across iterations shown in the inset. (b) Modeled variance ratio for BG-a, with parameters $B = 0.379, G = 5.41, a = 0.98$ (100M), $B = 0.132, G = 8.64, a = 1.38$ (600M) and $B = 0.0916, G = 9.77, a = 1.49$ (1B).

Table 1: Validation performance and optimization cost of OLMo2 models pretrained on C4. Validation loss is reported as mean±std across 3 seeds, with iteration percentages relative to small batch runs.

Model	Batch Size	Validation Loss	Iterations (T)	Iterations (%)
OLMo2-100M	Small ($b = 32$)	3.4609 ± 0.0037	30,518	100.00%
	Large ($b = 512$)	3.5139 ± 0.0071	1,908	6.25%
	BG-a	3.4347 ± 0.0085	2,673	8.76%
OLMo2-600M	Small ($b = 32$)	2.9188 ± 0.0062	183,106	100.00%
	Large ($b = 512$)	2.8710 ± 0.0079	11,444	6.25%
	BG-a	2.8482 ± 0.0098	14,824	8.10%
OLMo2-1B	Small ($b = 64$)	2.6845 ± 0.0034	152,588	100.00%
	Large ($b = 2048$)	2.6696 ± 0.0061	4,769	3.13%
	BG-a	2.6328 ± 0.0095	8,015	5.25%

by our empirical observations in language model pretraining, we introduced the BG-a noise model, which interpolates between bounded variance and BG-0 noise, and characterized how the growth of the noise governs the oracle complexity of finding a stationary point. We showed that increasing the batch size as the iterates move away from initialization offers a theoretical explanation for batch size schedules. We argued that small batches are sufficient early in training, while larger batches are needed later to control the growing noise. We also theoretically demonstrated that momentum provides a trade-off between the batch size and the number of iterations. Building on this analysis, we proposed an adaptive batch size scheduler that improves upon both small and large batch training in LLM pretraining. We hope that this perspective encourages further work connecting stochastic optimization theory with the design of batch size schedules in large scale training.

AI USE STATEMENT

We used AI tools to assist with drafting sections of the paper, refining the writing, developing code and proof check. The authors reviewed and edited the resulting text and code and take full responsibility for the paper's content, correctness, and reported results.

ETHICS STATEMENT

This work studies stochastic optimization and batch size scheduling. All models are trained from scratch on publicly available benchmark datasets (C4, CIFAR-10, CIFAR-100, and ImageNet-1K), and the work involves no human subjects, newly collected data, or deployment. Since C4 is derived from web text, it may contain biased or harmful content, but the models trained in this work are used only to evaluate optimization behavior and are not deployed or released as part of this study. We do not foresee additional ethical concerns beyond those generally associated with language model pretraining.

REPRODUCIBILITY STATEMENT

The theoretical assumptions are stated in Section 3, and the full proofs are provided in the appendix. Section 2 describes the procedure for estimating the BG-a noise parameters, and Section 4 specifies the adaptive batch scheduler and its integration with the optimizer. We provide the experimental configurations and computational costs in Appendix F. All main training experiments are repeated with three random seeds. Reported training times are averaged across these runs. Each noise-growth ablation study to determine the BG-a noise model parameters is performed once.

REFERENCES

Ahmet Alacaoglu, Yura Malitsky, and Stephen J Wright. Towards weaker variance assumptions for stochastic optimization. *arXiv preprint arXiv:2504.09951*, 2025.

Yossi Arjevani, Yair Carmon, John C. Duchi, Dylan J. Foster, Nathan Srebro, and Blake Woodworth. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, 199(1): 165–214, 2023.

Francis Bach and Eric Moulines. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems*, volume 24, 2011.

Lukas Balles, Javier Romero, and Philipp Hennig. Coupling adaptive batch sizes with learning rates. In Gal Elidan, Kristian Kersting, and Alexander Ihler (eds.), *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, August 11-15, 2017*. AUAI Press, 2017. URL http://auai.org/uai2017/proceedings/papers/141.pdf.

Julius R Blum. Approximation methods which converge with probability one. *The Annals of Mathematical Statistics*, pp. 382–386, 1954.

Raghu Bollapragada, Richard Byrd, and Jorge Nocedal. Adaptive sampling strategies for stochastic optimization. *SIAM Journal on Optimization*, 28(4):3312–3343, 2018.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Arda Fazla, Ege C Kaya, Antesh Upadhyay, and Abolfazl Hashemi. Lower bounds and proximally anchored SGD for non-convex minimization under unbounded variance. *arXiv preprint arXiv:2604.16620*, 2026.

Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM journal on optimization*, 23(4):2341–2368, 2013.

EG Gladyshev. On stochastic approximation. *Theory of Probability & Its Applications*, 10(2): 275–278, 1965.

Athanasios Glentis, Dawei Li, Chung-Yiu Yau, and Mingyi Hong. Revisiting the Adam-SGD gap in LLM pre-training: The role of large effective learning rates. *arXiv preprint arXiv:2605.17787*, 2026.

Nathan Godey and Yoav Artzi. Lost in backpropagation: The LM head is a gradient bottleneck. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=PMDKvhbE4x.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch SGD: Training ImageNet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Oriol Vinyals, Jack W. Rae, and Laurent Sifre. Training compute-optimal large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Kaiyi Ji, Zhe Wang, Bowen Weng, Yi Zhou, Wei Zhang, and Yingbin Liang. History-gradient aided batch size adaptation for variance reduced algorithms. In *International Conference on Machine Learning*, pp. 4762–4772. PMLR, 2020.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=H1oyRlYgg.

Ahmed Khaled and Peter Richtárik. Better theory for SGD in the nonconvex world. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=AU4qHN2VkS. Survey Certification.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Sadhika Malladi, Kaifeng Lyu, Abhishek Panigrahi, and Sanjeev Arora. On the SDEs and scaling rules for adaptive gradient algorithms. *Advances in Neural Information Processing Systems*, 35: 7697–7711, 2022.

Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*, 2018.

Will Merrill, Shane Arora, Dirk Groeneveld, and Hanna Hajishirzi. Critical batch size revisited: A simple empirical approach to large-batch language model training. *Advances in Neural Information Processing Systems*, 38:116936–116959, 2026.

Hiroki Naganuma, Shagun Gupta, Youssef Briki, Ioannis Mitliagkas, Irina Rish, Parameswaran Raman, and Hao-Jun Michael Shi. Adaptive batch sizes using non-Euclidean gradient noise scales for stochastic sign and spectral descent. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=XMSaWRpEPS.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Christopher J Shallue, Jaehoon Lee, Joseph Antognini, Jascha Sohl-Dickstein, Roy Frostig, and George E Dahl. Measuring the effects of data parallelism on neural network training. *Journal of Machine Learning Research*, 20(112):1–49, 2019.

Ohad Shamir and Tong Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *International Conference on Machine Learning*, pp. 71–79. PMLR, 2013.

Samuel L. Smith, Pieter-Jan Kindermans, and Quoc V. Le. Don't decay the learning rate, increase the batch size. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1Yy1BxCZ.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Michal Guerquin, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 OLMo 2 Furious, 2024. URL https://arxiv.org/abs/2501.00656.

Antesh Upadhyay, Arda Fazla, and Abolfazl Hashemi. Beyond bounded variance: Variance-reduced normalized methods for nonconvex optimization under Blum-Gladyshev noise. In *The Fortieth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=lFix7rA9Yq.

Nikhil Vyas, Depen Morwani, Rosie Zhao, Gal Kaplun, Sham M. Kakade, and Boaz Barak. Beyond implicit bias: The insignificance of SGD noise in online learning. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 49698–49716. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/vyas24a.html.

Hao Yu and Rong Jin. On the computation and communication complexity of parallel SGD with dynamic batch sizes for stochastic non-convex optimization. In *International conference on machine learning*, pp. 7174–7183. PMLR, 2019.

Hanlin Zhang, Depen Morwani, Nikhil Vyas, Jingfeng Wu, Difan Zou, Udaya Ghai, Dean Foster, and Sham Kakade. How does critical batch size scale in pre-training? In *International Conference on Learning Representations*, volume 2025, pp. 66756–66782, 2025.

Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? *Advances in Neural Information Processing Systems*, 33:15383–15393, 2020.

APPENDIX TABLE OF CONTENTS

A Related Work **14**

B Lower bound under L-Smoothness **15**

B.1 Definitions and auxiliary results 15

B.2 Lower-bound theorem and proof 17

B.3 Reduction to unrestricted randomized algorithms 21

B.4 Interpretation and relation to the upper bound 23

C Convergence of SGD **24**

C.1 Discussion on the Expected SFO bound when $a > 2$ 26

D Convergence of SGDM **28**

E Additional Experiments **32**

E.1 LLM Experiments 32

E.2 Vision Experiments 32

F Experimental Details **33**

G Discussion **37**

A Related Work

Large scale training. Larger batches enable greater data parallelism, but their benefits greatly depend on the workload and the optimizer. Extensive experiments conducted by Shallue et al. (2019) show that the reduction in training steps from increasing batch size varies substantially across tasks and depends on hyperparameter tuning and compute budgets. Learning rate scaling is also hugely dependent on the optimizer and training scheme. Goyal et al. (2017); McCandlish et al. (2018) use linear scaling for large batch SGD, whereas Malladi et al. (2022); Merrill et al. (2026) apply square root scaling for AdamW and RMSprop. In large language model pretraining, the critical batch size (CBS) (McCandlish et al., 2018) marks the scale beyond which additional parallelism comes at the expense of token efficiency. Batch size warmup, where training starts with a small batch that is increased as training progresses, is used in large scale pretraining (Brown et al., 2020) and can improve token efficiency compared to training with a large batch from initialization (Merrill et al., 2026). Zhang et al. (2025) find that the CBS depends more strongly on training data size than model size. Merrill et al. (2026) identify limitations of the gradient noise scale as a proxy for critical batch size and estimate it directly through branched training runs. These findings directly motivate our displacement dependent adaptive batch scheduler in language model pretraining, as we couple batch size changes with optimizer-dependent learning rate scaling rules and compare training quality under matched sample or token budgets. While these works approach adaptive batching from the perspective of training efficiency and predicting the critical batch size during training, our theoretical analysis of the BG-a noise model and the batch size rule derived from it offer a complementary theoretical perspective on why increasing the batch size during training helps.

Effect of batch size. Batch size controls both the accuracy of stochastic gradients and the cost of each update, which has motivated researchers to come up with methods to adjust the batch sizes throughout training. Early approaches increase batch size to maintain a suitable gradient signal-to-noise ratio (Bollapragada et al., 2018). As batch size and learning rate jointly influence optimization, the relation between them has been studied by Balles et al. (2017) and Smith et al. (2018), where it is shown that increasing the batch size can have similar effects to learning rate decay in common training schedules. Theoretical work also studies the computational and communication benefits of increasing batches in nonconvex SGD (Yu & Jin, 2019), and the effect of batch size adaptation based on past stochastic gradients in variance reduced methods (Ji et al., 2020). From the empirical side of things, McCandlish et al. (2018) propose the gradient noise scale (GNS) as a useful proxy to the critical batch size during training. Naganuma et al. (2026) extend the GNS framework to cover non Euclidean algorithms such as sign and spectral descent, and their variants.

Online and offline learning. The role of stochastic gradient noise differs between offline training, where the model is trained for multiple epochs over a fixed dataset, and online training, where each sample is seen at most once. In the offline setting, small batch training is known to generalize better than large batch training, which Keskar et al. (2017) attribute to large batches converging to sharper minima. This has led to the view of SGD noise as an implicit regularizer. Vyas et al. (2024) show that this implicit bias advantage disappears in the online setting, where the benefit of small batches becomes purely computational and lower noise runs consistently perform better once the number of gradient steps is controlled. They further hypothesize that online SGD takes noisy steps along the trajectory of noiseless gradient descent. LLM pretraining, which is typically performed for a single epoch over a large corpus, falls into this online regime. This distinction is consistent with the difference we observe between our language model and image classification experiments. Our analysis concerns the optimization of the training objective. In online LLM pretraining, each batch consists of previously unseen tokens, so the training and validation losses estimate the same population loss, and our scheduler attains a lower validation loss than small batch training. In multi-epoch image classification, our scheduler achieves a lower training loss than small batch training, whereas small batch training retains its generalization advantage in test loss.

Stochastic optimization and variance models. The classical analysis of smooth nonconvex optimization often assumes an unbiased stochastic gradient oracle with uniformly bounded variance. Under this setting, SGD achieves $\mathcal{O}(\epsilon^{-4})$ oracle complexity for finding a point satisfying $\mathbb{E}\|\nabla f(x)\|^2 \leq \epsilon^2$ (Ghadimi & Lan, 2013) matching the lower bound of Arjevani et al. (2023). Several analyses have attempted to relax the uniform variance bound, e.g., the expected smoothness framework of Khaled & Richtárik (2023) controls the second moment of the stochastic gradient through objective suboptimality and the gradient norm. The heavy-tail framework (Zhang et al., 2020) analyzes gradient clipping

under bounded moments of order below two, allowing the variance to be infinite. The BG-0 oracle allows variance to grow quadratically with distance from a fixed reference point, building on classical assumptions in stochastic approximation (Blum, 1954; Gladyshev, 1965). Alacaoglu et al. (2025) revisit this condition and develop convergence guarantees for convex optimization without requiring bounded domains. For smooth nonconvex objectives, Fazla et al. (2026) establish an $\Omega(\epsilon^{-6})$ lower bound under BG-0 and obtain matching upper bounds through dynamic batching. Upadhyay et al. (2026) analyze BG-0 under both standard and symmetric generalized smoothness, achieving $\mathcal{O}(\epsilon^{-6})$ oracle complexity with normalized SGD without increasing batch sizes. Existing analyses of the BG-0 oracle assume quadratic growth of the gradient variance with distance from a reference point, while the empirical validity of this assumption in practical training settings remains unexplored. Our work considers variance growth of order $\|x - x_0\|^a$ for $a \in [0, 2]$ and characterizes how the growth exponent determines the optimal dependence of oracle complexity on accuracy. We also assess our proposed BG-a noise model empirically in language model pretraining, with additional vision experiments in Appendix E.2, and use the observed relationship between displacement and gradient variance to explain why increasing the batch size during training helps and to guide adaptive batch size selection.

B Lower bound under L-Smoothness

We build on the lower bound in Theorem 1 of Fazla et al. (2026) with the oracle model

$$\mathbb{E}_\xi[\nabla f(x;\xi)] = \nabla f(x), \qquad \mathbb{E}_\xi \left\|\nabla f(x;\xi) - \nabla f(x)\right\|^2 \le B^2 \left\|x - x_0\right\|^a + G^2, \tag{14}$$

where we use the convention $r^0 = 1$ for all $r \ge 0$, including $r = 0$. Here $a = 2$ gives BG-0, while $a = 0$ gives bounded variance. We consider lower bounded, globally L-smooth objectives with $f(x_0) - f^{\inf} \le \Delta$, where $f^{\inf} := \inf_x f(x)$. We use $\nabla f(x;\xi)$ for the stochastic oracle calls. For the proof of the lower bound, this notation denotes a general unbiased vector estimator and does not require a sample loss whose gradient equals the estimator. We write $g(x;\xi) := \nabla f(x;\xi)$ when referring to an objective–oracle pair (f, g). Each call uses a fresh independent seed and counts as one SFO call.

B.1 Definitions and auxiliary results

Before getting into the proof details, we restate the definitions and chain properties used in the proof. The deterministic chain and masked oracle are those of Arjevani et al. (2023), and the range bound below is adopted from Fazla et al. (2026), which is used to shift the chain to a nonpositive function.

Definition 1 (Progress function)**.** *For $w \in \mathbb{R}^T$ and $\gamma \ge 0$, define*

$$\mathrm{prog}_\gamma(w) := \max\big(\{i \in \{1, \dots, T\} : |w_i| > \gamma\} \cup \{0\}\big).$$

This is the largest coordinate index whose absolute value exceeds the threshold, and is defined to be zero if no such coordinate exists. E.g., $\mathrm{prog}_1(w) = T$ means that $|w_T| > 1$, without imposing any condition on the preceding coordinates.

Definition 2 (Zero-respecting algorithm)**.** *Let $\mathrm{supp}(v) := \{i : v_i \ne 0\}$. Suppose we initialize with $x_0 = 0$ and g_t is the stochastic gradient returned at query x_t. An algorithm is zero-respecting if, almost surely, each query made before termination satisfies*

$$\mathrm{supp}(x_t) \subseteq \bigcup_{j=0}^{t-1} \mathrm{supp}(g_j).$$

For the lower bound in Stage 1 below, we also require its output after τ oracle calls to satisfy

$$\mathrm{supp}(\widehat{x}) \subseteq \bigcup_{j=0}^{\tau-1} \mathrm{supp}(g_j).$$

These restrictions allow a coordinate to become nonzero only after it has appeared in an oracle response.

Definition 3 (Deterministic chain). *For $T \geq 1$ and $w \in \mathbb{R}^T$, define*

$$\bar{f}_T(w) := -\Psi(1)\Phi(w_1) + \sum_{i=2}^{T} \big[\Psi(-w_{i-1})\Phi(-w_i) - \Psi(w_{i-1})\Phi(w_i)\big], \tag{15}$$

where

$$\Psi(r) := \begin{cases} 0, & r \leq 1/2, \\ \exp\left(1 - \dfrac{1}{(2r-1)^2}\right), & r > 1/2, \end{cases} \qquad \Phi(r) := \sqrt{e} \int_{-\infty}^{r} e^{-z^2/2}\, dz.$$

These scalar functions define the base chain used in the construction of the lower bound.

Lemma 1 (Deterministic chain properties (Lemma 2 of Arjevani et al. (2023) restated)). *Setting $\ell_1 = 152$ and $\varsigma = 23$, the function in Definition 3 satisfies the following properties with:*

1. $\bar{f}_T(0) - \inf_w \bar{f}_T(w) \leq 12T$.
2. *Its gradient is globally ℓ_1-Lipschitz and, for every $w \in \mathbb{R}^T$,*
$$\mathrm{Lip}(\nabla \bar{f}_T) \leq \ell_1, \qquad \left\|\nabla \bar{f}_T(w)\right\|_\infty \leq \varsigma. \tag{16}$$
3. *The gradient can extend the chain by at most one coordinate:*
$$\mathrm{prog}_0(\nabla \bar{f}_T(w)) \leq \mathrm{prog}_{1/2}(w) + 1.$$
4. *If $\mathrm{prog}_1(w) < T$, then*
$$\left|\partial_{\mathrm{prog}_1(w)+1} \bar{f}_T(w)\right| > 1, \qquad \left\|\nabla \bar{f}_T(w)\right\|_\infty > 1.$$

Lemma 2 (Range bound from Fazla et al. (2026)). *Let $M_T := \sup_w \bar{f}_T(w)$. Then M_T is finite and for every $w \in \mathbb{R}^T$,*

$$-24T \leq \bar{f}_T(w) - M_T \leq 0. \tag{17}$$

Proof. The definitions above give $0 \leq \Psi \leq e$, $\Psi(1) = 1$, and $0 \leq \Phi \leq \sqrt{2\pi e}$. For every real r, at most one of $\Psi(r)$ and $\Psi(-r)$ is nonzero. Thus each summand with $i \geq 2$ in equation 15 has absolute value at most $e\sqrt{2\pi e}$, and the first term has absolute value at most $\sqrt{2\pi e}$. Hence

$$|\bar{f}_T(w)| \leq \sqrt{2\pi e} + (T-1)e\sqrt{2\pi e} \leq Te\sqrt{2\pi e} < 12T.$$

It follows that $M_T \leq 12T$ and $\inf_w \bar{f}_T(w) \geq -12T$, proving equation 17. □

Definition 4 (Bernoulli masked chain oracle). *Fix $p \in (0, 1]$ and draw $\xi \sim \mathrm{Bernoulli}(p)$. For $w \in \mathbb{R}^T$, we define $\bar{g}_T(w; \xi) \in \mathbb{R}^T$ (coordinate wise) by*

$$[\bar{g}_T(w;\xi)]_i := \partial_i \bar{f}_T(w) \left[1 + \mathbf{1}_{\{i > \mathrm{prog}_{1/4}(w)\}}(\xi/p - 1)\right], \qquad i = 1, \ldots, T. \tag{18}$$

Each oracle call uses a fresh independent seed. Coordinates beyond the progress index are suppressed when $\xi = 0$ and multiplied by $1/p$ when $\xi = 1$.

Lemma 3 (Masked oracle properties from Lemma 3 of Arjevani et al. (2023)). *For every $w \in \mathbb{R}^T$, the oracle in Definition 4 satisfies*

$$\mathbb{E}_\xi[\bar{g}_T(w;\xi)] = \nabla \bar{f}_T(w), \qquad \mathbb{E}_\xi \left\|\bar{g}_T(w;\xi) - \nabla \bar{f}_T(w)\right\|^2 \leq \varsigma^2 \frac{1-p}{p}. \tag{19}$$

Writing $k = \mathrm{prog}_{1/4}(w)$, for both possible seeds its support satisfies

$$\mathrm{supp}(\bar{g}_T(w;\xi)) \subseteq \{1, \ldots, \min(T, k+1)\}, \qquad \mathrm{supp}(\bar{g}_T(w;0)) \subseteq \{1, \ldots, k\}.$$

Thus, a response can extend progress by at most one coordinate, and such an extension requires $\xi = 1$. For each fixed seed, these support statements hold simultaneously for every w, yielding the probability-p zero-chain property used in Arjevani et al. (2023).

Proof. Since $\mathbb{E}_\xi[\xi/p] = 1$, we achieve unbiasedness by taking expectation in equation 18. Furthermore, $\mathrm{prog}_{1/2}(w) \le k$, hence, Lemma 1 implies $\partial_i \bar{f}_T(w) = 0$ for $i > k+1$. When $\xi = 0$, the mask also removes coordinate $k+1$, which proves the support statements.

If $k < T$, the only possibly nonzero component of the oracle error is

$$[\bar{g}_T(w;\xi) - \nabla \bar{f}_T(w)]_{k+1} = \partial_{k+1}\bar{f}_T(w)(\xi/p - 1).$$

Using $\mathbb{E}_\xi[(\xi/p-1)^2] = (1-p)/p$ and applying the gradient bound in Lemma 1 gives

$$\mathbb{E}_\xi \left\|\bar{g}_T(w;\xi) - \nabla \bar{f}_T(w)\right\|^2 = |\partial_{k+1}\bar{f}_T(w)|^2 \frac{1-p}{p} \le \varsigma^2 \frac{1-p}{p}.$$

If $k = T$, the error is zero as the mask does not change any coordinates. Hence, the same bound holds. □

B.2 Lower-bound theorem and proof

We prove the lower bound under the weaker requirement $\mathbb{E}\,\|\nabla f(\widehat{x})\| \le \epsilon$. Since $\mathbb{E}\|\nabla f(\widehat{x})\|^2 \le \epsilon^2$ implies this requirement by Cauchy–Schwarz, the lower bound also applies to the squared gradient criterion in Theorem 1.

Theorem 1 (Lower bound under smoothness)**.** *Under fixed $L, \Delta > 0$, $B, G \ge 0$, and a finite $a \ge 0$, let $\ell = 155$ and $\varsigma = 23$, and suppose*

$$0 < \epsilon^2 \le \frac{L\Delta}{98304\ell}. \tag{20}$$

Any algorithm that returns $\widehat{x}$ after a random number τ of SFO calls and satisfies

$$\mathbb{E}\,\|\nabla f(\widehat{x})\| \le \epsilon \tag{21}$$

for every admissible instance must satisfy

$$\sup_{(f,g)} \mathbb{E}[\tau] \ge \frac{L\Delta}{786432\ell\epsilon^2}\left[1 + \frac{G^2 + B^2(\Delta/(16\epsilon))^a}{64\varsigma^2\epsilon^2}\right]. \tag{22}$$

Consequently, the worst-case expected SFO complexity is

$$\Omega_a\left(\frac{L\Delta}{\epsilon^2} + \frac{G^2 L\Delta}{\epsilon^4} + \frac{B^2 L\Delta^{a+1}}{\epsilon^{4+a}}\right). \tag{23}$$

Proof. **Proof roadmap and relation to prior works.** We first explain the roadmap that we follow for the proof and explain its relation to Arjevani et al. (2023) and PASTA (Fazla et al., 2026). We follow PASTA's travel coordinate and activation construction. This choice is intuitive, as PASTA's variance assumption (BG-0) is a special case of the BG-a noise model with $a = 2$, and its extension to general a is our goal. The travel coordinate forces points with sufficiently small gradient to lie at distance of order Δ/ϵ from initialization, whereas the activation function keeps the noisy chain inactive near initialization and allows it to become active only where the distance-dependent variance budget is sufficiently large. Note that, this is a geometric requirement, as an unrestricted algorithm may jump to a distant point in one query. The established oracle complexity comes from learning the chain coordinates, rather than from the number of steps needed to travel that distance. Note that, the proof strategies in Fazla et al. (2026) are mostly built on top of Arjevani et al. (2023).

In our extension to BG-a, we replace PASTA's quadratic activation variance budget by $B^2(D/2)^a + G^2$ and recalibrate the Bernoulli reveal probability p. The chain length remains of order $L\Delta/\epsilon^2$, while each coordinate revelation requires order $1/p$ calls, with

$$\frac{1}{p} = 1 + \Theta_a\left(\frac{G^2 + B^2(\Delta/\epsilon)^a}{\epsilon^2}\right).$$

Multiplying these two scales gives the claimed rate. We first establish this mechanism for zero-respecting algorithms, then give the compressed random-rotation construction and the random-stopping argument needed for unrestricted randomized algorithms.

Stage 1: Construction and lower bound for zero-respecting algorithms. Let $D = \Delta/(4\epsilon)$ and define $f_0(0) = 0$ with

$$f_0'(u) = \begin{cases} -2\epsilon, & u \le D, \\ -2\epsilon + \frac{L}{2}(u - D), & D < u < D + 4\epsilon/L, \\ 0, & u \ge D + 4\epsilon/L. \end{cases} \tag{24}$$

Writing $f_0^{\inf} := \inf_u f_0(u)$, integration gives $f_0^{\inf} = -\Delta/2 - 4\epsilon^2/L$. Next, we define the twice continuously differentiable activation

$$\phi(u) = \begin{cases} 0, & u \le D/2, \\ 10t^3 - 15t^4 + 6t^5, & D/2 < u < D, \quad t = 2u/D - 1, \\ 1, & u \ge D, \end{cases} \tag{25}$$

which satisfies $0 \le \phi \le 1$, $\phi' \ge 0$, and

$$\|\phi'\|_\infty = \frac{15\epsilon}{\Delta}, \qquad \|\phi''\|_\infty = \frac{640\epsilon^2}{\sqrt{3}\Delta^2}.$$

Use the chain $\bar{f}_T$ and oracle $\bar{g}_T$ from Definitions 3 and 4, with $M_T = \sup_w \bar{f}_T(w)$. Lemmas 1–3 provide the smoothness, range, gradient, and variance bounds used below.

Using the same scaling as in Fazla et al. (2026), we set

$$f_{\text{scaled}}(y) = \frac{L\lambda^2}{2\ell_1}[\bar{f}_T(y/\lambda) - M_T], \qquad \lambda = \frac{4\ell_1\epsilon}{L}, \qquad T = \left\lfloor \frac{L\Delta}{768\ell_1\epsilon^2} \right\rfloor. \tag{26}$$

Then $-\Delta/4 \le f_{\text{scaled}} \le 0$, $\text{Lip}(\nabla f_{\text{scaled}}) \le L/2$, and $\nabla f_{\text{scaled}}(y) = 2\epsilon\nabla\bar{f}_T(y/\lambda)$. Hence, subtracting M_T makes the scaled chain nonpositive. Consequently, since $\phi' \ge 0$, the activation contribution $\phi'(u)f_{\text{scaled}}(y)$ cannot cancel the negative travel-coordinate gradient. Defining

$$f(u, y) = f_0(u) + \phi(u)f_{\text{scaled}}(y), \qquad x_0 = (0, 0), \tag{27}$$

and writing $x = (u, y)$ and $f^{\inf} := \inf_x f(x)$, we get $f(x_0) = 0$ and

$$f(x_0) - f^{\inf} \le \frac{3\Delta}{4} + \frac{4\epsilon^2}{L} \le \Delta.$$

The block-diagonal part of $\nabla^2 f$ has operator norm at most $L/2 + \|\phi''\|_\infty \|f_{\text{scaled}}\|_\infty$. The off-diagonal part has operator norm $\|\phi'\nabla f_{\text{scaled}}\|$, and $\|\nabla f_{\text{scaled}}\| \le 46\epsilon\sqrt{T}$. Consequently,

$$\left\|\nabla^2 f\right\|_{\text{op}} \le \frac{L}{2} + \frac{160\epsilon^2}{\sqrt{3}\Delta} + \frac{690\epsilon\sqrt{L}}{\sqrt{768\ell_1\Delta}} \le L. \tag{28}$$

Indeed, $\epsilon^2 \le L\Delta/(1536\ell_1)$, which follows from equation 20, bounds each of the last two terms by $L/4$. The gradient is continuous at the junctions of f_0', so integration along line segments extends the bound to global L-smoothness.

Next, observe that the composite gradient is

$$\nabla f(u, y) = \begin{pmatrix} f_0'(u) + \phi'(u)f_{\text{scaled}}(y) \\ \phi(u)\nabla f_{\text{scaled}}(y) \end{pmatrix}.$$

We consider two cases to identify necessary conditions for $\|\nabla f(u, y)\| < 2\epsilon$.

First, if $u \le D$, then $f_0'(u) = -2\epsilon$. Since $\phi'(u) \ge 0$ and $f_{\text{scaled}}(y) \le 0$, we have

$$\partial_u f(u, y) = -2\epsilon + \phi'(u)f_{\text{scaled}}(y) \le -2\epsilon.$$

Therefore, $\|\nabla f(u, y)\| \ge |\partial_u f(u, y)| \ge 2\epsilon$. Thus, a gradient norm strictly below 2ϵ requires $u > D$.

Second, suppose $u > D$ but $\text{prog}_1(y/\lambda) < T$. Then $\phi(u) = 1$, and the scaling in equation 26 together with the gradient lower bound in Lemma 1 gives

$$\begin{aligned} \|\nabla f(u, y)\| &\ge \|\nabla_y f(u, y)\| \\ &= \|\nabla f_{\text{scaled}}(y)\| \\ &= 2\epsilon \left\|\nabla\bar{f}_T(y/\lambda)\right\| \\ &\ge 2\epsilon \left\|\nabla\bar{f}_T(y/\lambda)\right\|_\infty > 2\epsilon. \end{aligned}$$

This also rules out a gradient norm strictly below 2ϵ. Since $\mathrm{prog}_1(y/\lambda) \in \{0, \dots, T\}$, the two cases imply

$$\|\nabla f(u, y)\| < 2\epsilon \quad \Longrightarrow \quad u > D \text{ and } \mathrm{prog}_1(y/\lambda) = T. \tag{29}$$

We now choose a constant variance budget that is admissible wherever the chain is active. Recall that $x = (u, y)$, $x_0 = (0, 0)$, and $\phi(u) = 0$ for $u \le D/2$. Thus, whenever $\phi(u) > 0$,

$$u > D/2 > 0, \qquad \|x - x_0\| = \sqrt{u^2 + \|y\|^2} \ge u > D/2.$$

Since $r \mapsto r^a$ is nondecreasing for $a \ge 0$, the BG-a variance bound throughout this active region satisfies

$$B^2 \|x - x_0\|^a + G^2 \ge B^2 (D/2)^a + G^2.$$

We therefore define the constant budget

$$\widetilde{\sigma}_a^2 := B^2 (D/2)^a + G^2 = B^2 \left(\frac{\Delta}{8\epsilon}\right)^a + G^2, \tag{30}$$

where we use $D = \Delta/(4\epsilon)$ in the last equality. Note that, this budget is a permissible upper bound for the noise we introduce on the active region, it is not a lower bound on the oracle's actual variance. Thus, we construct the oracle below so that its variance is at most $\phi(u)^2 \widetilde{\sigma}_a^2$. Since $0 \le \phi \le 1$, the factor $\phi(u)^2$ makes the noise vanish on the inactive region and keeps it within the chosen budget on the active region.

As the next step, we keep the travel-coordinate component of the gradient exact and introduce randomness only through the chain estimator $\bar{g}_T$. Its Bernoulli seed $\xi \sim \mathrm{Bernoulli}(p)$ permits the next unrevealed chain coordinate to be revealed only when $\xi = 1$. The factor ξ/p in the masked estimator preserves its expectation, while a smaller p increases its variance through the factor $(1-p)/p$ in equation 19. We define the composite oracle and choose p to match this variance bound to the budget above:

$$\nabla f(u, y; \xi) = \begin{pmatrix} f_0'(u) + \phi'(u) f_{\mathrm{scaled}}(y) \\ 2\epsilon\phi(u) \bar{g}_T(y/\lambda; \xi) \end{pmatrix}, \qquad p = \frac{4\varsigma^2\epsilon^2}{4\varsigma^2\epsilon^2 + \widetilde{\sigma}_a^2}. \tag{31}$$

To verify conditional unbiasedness, let $x_t = (u_t, y_t)$ be a query measurable with respect to $\mathcal{F}_t$, and draw $\xi_t \sim \mathrm{Bernoulli}(p)$ independently of $\mathcal{F}_t$. Conditioned on this history, the query is fixed, so equation 19 gives

$$\mathbb{E}_t[\bar{g}_T(y_t/\lambda; \xi_t)] = \nabla \bar{f}_T(y_t/\lambda).$$

Consequently, using $\nabla f_{\mathrm{scaled}}(y_t) = 2\epsilon \nabla \bar{f}_T(y_t/\lambda)$,

$$\begin{aligned} \mathbb{E}_t[\nabla f(x_t; \xi_t)] &= \begin{pmatrix} f_0'(u_t) + \phi'(u_t) f_{\mathrm{scaled}}(y_t) \\ 2\epsilon\phi(u_t) \nabla \bar{f}_T(y_t/\lambda) \end{pmatrix} \\ &= \nabla f(x_t). \end{aligned}$$

The oracle error has a zero travel-coordinate component and can be written explicitly as

$$e_t := \nabla f(x_t; \xi_t) - \nabla f(x_t) = \begin{pmatrix} 0 \\ 2\epsilon\phi(u_t) \left[\bar{g}_T(y_t/\lambda; \xi_t) - \nabla \bar{f}_T(y_t/\lambda)\right] \end{pmatrix}.$$

Taking its conditional second moment and applying equation 19 yields

$$\begin{aligned} \mathbb{E}_t \|e_t\|^2 &= 4\epsilon^2 \phi(u_t)^2 \mathbb{E}_t \left\|\bar{g}_T(y_t/\lambda; \xi_t) - \nabla \bar{f}_T(y_t/\lambda)\right\|^2 \\ &\le 4\epsilon^2 \phi(u_t)^2 \varsigma^2 \frac{1-p}{p}. \end{aligned}$$

On the other hand, the choice of p in equation 31 lies in $(0, 1]$ and satisfies

$$\frac{1-p}{p} = \frac{\widetilde{\sigma}_a^2}{4\varsigma^2\epsilon^2}.$$

Thus, the conditional variance satisfies the desired bound

$$\mathbb{E}_t \|\nabla f(x_t; \xi_t) - \nabla f(x_t)\|^2 \le \phi(u_t)^2 \widetilde{\sigma}_a^2.$$

For $u_t \le D/2$, we have $\phi(u_t) = 0$, so the conditional variance is zero. For $u_t > D/2$, the active-region bound above gives

$$\phi(u_t)^2 \widetilde{\sigma}_a^2 \le \widetilde{\sigma}_a^2 = B^2 (D/2)^a + G^2 \le B^2 \|x_t - x_0\|^a + G^2.$$

Hence, conditional unbiasedness and the BG-a variance bound hold at every query and for every $a \ge 0$, including those in the activation interval $D/2 < u_t < D$. To recover the BG-0 case, setting $a = 2$ in equation 30 gives $\widetilde{\sigma}_2^2 = B^2\Delta^2/(64\epsilon^2) + G^2$. Substituting this expression into p and multiplying its numerator and denominator by $64\epsilon^2$ gives

$$p = \frac{256\varsigma^2\epsilon^4}{B^2\Delta^2 + 64G^2\epsilon^2 + 256\varsigma^2\epsilon^4},$$

recovering the choice in Fazla et al. (2026).

We now translate the stationarity requirement into a lower bound on the number of oracle calls. Consider a zero-respecting algorithm as in Definition 2, with output written as $\widehat{x} = (\widehat{u}, \widehat{y})$, and define the event $A := \{\|\nabla f(\widehat{x})\| < 2\epsilon\}$. By equation 21 and Markov's inequality, we have

$$\Pr(A^c) = \Pr\big(\|\nabla f(\widehat{x})\| \ge 2\epsilon\big) \le \frac{\mathbb{E}\,\|\nabla f(\widehat{x})\|}{2\epsilon} \le \frac{1}{2},$$

so $\Pr(A) \ge 1/2$. Within the event A, equation 29 requires $\mathrm{prog}_1(\widehat{y}/\lambda) = T$, and in particular the T-th chain coordinate of the output must be nonzero. The zero-respecting restriction requires this coordinate to have been revealed before the algorithm outputs it. Starting from $y_0 = 0$, Lemma 3 implies that the masked chain oracle can reveal at most one new chain coordinate per oracle call, and only when the Bernoulli seed equals one. Thus, reaching the T-th coordinate requires at least T Bernoulli successes.

Next, we define τ as the number of oracle calls and denote the number of successes before termination by

$$S_\tau := \sum_{t=0}^{\tau-1} \xi_t.$$

The preceding argument gives $S_\tau \ge T\mathbf{1}_A$, hence

$$\mathbb{E}[S_\tau] \ge T\Pr(A) \ge T/2.$$

Observing that the event $\{\tau > t\}$ means that call t is made and is $\mathcal{F}_t$-measurable and using independence, we obtain

$$\begin{aligned}\mathbb{E}[\mathbf{1}_{\{\tau>t\}}\xi_t] &= \mathbb{E}\big[\mathbf{1}_{\{\tau>t\}}\mathbb{E}[\xi_t \mid \mathcal{F}_t]\big] \\ &= p\Pr(\tau > t).\end{aligned}$$

Next, using Tonelli's theorem and the tail-sum identity for τ yields

$$\begin{aligned}\mathbb{E}[S_\tau] &= \mathbb{E}\big[\sum_{t\ge0} \mathbf{1}_{\{\tau>t\}}\xi_t\big] \\ &= \sum_{t\ge0} \mathbb{E}[\mathbf{1}_{\{\tau>t\}}\xi_t] \\ &= p\sum_{t\ge0} \Pr(\tau > t) = p\mathbb{E}[\tau].\end{aligned}$$

These identities also hold for infinite expectations, in which case the claimed lower bound is immediate. Otherwise, combining them with $\mathbb{E}[S_\tau] \ge T/2$ and dividing by $p > 0$ gives $\mathbb{E}[\tau] \ge T/(2p)$.

For convenience, we weaken this to $\mathbb{E}[\tau] \ge T/(4p)$. The accuracy restriction equation 20 ensures $L\Delta/(768\ell_1\epsilon^2) \ge 2$, so the definition of T and $\lfloor r \rfloor \ge r/2$ for $r \ge 2$ give

$$T \ge \frac{L\Delta}{1536\ell_1\epsilon^2}.$$

Furthermore, equation 31 and equation 30 imply

$$\frac{1}{p} = 1 + \frac{\widetilde{\sigma}_a^2}{4\varsigma^2\epsilon^2} = 1 + \frac{G^2 + B^2(\Delta/(8\epsilon))^a}{4\varsigma^2\epsilon^2}.$$

Substituting both bounds yields

$$\mathbb{E}[\tau] \geq \frac{T}{4p} \geq \frac{L\Delta}{6144\ell_1\epsilon^2}\left[1 + \frac{G^2 + B^2(\Delta/(8\epsilon))^a}{4\varsigma^2\epsilon^2}\right]. \tag{32}$$

Hence, the expected number of oracle calls is lower bounded by the ϵ^{-6} growth term at $a = 2$, which becomes $\epsilon^{-(4+a)}$ for general a.

Stage 2: Applying the reduction to unrestricted randomized algorithms. Next, we extend the analysis to unrestricted randomized algorithms.

Lemma 5 is built on an admissible activated-chain instance for unrestricted randomized algorithms, including unbounded queries, arbitrary output points, and random stopping. It combines the travel-coordinate and activation mechanism of Fazla et al. (2026) with the compressed random-rotation construction of Arjevani et al. (2023). For its construction, we rely on a new chain length $T_\star$ and reveal probability $p_\star$, which replace the parameters in the first stage and are given as

$$T_\star = \left\lfloor \frac{L\Delta}{12288\ell\epsilon^2} \right\rfloor, \qquad p_\star = \frac{64\varsigma^2\epsilon^2}{64\varsigma^2\epsilon^2 + G^2 + B^2(\Delta/(16\epsilon))^a}.$$

Equation 20 ensures that the expression inside the floor defining $T_\star$ is at least eight, which gives

$$T_\star \geq \frac{L\Delta}{24576\ell\epsilon^2}, \qquad \frac{1}{p_\star} = 1 + \frac{G^2 + B^2(\Delta/(16\epsilon))^a}{64\varsigma^2\epsilon^2}.$$

Substituting these expressions into the inequality $\sup_{(f,g)} \mathbb{E}[\tau] \geq T_\star/(32p_\star)$, obtained in Lemma 5, yields

$$\sup_{(f,g)} \mathbb{E}[\tau] \geq \frac{L\Delta}{786432\ell\epsilon^2}\left[1 + \frac{G^2 + B^2(\Delta/(16\epsilon))^a}{64\varsigma^2\epsilon^2}\right],$$

which proves equation 22 and equation 23. Finally, Cauchy–Schwarz gives $\mathbb{E}[\|\nabla f(\widehat{x})\|] \leq (\mathbb{E}[\|\nabla f(\widehat{x})\|^2])^{1/2}$, so the same bound applies to the squared gradient guarantee, which we utilize in our upper bound proofs. □

B.3 Reduction to unrestricted randomized algorithms

We first state the random rotation result used in the reduction. This is the single-query specialization of Lemma 6 of Arjevani et al. (2023), whose oracle returns the exact objective value together with the stochastic gradient. We use this version because the activated construction below requires the value of the rotated chain in order to compute the first, travel-coordinate component of the stochastic gradient oracle.

Lemma 4 (Lemma 6 of Arjevani et al. (2023) restated)**.** *Let $T \geq 1$, $p \in (0, 1]$, and $\delta \in (0, 1)$. Suppose*

$$d \geq \left\lceil 18 \cdot 230^2 \frac{T^2}{p} \log\left(\frac{2T^2}{p\delta}\right) \right\rceil,$$

and let $U \in \mathbb{R}^{d\times T}$ be uniformly distributed over matrices with orthonormal columns, independently of the algorithm's internal randomness and oracle seeds. We define

$$\begin{aligned} \rho(z) &= z/\sqrt{1 + \|z\|^2/R^2}, \qquad R = 230\sqrt{T}, \qquad J(z) = D\rho(z), \\ h_U(z) &= \bar{f}_T(U^\top\rho(z)), \qquad q(z) = \|z\|^2/10, \qquad H_U = h_U + q, \\ v_U(z;\xi) &= J(z)^\top U\bar{g}_T(U^\top\rho(z);\xi) + z/5, \end{aligned}$$

where $J(z) = D\rho(z)$ denotes the Jacobian of ρ and $\xi \sim \mathrm{Bernoulli}(p)$. The algorithm accesses the unknown instance through the responses $(H_U(z), v_U(z;\xi))$. For any adaptive randomized algorithm making one query per call, with no bound on the query norms, and with probability at least $1 - \delta$,

$$\|\nabla H_U(z_j)\| \geq \frac{1}{2} \quad \textit{for every integer } 1 \leq j \leq \frac{T - \log(2/\delta)}{2p}.$$

Lemma 5 (Unrestricted algorithm reduction of the activated BG-a chain)**.** *Fix $L, \Delta > 0$, $B, G \geq 0$, and a finite $a \geq 0$. Let $\ell = 155$, $\varsigma = 23$, and $0 < \epsilon^2 \leq L\Delta/(98304\ell)$. Define*

$$T = \left\lfloor \frac{L\Delta}{12288\ell\epsilon^2} \right\rfloor, \qquad p = \frac{64\varsigma^2\epsilon^2}{64\varsigma^2\epsilon^2 + G^2 + B^2(\Delta/(16\epsilon))^a}.$$

For any integer d satisfying

$$d \geq \left\lceil 18 \cdot 230^2 \frac{T^2}{p} \log\left(\frac{8T^2}{p}\right) \right\rceil, \tag{33}$$

there is a family of lower bounded, globally L-smooth objectives $f_U : \mathbb{R}^{d+1} \to \mathbb{R}$, indexed by matrices $U \in \mathbb{R}^{d\times T}$ with orthonormal columns, and unbiased vector oracles $g_U(x;\xi) = \nabla f_U(x;\xi)$ such that, for $x_0 = 0$ and $f_U^{\inf} := \inf_x f_U(x)$,

$$f_U(x_0) - f_U^{\inf} \leq \Delta, \qquad \mathbb{E}_\xi \left\|g_U(x;\xi) - \nabla f_U(x)\right\|^2 \leq B^2 \left\|x - x_0\right\|^a + G^2 \quad \textit{for every } x.$$

Every adaptive randomized algorithm that returns $\widehat{x}$ at a stopping time τ and satisfies $\mathbb{E}\left\|\nabla f_U(\widehat{x})\right\| \leq \epsilon$ for every function in this family must obey

$$\sup_U \mathbb{E}[\tau \mid U] \geq \frac{T}{32p}.$$

Proof. The random rotation makes the chain directions inaccessible to the algorithm. Compression keeps the input to the chain bounded, even for unbounded queries. Finally, the quadratic penalty ensures that large-norm queries cannot exploit the compression to produce small gradients.

Now, we use U, ρ, J, h_U, q, H_U, and v_U from Lemma 4 with the parameters T, p specified above. Here $\|J\|_{\mathrm{op}} \leq 1$ and $\mathrm{Lip}(J) \leq 3/R$, so we have $\mathrm{Lip}(\nabla h_U) \leq 152 + 69/230 < \ell$, $\|\nabla h_U\| \leq 23\sqrt{T}$, and v_U is unbiased for ∇H_U with variance at most $\varsigma^2(1-p)/p$. Taking $\delta = 1/4$ in Lemma 4, we obtain the dimension requirement equation 33. Hence, with probability at least $3/4$, any adaptive randomized query sequence satisfies

$$\|\nabla H_U(z_j)\| \geq \frac{1}{2} \quad \text{for every integer } 1 \leq j \leq \frac{T - \log 8}{2p}. \tag{34}$$

For the activated construction, we use the following scales

$$\begin{aligned} &D = \frac{\Delta}{8\epsilon}, \qquad s = \frac{16\ell\epsilon}{L}, \qquad \kappa = \frac{Ls^2}{2\ell}, \\ &T = \left\lfloor \frac{L\Delta}{12288\ell\epsilon^2} \right\rfloor, \qquad V_a = B^2(D/2)^a + G^2, \qquad p = \frac{64\varsigma^2\epsilon^2}{64\varsigma^2\epsilon^2 + V_a}. \end{aligned} \tag{35}$$

Now, in a similar way to equation 24–equation 25, with ϵ replaced by 2ϵ, we set

$$A_U(y) = \kappa[h_U(y/s) - M_T], \qquad f_U(u,y) = f_0(u) + \phi(u)A_U(y) + \kappa q(y/s). \tag{36}$$

The same shifted chain bounds give $-\Delta/4 \leq A_U \leq 0$, $\|\nabla A_U\| \leq 184\epsilon\sqrt{T}$, and $\mathrm{Lip}(\nabla A_U) \leq L/2$. The gap is at most $3\Delta/4 + 16\epsilon^2/L \leq \Delta$, and the Hessian bound of the block becomes

$$\left\|\nabla^2 f_U\right\|_{\mathrm{op}} \leq \frac{L}{2} + \frac{L}{10\ell} + \frac{640\epsilon^2}{\sqrt{3}\Delta} + \frac{5520\epsilon^2\sqrt{T}}{\Delta} \leq \frac{7L}{8}.$$

The last three terms are each at most $L/8$ by equation 20, and continuity at the travel-function junctions again gives global smoothness. For $u \leq D$, we have $\partial_u f_U \leq -4\epsilon$; and for $u > D$, we get $\nabla_y f_U = 8\epsilon\nabla H_U(y/s)$. Thus equation 34 forces $\|\nabla f_U\| \geq 4\epsilon$ regardless of u. Denoting $z = y/s$, we get

$$\nabla f_U(u,y;\xi) = \begin{pmatrix} f_0'(u) + \phi'(u)A_U(y) \\ 8\epsilon\{\phi(u)[v_U(z;\xi) - z/5] + z/5\} \end{pmatrix}, \tag{37}$$

which is conditionally unbiased with conditional variance at most $64\epsilon^2\phi(u)^2\varsigma^2(1-p)/p = \phi(u)^2 V_a$. Note that both the inactive and active cases used above satisfy equation 14 at every point. As

$A_U(y) = \kappa[H_U(z) - q(z) - M_T]$, each composite call can be simulated with one call to (H_U, v_U) at $z = y/s$. Hence the first coordinate reveals no additional information about U.

It remains to turn the query bound in equation 34 into a lower bound on the expected stopping time. Setting $m = \lfloor T/(8p) \rfloor$, the accuracy restriction gives $T \geq 8$, and $p \leq 1$, so $T/(8p) \geq 1$. Therefore,

$$m \geq \frac{T}{16p}, \qquad m + 1 \leq \frac{T}{8p} + 1 \leq \frac{T}{4p} \leq \frac{T - \log 8}{2p},$$

where we use $T \geq 8 > 2 \log 8$ in the last inequality. Thus the bound in equation 34 covers m oracle calls with an additional query at the output point.

First, lets consider any algorithm that makes at most m calls and returns $\widehat{x} = (\widehat{u}, \widehat{y})$, and simulate these calls using the oracle (H_U, v_U) as described above. If the algorithm stops before m calls, we keep its output fixed and make the remaining queries at the origin, ignoring their responses. We then query $\widehat{y}/s$ as query $m + 1$. This final query is used only to apply equation 34 and its response does not change the output. With probability at least $3/4$, we get

$$\|\nabla H_U(\widehat{y}/s)\| \geq \frac{1}{2}.$$

If $\widehat{u} \leq D$, the travel-coordinate derivative already has magnitude at least 4ϵ. If $\widehat{u} > D$, then

$$\|\nabla f_U(\widehat{x})\| \geq \|\nabla_y f_U(\widehat{x})\| = 8\epsilon \, \|\nabla H_U(\widehat{y}/s)\| \geq 4\epsilon.$$

Hence every algorithm with a budget of m calls satisfies

$$\Pr\big(\|\nabla f_U(\widehat{x})\| \geq 4\epsilon\big) \geq \frac{3}{4}.$$

Now let $\mathcal{A}$ be the algorithm in Lemma 5, with stopping time τ and output $\widehat{x}$. Its guarantee holds for every fixed U, so averaging over U gives $\mathbb{E}\,\|\nabla f_U(\widehat{x})\| \leq \epsilon$. Markov's inequality then yields

$$\Pr\big(\|\nabla f_U(\widehat{x})\| \geq 4\epsilon\big) \leq \frac{\mathbb{E}\,\|\nabla f_U(\widehat{x})\|}{4\epsilon} \leq \frac{1}{4}.$$

Next, we define a truncated algorithm $\mathcal{A}_m$ that runs $\mathcal{A}$ for at most m calls. If $\mathcal{A}$ stops within this budget, $\mathcal{A}_m$ returns the same output. Otherwise, it returns x_0. Denoting $\widehat{x}_m$ as this truncated algorithm's output, we run the two algorithms with the same U, oracle seeds, and internal randomness. Their outputs agree whenever $\tau \leq m$. Consequently,

$$\begin{aligned}\Pr\big(\|\nabla f_U(\widehat{x}_m)\| \geq 4\epsilon\big) &\leq \Pr\big(\|\nabla f_U(\widehat{x})\| \geq 4\epsilon\big) + \Pr(\tau > m) \\ &\leq \frac{1}{4} + \Pr(\tau > m).\end{aligned}$$

Since $\mathcal{A}_m$ uses at most m calls, its probability on the left is at least $3/4$. Thus

$$\Pr(\tau > m) \geq \frac{3}{4} - \frac{1}{4} = \frac{1}{2}.$$

Finally, $\tau \geq m\mathbf{1}_{\{\tau > m\}}$, and the supremum over fixed rotations is at least the average over U. Therefore,

$$\sup_U \mathbb{E}[\tau \mid U] \geq \mathbb{E}[\tau] \geq m \Pr(\tau > m) \geq \frac{m}{2} \geq \frac{T}{32p}, \tag{38}$$

which proves the expected SFO lower bound and completes the lemma. □

B.4 Interpretation and relation to the upper bound

Based on our lower bound proofs from the previous part, the chain length is $T = \Theta(L\Delta/\epsilon^2)$, while the inverse reveal probability is

$$\frac{1}{p} = 1 + \Theta_a\left(\frac{G^2 + B^2(\Delta/\epsilon)^a}{\epsilon^2}\right). \tag{39}$$

Therefore, the expected number of oracle calls is at least of order T/p. For varying values of a, this gives:

Distance Exponent	Variance Bound	Lower Bound
$a = 0$	$B^2 + G^2$	$\Omega(\epsilon^{-4})$
$0 < a < 2$	$B^2 \|x - x_0\|^a + G^2$	$\Omega_a(\epsilon^{-(4+a)})$
$a = 2$	$B^2 \|x - x_0\|^2 + G^2$	$\Omega(\epsilon^{-6})$

Note that, if $B = 0$ and $G > 0$, the BG-a noise model reduces to the more restrictive bounded variance case, and the lower bound becomes $\Omega(\epsilon^{-4})$, recovering Arjevani et al. (2023). Similarly, if $B = G = 0$, only the deterministic ϵ^{-2} term remains, and the lower bound reduces to the deterministic case. The additional factor ϵ^{-a} to the lower bounds obtained in Fazla et al. (2026) is the distance-dependent term and comes from the allowable variance at distance $\Theta(\Delta/\epsilon)$ from initialization. For $0 \le a \le 2$ and $B > 0$, the lower bound matches the ϵ-dependence of the expected SFO upper bounds for dynamically batched SGD and SGDM, obtained in Theorems 2 and 3, for fixed problem parameters, target variance σ^2, and momentum coefficient β.

C Convergence of SGD

Theorem 2 (Convergence of SGD with Dynamic Batching)**.** *Suppose Assumptions 1 and 2 hold with $0 \le a \le 2$. Under a fixed target variance $\sigma^2 > 0$ and a target accuracy $\epsilon > 0$ satisfying $\epsilon^2 \le \min\{2\sigma^2, L\Delta\}$, denote the batch size b_t, step size η, and total iterations T as*

$$b_t = \max\left\{1, \left\lceil \frac{B^2 \|x_t - x_0\|^a + G^2}{\sigma^2} \right\rceil \right\}, \qquad \eta = \frac{\epsilon^2}{2L\sigma^2}, \qquad T = \left\lceil \frac{4\Delta}{\eta\epsilon^2} \right\rceil. \tag{40}$$

For $t = 0, \dots, T-1$, form the minibatch gradient $g_t = \frac{1}{b_t}\sum_{i=1}^{b_t} \nabla f(x_t; \xi_{t,i})$ from fresh conditionally independent samples and update $x_{t+1} = x_t - \eta g_t$. Then,

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\|\nabla f(x_t)\|^2 \le \epsilon^2, \tag{41}$$

and the expected total stochastic first-order (SFO) oracle complexity satisfies

$$\mathbb{E}[SFO] = \mathbb{E}\left[\sum_{t=0}^{T-1} b_t\right] \le 10L\Delta\left[(\sigma^2 + G^2)\epsilon^{-4} + 50^{a/2}B^2\Delta^a\epsilon^{-(4+a)}\right]. \tag{42}$$

In particular, for a fixed σ^2 independent of ϵ, the expected SFO complexity is $\mathcal{O}(\epsilon^{-(4+a)})$.

Proof. Let $e_t := g_t - \nabla f(x_t)$, and $z_{t,i} := \nabla f(x_t; \xi_{t,i}) - \nabla f(x_t)$. Note that the batch size b_t is chosen *before* sampling that batch. Hence, conditioned on $\mathcal{F}_t$, b_t is a fixed positive integer and $z_{t,i}$ are independent with conditional mean zero. Therefore, we have

$$\mathbb{E}_t[e_t] = 0, \qquad \mathbb{E}_t\|e_t\|^2 = \frac{1}{b_t^2}\sum_{i=1}^{b_t}\mathbb{E}_t\|z_{t,i}\|^2 \le \frac{B^2\|x_t - x_0\|^a + G^2}{b_t} \le \sigma^2, \tag{43}$$

where the cross terms vanish by conditional independence and zero means.

Step 1: Descent lemma. Using L-smoothness and with the SGD update iterates x_t, we get

$$f(x_{t+1}) \le f(x_t) - \eta\langle\nabla f(x_t), g_t\rangle + \frac{L\eta^2}{2}\|g_t\|^2. \tag{44}$$

Taking conditional expectation $\mathbb{E}_t[\cdot]$ and using equation 43

$$\mathbb{E}_t\|g_t\|^2 = \|\nabla f(x_t)\|^2 + \mathbb{E}_t\|e_t\|^2 \le \|\nabla f(x_t)\|^2 + \sigma^2. \tag{45}$$

Consequently, since $\eta \le 1/L$ (as $\epsilon^2 \le 2\sigma^2$),

$$\begin{aligned}\mathbb{E}_t f(x_{t+1}) &\le f(x_t) - \eta\left(1 - \frac{L\eta}{2}\right)\|\nabla f(x_t)\|^2 + \frac{L\eta^2\sigma^2}{2} \\ &\le f(x_t) - \frac{\eta}{2}\|\nabla f(x_t)\|^2 + \frac{L\eta^2\sigma^2}{2}.\end{aligned} \tag{46}$$

Step 2: Telescoping to obtain the stationarity criterion. Taking total expectation of each side of equation 46, summing over $t = 0, \dots, T-1$, and since $\mathbb{E}[f(x_T)] \geq f^{\inf}$, we get

$$\frac{\eta}{2} \sum_{t=0}^{T-1} \mathbb{E} \left\| \nabla f(x_t) \right\|^2 \leq f(x_0) - \mathbb{E}[f(x_T)] + \frac{TL\eta^2\sigma^2}{2} \leq \Delta + \frac{TL\eta^2\sigma^2}{2}. \tag{47}$$

Dividing both sides by $T\eta/2$ and substituting the parameters from equation 40, we end up with

$$\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E} \left\| \nabla f(x_t) \right\|^2 \leq \frac{2\Delta}{T\eta} + L\eta\sigma^2 \leq \frac{\epsilon^2}{2} + \frac{\epsilon^2}{2} = \epsilon^2. \tag{48}$$

Step 3: SFO computation. Telescoping the iterates x_t gives:

$$x_t - x_0 = -\eta \sum_{j=0}^{t-1} \nabla f(x_j) - \eta \sum_{j=0}^{t-1} e_j. \tag{49}$$

Taking the norm squared into total expectation of both sides gives

$$\mathbb{E} \left\| x_t - x_0 \right\|^2 = \mathbb{E} \left\| \eta \sum_{j=0}^{t-1} \nabla f(x_j) + \eta \sum_{j=0}^{t-1} e_j \right\|^2 \tag{50}$$

Applying the inequality $\left\| u + v \right\|^2 \leq 2 \left\| u \right\|^2 + 2 \left\| v \right\|^2$ and Cauchy–Schwarz on the gradient sum gives

$$\mathbb{E} \left\| x_t - x_0 \right\|^2 \leq 2\eta^2 t \sum_{j=0}^{t-1} \mathbb{E} \left\| \nabla f(x_j) \right\|^2 + 2\eta^2 \mathbb{E} \left\| \sum_{j=0}^{t-1} e_j \right\|^2. \tag{51}$$

Now, notice that for $i < j$, e_i is $\mathcal{F}_j$-measurable, hence

$$\mathbb{E}[\langle e_i, e_j \rangle] = \mathbb{E}[\mathbb{E}[\langle e_i, e_j \rangle \mid \mathcal{F}_j]] = \mathbb{E}[\langle e_i, \mathbb{E}[e_j \mid \mathcal{F}_j] \rangle] = 0. \tag{52}$$

Thus, martingale orthogonality and equation 43 imply

$$\mathbb{E} \left\| \sum_{j=0}^{t-1} e_j \right\|^2 = \sum_{j=0}^{t-1} \mathbb{E} \left\| e_j \right\|^2 \leq t\sigma^2. \tag{53}$$

Substituting this and equation 47 into equation 51, and using $t \leq T$, we obtain

$$\mathbb{E} \left\| x_t - x_0 \right\|^2 \leq 2\eta^2 T \left(\frac{2\Delta}{\eta} + TL\eta\sigma^2 \right) + 2\eta^2 T\sigma^2 = 4\eta T\Delta + 2L\eta^3 T^2\sigma^2 + 2\eta^2 T\sigma^2 =: D_T. \tag{54}$$

Step 4: a power growth. For $0 \leq a \leq 2$, the map $u \mapsto u^{a/2}$ is concave on $[0, \infty)$. Hence, using Jensen's inequality gives

$$\mathbb{E} \left\| x_t - x_0 \right\|^a = \mathbb{E} \left[(\left\| x_t - x_0 \right\|^2)^{a/2} \right] \leq \left(\mathbb{E} \left\| x_t - x_0 \right\|^2 \right)^{a/2} \leq (D_T)^{a/2}. \tag{55}$$

Summing b_t, taking expectations and using $\max\{1, \lceil u \rceil\} \leq 1 + u$ for $u \geq 0$, we get

$$\mathbb{E} \left[\sum_{t=0}^{T-1} b_t \right] \leq \sum_{t=0}^{T-1} \left[1 + \frac{G^2}{\sigma^2} + \frac{B^2}{\sigma^2} \mathbb{E}[\left\| x_t - x_0 \right\|^a] \right] \leq T \left(1 + \frac{G^2}{\sigma^2} + \frac{B^2}{\sigma^2} (D_T)^{a/2} \right). \tag{56}$$

Step 5: Substituting the explicit constants. Setting $\epsilon^2 \leq \min\{2\sigma^2, L\Delta\}$ and using the explicit constant values from Theorem 2 gives us

$$\eta = \frac{\epsilon^2}{2L\sigma^2}, \qquad m := \frac{4\Delta}{\eta\epsilon^2} = \frac{8L\sigma^2\Delta}{\epsilon^4} \geq 4. \tag{57}$$

Since $m \geq 4$, the ceiling satisfies $T = \lceil m \rceil \leq m + 1 \leq (5/4)m$. Hence

$$T \leq \frac{10L\sigma^2\Delta}{\epsilon^4}, \qquad \eta T \leq \frac{5\Delta}{\epsilon^2}. \tag{58}$$

We now bound the three terms of D_T (equation 54) separately:

$$4\eta T\Delta \leq \frac{20\Delta^2}{\epsilon^2}, \tag{59}$$

$$2L\eta^3T^2\sigma^2 = 2L\eta\sigma^2(\eta T)^2 \leq \epsilon^2\frac{25\Delta^2}{\epsilon^4} = \frac{25\Delta^2}{\epsilon^2}, \tag{60}$$

$$2\eta^2T\sigma^2 = 2\eta\sigma^2(\eta T) \leq \frac{\epsilon^2}{L}\frac{5\Delta}{\epsilon^2} = \frac{5\Delta}{L} \leq \frac{5\Delta^2}{\epsilon^2}, \tag{61}$$

where the final inequality uses $\epsilon^2 \leq L\Delta$. Combining all three bounds, we get

$$D_T \leq \frac{50\Delta^2}{\epsilon^2}. \tag{62}$$

Combining equation 58, equation 62, and equation 56, we arrive at

$$\mathbb{E}\left[\text{SFO}\right] \leq 10L\Delta\left[(\sigma^2 + G^2)\epsilon^{-4} + 50^{a/2}B^2\Delta^a\epsilon^{-(4+a)}\right], \tag{63}$$

completing the proof.

□

Interpretation of equation 63. We demonstrate below the expected SFO based on the values of a, and discuss the relation to the convergence results in the literature.

Distance exponent	Variance upper bound	Expected SFO upper bound
$a = 0$	$B^2 + G^2$	$\mathcal{O}(\epsilon^{-4})$
$0 < a < 2$	$B^2\left\|x - x_0\right\|^a + G^2$	$\mathcal{O}(\epsilon^{-(4+a)})$
$a = 2$	$B^2\left\|x - x_0\right\|^2 + G^2$	$\mathcal{O}(\epsilon^{-6})$

Connecting bounded variance and BG-0 through a. The exponent a connects the bounded variance setting of ordinary SGD with the BG-0 model of distance-dependent, potentially unbounded variance. At $a = 0$, the convention $\left\|x - x_0\right\|^0 = 1$ gives the uniform variance bound $B^2 + G^2$. The batch size in equation 40 is then constant throughout the run, and equation 63 recovers the $\mathcal{O}(\epsilon^{-4})$ SFO rate of SGD under bounded variance. Independently of a, setting $B = 0$ also gives a uniform variance bound, now equal to G^2, and removes the distance-dependent term from the complexity bound.

At $a = 2$, the oracle bound becomes $B^2\left\|x - x_0\right\|^2 + G^2$, which is the BG-0 model considered in Fazla et al. (2026). The expected SFO bound correspondingly becomes $\mathcal{O}(\epsilon^{-6})$, recovering the smooth nonconvex BG-0 rate. For $0 < a < 2$, the rate $\mathcal{O}(\epsilon^{-(4+a)})$ interpolates between these two endpoints: a specifies the power of the allowed variance growth with distance from initialization. Note that, in our proof, the displacement scale Δ/ϵ contributes a factor $(\Delta/\epsilon)^a$ to the expected batch cost, while the iteration budget remains $\mathcal{O}(\epsilon^{-4})$ for fixed σ^2.

C.1 Discussion on the Expected SFO bound when $a > 2$

From our analysis in Theorem 2, we see that the descent and expected SFO analyses require different moments, given that the variance is of order 2 while the distance norm is of order a. When $0 \leq a \leq 2$, meaning that the moment of the distance growth is of order smaller than or equal to the gradient noise variance, the mismatch in orders do not cause a problem, since we can utilize steps similar

to equation 55. However, when the order of the moment of the distance growth exceed that of the variance, we can no longer control it.

Formally, the variance bound in equation 43 controls the expected decrease in the objective, whereas the batch rule in equation 40 gives

$$\frac{B^2\mathbb{E}\|x_t - x_0\|^a + G^2}{\sigma^2} \le \mathbb{E}[b_t] \le 1 + \frac{B^2\mathbb{E}\|x_t - x_0\|^a + G^2}{\sigma^2}. \tag{64}$$

Thus, when $B > 0$, finite expected batch cost requires a finite a-th moment of the displacement. In the proof of Theorem 2, the second-moment bound equation 54 supplies this control through Jensen's inequality for $0 < a \le 2$. For $a > 2$, a second-moment bound is no longer sufficient. To obtain the same $\mathcal{O}(\epsilon^{-(4+a)})$ expected SFO bound with the prescribed $T = \mathcal{O}(\epsilon^{-4})$ iterations, we need to establish

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|x_t - x_0\|^a = \mathcal{O}\big((\Delta/\epsilon)^a\big). \tag{65}$$

However, this construction may not be possible. As an example, the following construction shows why Assumptions 1 and 2 alone cannot satisfy even the finiteness of this moment: the first SGD update can inherit an infinite a-th moment from a finite variance oracle.

Proposition 1 (Expected SFO cost for $a > 2$)**.** *For every fixed $a > 2$ and target variance $\sigma^2 > 0$, there exist an objective and a stochastic gradient oracle satisfying Assumptions 1 and 2 such that SGD with the batch rule in equation 40 and any fixed step size $\eta > 0$ satisfies $\mathbb{E}[b_1] = \infty$. Consequently, its expected total SFO cost is infinite for every $T \ge 2$.*

Proof. We demonstrate the proof by direct construction of such case. We set $d = 1$, $x_0 = 0$, and $f(x) = (x-1)^2/2$, which gives $L = 1$ and $\Delta = 1/2$. Then, we choose a random variable Z such that

$$\mathbb{E}[Z] = 0, \qquad \mathbb{E}[Z^2] = 1, \qquad \mathbb{E}[|Z|^a] = \infty. \tag{66}$$

For example, a symmetric Pareto variable with tail exponent $p \in (2, a)$ can be rescaled to have unit variance. We denote the stochastic signal $f(x; Z) = f(x) + Z\int_0^x \sqrt{\sigma^2 + |s|^a}\, ds$, whose expectation gives the objective function $f(x)$ and its gradient can be written as

$$g(x; Z) = x - 1 + \sqrt{\sigma^2 + |x|^a}\, Z, \tag{67}$$

using an independent draw of Z at each oracle call. Moreover, we have

$$\mathbb{E}_Z[g(x; Z)] = \nabla f(x), \qquad \mathbb{E}_Z\left\|g(x; Z) - \nabla f(x)\right\|^2 = \sigma^2 + |x|^a. \tag{68}$$

Thus, the oracle satisfies Assumption 2 with $B = 1$ and $G = \sigma$. Now, since the variance at $x_0 = 0$ equals the target variance, the initial batch size is $b_0 = 1$. The first update is therefore

$$g_0 = -1 + \sigma Z_{0,1}, \qquad x_1 = \eta(1 - \sigma Z_{0,1}), \tag{69}$$

Where $Z_{0,1}$ is a single instantiation of Z sampled at time x_0. From this update step, we can see that the second moment $\mathbb{E}[|x_1|^2] = \eta^2(1 + \sigma^2)$ and is finite. However, its a-th moment is infinite. Indeed, $|1 - \sigma Z_{0,1}| \ge \sigma|Z_{0,1}|/2$ whenever $|Z_{0,1}| \ge 2/\sigma$, giving

$$\mathbb{E}[|x_1|^a] \ge \left(\frac{\eta\sigma}{2}\right)^a \mathbb{E}\left[|Z_{0,1}|^a \mathbf{1}_{\{|Z_{0,1}| \ge 2/\sigma\}}\right] = \infty. \tag{70}$$

The next batch size satisfies

$$b_1 = \left\lceil 1 + \frac{|x_1|^a}{\sigma^2} \right\rceil \ge 1 + \frac{|x_1|^a}{\sigma^2}. \tag{71}$$

Hence equation 70 gives $\mathbb{E}[b_1] = \infty$, and $\mathbb{E}[\sum_{t=0}^{T-1} b_t] \ge \mathbb{E}[b_1] = \infty$ for $T \ge 2$. □

The obstruction is a mismatch between the moment controlled by the oracle assumption and the moment required by the batch rule. Variance control remains sufficient for the expected stationarity analysis, but it does not control the a-th displacement moment that determines expected SFO cost when $a > 2$. Recovering the rate requires a quantitative bound such as equation 65 and finiteness alone would not establish its dependence on ϵ.

Note that, such a quantitative bound can be obtained with new assumptions that bound the gradient noise moments of order a when $a \geq 2$, hence, matching the orders of noise growth and distance growth. This type of analysis, notation given below for convenience, requires further thought and new analysis. Interestingly, it resembles the heavy-tailed p-moment analysis common in the literature (Zhang et al., 2020). We leave this interesting intersection of p-th moment growth and a-th moment distance growth for future work.

$$\mathbb{E}_\xi \left\|\nabla f(x;\xi) - \nabla f(x)\right\|^p \leq B^2 \left\|x - x_0\right\|^a + G^2. \tag{72}$$

D CONVERGENCE OF SGDM

Theorem 3 (Convergence of SGDM with Dynamic Batching)**.** *Suppose Assumptions 1 and 2 hold with $0 \leq a \leq 2$. Fix the momentum coefficient $\beta \in [0,1)$ and a target variance $\sigma^2 > 0$, and define $c_\beta := \frac{1-\beta}{1+\beta}$. For a target accuracy $\epsilon > 0$ satisfying $\epsilon^2 \leq \min\{8\sigma^2(1-\beta^2), L\Delta\}$, denote the batch size b_t, step size η, and total iterations T as*

$$\begin{aligned} b_t &= \max\left\{1, \left\lceil c_\beta \frac{B^2 \left\|x_t - x_0\right\|^a + G^2}{\sigma^2} \right\rceil \right\}, \\ \eta &= \frac{c_\beta \epsilon^2}{16 L \sigma^2}, \qquad T = \left\lceil \frac{16\Delta}{\eta \epsilon^2} \right\rceil. \end{aligned} \tag{73}$$

For $t = 0, \ldots, T-1$, form the minibatch gradient $g_t = \frac{1}{b_t}\sum_{i=1}^{b_t} \nabla f(x_t;\xi_{t,i})$ from fresh conditionally independent samples and update

$$m_{-1} = 0, \qquad m_t = \beta m_{t-1} + (1-\beta) g_t, \qquad x_{t+1} = x_t - \eta m_t. \tag{74}$$

Then

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left\|\nabla f(x_t)\right\|^2 \leq \epsilon^2, \tag{75}$$

and the expected total stochastic first-order (SFO) oracle complexity satisfies

$$\mathbb{E}[SFO] = \mathbb{E}\left[\sum_{t=0}^{T-1} b_t\right] \leq 320 L\Delta \left[\left(\frac{\sigma^2}{c_\beta} + G^2\right)\epsilon^{-4} + 725^{a/2} B^2 \Delta^a \epsilon^{-(4+a)}\right]. \tag{76}$$

In particular, for fixed β and σ^2 independent of ϵ, the expected SFO complexity is $\mathcal{O}(\epsilon^{-(4+a)})$.

Proof. Let $e_t := g_t - \nabla f(x_t)$ and $z_{t,i} := \nabla f(x_t;\xi_{t,i}) - \nabla f(x_t)$. The history $\mathcal{F}_t$ contains all samples revealed before iteration t, therefore, x_t, m_{t-1}, and b_t are $\mathcal{F}_t$-measurable. We choose the batch size before sampling, and the errors $z_{t,i}$ are independent with conditional mean zero. Therefore,

$$\mathbb{E}_t[e_t] = 0, \qquad \mathbb{E}_t\left\|e_t\right\|^2 = \frac{1}{b_t^2}\sum_{i=1}^{b_t} \mathbb{E}_t\left\|z_{t,i}\right\|^2 \leq \frac{B^2\left\|x_t - x_0\right\|^a + G^2}{b_t} \leq \frac{\sigma^2}{c_\beta}, \tag{77}$$

where the cross terms vanish due to conditional independence and zero means. The factor c_β corresponds to the variance reduction of the momentum filter. Indeed, the filtered noise $r_t := (1-\beta)\sum_{j=0}^{t} \beta^{t-j} e_j$ satisfies, by martingale orthogonality

$$\mathbb{E}\left\|r_t\right\|^2 = (1-\beta)^2 \sum_{j=0}^{t} \beta^{2(t-j)} \mathbb{E}\left\|e_j\right\|^2 \leq \frac{(1-\beta)^2}{1-\beta^2}\frac{\sigma^2}{c_\beta} = \sigma^2.$$

This controls the filtered noise component.

Step 1: Descent lemma. To handle the components introduced by momentum, we define the auxiliary iterate

$$\widetilde{x}_t := x_t - \frac{\eta\beta}{1-\beta} m_{t-1},$$

where $\widetilde{x}_0 = x_0$ since $m_{-1} = 0$. Expanding the update gives

$$\begin{aligned}\widetilde{x}_{t+1} &= x_t - \eta m_t - \frac{\eta\beta}{1-\beta} m_t \\ &= x_t - \frac{\eta}{1-\beta}[\beta m_{t-1} + (1-\beta) g_t] = \widetilde{x}_t - \eta g_t. \end{aligned} \tag{78}$$

Hence, by using L-smoothness at $\widetilde{x}_t$, we obtain

$$f(\widetilde{x}_{t+1}) \le f(\widetilde{x}_t) - \eta \langle \nabla f(\widetilde{x}_t), g_t \rangle + \frac{L\eta^2}{2} \|g_t\|^2 . \tag{79}$$

Taking conditional expectation $\mathbb{E}_t[\cdot]$ and using

$$\mathbb{E}_t[g_t] = \nabla f(x_t), \qquad \mathbb{E}_t \|g_t\|^2 = \|\nabla f(x_t)\|^2 + \mathbb{E}_t \|e_t\|^2 ,$$

along with the polarization identity and using L-smoothness, we have

$$\begin{aligned} -\langle \nabla f(\widetilde{x}_t), \nabla f(x_t) \rangle &\le -\frac{1}{2} \|\nabla f(x_t)\|^2 + \frac{1}{2} \|\nabla f(\widetilde{x}_t) - \nabla f(x_t)\|^2 \\ &\le -\frac{1}{2} \|\nabla f(x_t)\|^2 + \frac{L^2\eta^2\beta^2}{2(1-\beta)^2} \|m_{t-1}\|^2 . \end{aligned} \tag{80}$$

Applying the previous steps to equation 79, we arrive at

$$\mathbb{E}_t[f(\widetilde{x}_{t+1})] \le f(\widetilde{x}_t) - \left(\frac{\eta}{2} - \frac{L\eta^2}{2}\right) \|\nabla f(x_t)\|^2 + \frac{L^2\eta^3\beta^2}{2(1-\beta)^2} \|m_{t-1}\|^2 + \frac{L\eta^2}{2} \mathbb{E}_t \|e_t\|^2 . \tag{81}$$

Step 2: Telescoping to obtain the stationarity criterion. We first control the additional momentum term in equation 81. Conditional unbiasedness and convexity of the squared norm imply

$$\begin{aligned} \mathbb{E}_t \|m_t\|^2 &= \|\beta m_{t-1} + (1-\beta)\nabla f(x_t)\|^2 + (1-\beta)^2 \mathbb{E}_t \|e_t\|^2 \\ &\le \beta \|m_{t-1}\|^2 + (1-\beta) \|\nabla f(x_t)\|^2 + (1-\beta)^2 \mathbb{E}_t \|e_t\|^2 . \end{aligned} \tag{82}$$

Take expectations and sum over $t = 0, \dots, T-2$. The left side is $\sum_{t=0}^{T-1} \mathbb{E} \|m_{t-1}\|^2$, since $m_{-1} = 0$; the momentum sum on the right is at most this same sum multiplied by β. Moving it to the left, dividing by $1-\beta$, and enlarging the remaining sums gives

$$\sum_{t=0}^{T-1} \mathbb{E} \|m_{t-1}\|^2 \le \sum_{t=0}^{T-1} \mathbb{E} \|\nabla f(x_t)\|^2 + (1-\beta) \sum_{t=0}^{T-1} \mathbb{E} \|e_t\|^2 . \tag{83}$$

For $T = 1$, the same inequality holds because the left side is zero. Next, we take total expectations in equation 81, sum over $t = 0, \dots, T-1$, and use $f(\widetilde{x}_T) \ge f^{\inf}$ with $\widetilde{x}_0 = x_0$. Substituting equation 83 gives

$$\left(\frac{\eta}{2} - \frac{L\eta^2}{2} - A\right) \sum_{t=0}^{T-1} \mathbb{E} \|\nabla f(x_t)\|^2 \le \Delta + \left(A(1-\beta) + \frac{L\eta^2}{2}\right) \sum_{t=0}^{T-1} \mathbb{E} \|e_t\|^2 , \tag{84}$$

where we have denoted $A := \frac{L^2\eta^3\beta^2}{2(1-\beta)^2}$ for convenience. Applying the restriction on the step size $\eta \le (1-\beta)^2/(2L)$, we get

$$\begin{aligned} \frac{L\eta^2}{2} \le \frac{\eta}{4}, \qquad A \le \frac{\eta\beta^2(1-\beta)^2}{8} \le \frac{\eta}{8}, \\ A(1-\beta) = \frac{L\eta^2}{2} \frac{L\eta\beta^2}{1-\beta} \le \frac{L\eta^2}{2} \frac{\beta^2(1-\beta)}{2} \le \frac{L\eta^2}{2}. \end{aligned}$$

Thus the coefficient on the left of equation 84 is at least $\eta/8$, whereas the coefficient of the error sum is at most $L\eta^2$. Dividing and using equation 77 yields

$$\sum_{t=0}^{T-1} \mathbb{E} \|\nabla f(x_t)\|^2 \le \frac{8\Delta}{\eta} + \frac{8L\eta T\sigma^2}{c_\beta}. \tag{85}$$

Finally, dividing by T and substituting equation 73

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla f(x_t)\right\|^2 \le \frac{8\Delta}{\eta T} + \frac{8L\eta\sigma^2}{c_\beta} \le \frac{\epsilon^2}{2} + \frac{\epsilon^2}{2} = \epsilon^2,$$

which proves equation 75.

Step 3: SFO computation. For $1 \le t \le T$, unrolling the momentum recursion and then summing the iterate updates gives

$$m_j = (1-\beta)\sum_{i=0}^{j}\beta^{j-i}g_i,$$

$$x_t - x_0 = -\eta(1-\beta)\sum_{j=0}^{t-1}\sum_{i=0}^{j}\beta^{j-i}g_i = -\eta\sum_{i=0}^{t-1}(1-\beta^{t-i})g_i. \tag{86}$$

For fixed t, denote $w_{t,i} = 1-\beta^{t-i} \in [0,1]$, which are deterministic constants. Hence, splitting $g_i = \nabla f(x_i) + e_i$ gives

$$\mathbb{E}\left\|x_t - x_0\right\|^2 \le 2\eta^2\mathbb{E}\left\|\sum_{i=0}^{t-1}w_{t,i}\nabla f(x_i)\right\|^2 + 2\eta^2\mathbb{E}\left\|\sum_{i=0}^{t-1}w_{t,i}e_i\right\|^2. \tag{87}$$

Cauchy–Schwarz bounds the first term inside its expectation by

$$\mathbb{E}\left\|\sum_{i=0}^{t-1}w_{t,i}\nabla f(x_i)\right\|^2 \le t\sum_{i=0}^{t-1}\mathbb{E}\left\|\nabla f(x_i)\right\|^2 \le T\sum_{i=0}^{T-1}\mathbb{E}\left\|\nabla f(x_i)\right\|^2.$$

For $i < j$, e_i is $\mathcal{F}_j$-measurable and

$$\mathbb{E}\langle e_i, e_j\rangle = \mathbb{E}\big[\langle e_i, \mathbb{E}[e_j \mid \mathcal{F}_j]\rangle\big] = 0.$$

The deterministic weights therefore preserve martingale orthogonality, and equation 77 gives

$$\mathbb{E}\left\|\sum_{i=0}^{t-1}w_{t,i}e_i\right\|^2 = \sum_{i=0}^{t-1}w_{t,i}^2\mathbb{E}\left\|e_i\right\|^2 \le \frac{T\sigma^2}{c_\beta}.$$

Substituting these bounds and equation 85 into equation 87 yields

$$\begin{aligned}\mathbb{E}\left\|x_t - x_0\right\|^2 &\le 2\eta^2 T\left(\frac{8\Delta}{\eta} + \frac{8L\eta T\sigma^2}{c_\beta}\right) + \frac{2\eta^2 T\sigma^2}{c_\beta} \\ &= 16\eta T\Delta + \frac{16L\eta^3T^2\sigma^2}{c_\beta} + \frac{2\eta^2T\sigma^2}{c_\beta} =: D_T.\end{aligned} \tag{88}$$

The same bound holds at $t = 0$ because the displacement is zero.

Step 4: a power growth. For $0 \le a \le 2$, the map $u \mapsto u^{a/2}$ is concave on $[0,\infty)$. Hence, Jensen's inequality and equation 88 imply

$$\mathbb{E}\left\|x_t - x_0\right\|^a = \mathbb{E}\left[(\left\|x_t - x_0\right\|^2)^{a/2}\right] \le \left(\mathbb{E}\left\|x_t - x_0\right\|^2\right)^{a/2} \le D_T^{a/2}. \tag{89}$$

Using $\max\{1, \lceil u\rceil\} \le 1 + u$ for $u \ge 0$, summing the batch sizes, and taking total expectation gives

$$\begin{aligned}\mathbb{E}\left[\sum_{t=0}^{T-1}b_t\right] &\le \sum_{t=0}^{T-1}\left[1 + \frac{c_\beta G^2}{\sigma^2} + \frac{c_\beta B^2}{\sigma^2}\mathbb{E}[\left\|x_t - x_0\right\|^a]\right] \\ &\le T\left(1 + \frac{c_\beta G^2}{\sigma^2} + \frac{c_\beta B^2}{\sigma^2}D_T^{a/2}\right).\end{aligned} \tag{90}$$

Step 5: Substituting the explicit constants. The restriction $\epsilon^2 \le \min\{8\sigma^2(1-\beta^2), L\Delta\}$ that we set gives

$$\frac{c_\beta\epsilon^2}{16L\sigma^2} \le \frac{c_\beta(1-\beta^2)}{2L} = \frac{(1-\beta)^2}{2L}.$$

Hence $\eta \le (1-\beta)^2/(2L)$ as

$$\eta = \frac{c_\beta \epsilon^2}{16L\sigma^2}, \qquad \frac{16\Delta}{\eta\epsilon^2} = \frac{256L\sigma^2\Delta}{c_\beta\epsilon^4}.$$

Since $\eta \le 1/(2L)$ and $\epsilon^2 \le L\Delta$, the expression inside the ceiling for T is at least 32, and hence at least 4. Applying $\lceil u \rceil \le (5/4)u$ for $u \ge 4$ yields

$$T \le \frac{320L\sigma^2\Delta}{c_\beta\epsilon^4}, \qquad \eta T \le \frac{20\Delta}{\epsilon^2}. \tag{91}$$

Next, we bound the three terms of D_T separately

$$16\eta T\Delta \le \frac{320\Delta^2}{\epsilon^2}, \tag{92}$$

$$\frac{16L\eta^3T^2\sigma^2}{c_\beta} = \frac{16L\eta\sigma^2}{c_\beta}(\eta T)^2 \le \epsilon^2\frac{400\Delta^2}{\epsilon^4} = \frac{400\Delta^2}{\epsilon^2}, \tag{93}$$

$$\frac{2\eta^2T\sigma^2}{c_\beta} = \frac{2\eta\sigma^2}{c_\beta}(\eta T) \le \frac{\epsilon^2}{8L}\frac{20\Delta}{\epsilon^2} = \frac{5\Delta}{2L} \le \frac{5\Delta^2}{2\epsilon^2}. \tag{94}$$

The last inequality uses $\epsilon^2 \le L\Delta$. Therefore,

$$D_T \le \frac{725\Delta^2}{\epsilon^2}. \tag{95}$$

Combining equation 91, equation 95, and equation 90 yields

$$\begin{aligned}\mathbb{E}[\mathrm{SFO}] &\le \frac{320L\sigma^2\Delta}{c_\beta\epsilon^4}\left[1 + \frac{c_\beta G^2}{\sigma^2} + \frac{c_\beta B^2}{\sigma^2}\left(\frac{725\Delta^2}{\epsilon^2}\right)^{a/2}\right] \\ &= 320L\Delta\left[\left(\frac{\sigma^2}{c_\beta} + G^2\right)\epsilon^{-4} + 725^{a/2}B^2\Delta^a\epsilon^{-(4+a)}\right].\end{aligned}$$

This proves equation 76 and completes the proof. □

Interpretation of equation 76. For fixed β and σ^2, the dependence on the distance exponent is

Distance exponent	Variance upper bound	Expected SFO upper bound
$a = 0$	$B^2 + G^2$	$\mathcal{O}(\epsilon^{-4})$
$0 < a < 2$	$B^2\|x - x_0\|^a + G^2$	$\mathcal{O}(\epsilon^{-(4+a)})$
$a = 2$	$B^2\|x - x_0\|^2 + G^2$	$\mathcal{O}(\epsilon^{-6})$

When $\beta = 0$, we get $c_\beta = 1$ and $m_t = g_t$, so the update and batch rule reduce to SGD. When $a = 2$, we achieve the $\mathcal{O}(\epsilon^{-6})$ rate from Fazla et al. (2026). When $B = 0$, the batch size is constant and the bound reduces to the classical $\mathcal{O}(\epsilon^{-4})$ rate.

Momentum as a batch size and iteration tradeoff. The momentum coefficient β acts as a hyperparameter controlling the allocation of oracle calls between the batch size and the number of iterations T. Since $c_\beta = (1-\beta)/(1+\beta)$ decreases as β increases, the batch rule in equation 73 prescribes a smaller batch at the same query point. This is observed in the proof, as the momentum filter reduces the noise second moment by the factor c_β, allowing the minibatch estimator to have variance at most σ^2/c_β while its filtered noise component has second moment at most σ^2.

Under the theorem's accuracy restriction, the learning rate and iteration budget satisfy

$$\eta = \frac{c_\beta\epsilon^2}{16L\sigma^2}, \qquad T = \left\lceil \frac{256L\sigma^2\Delta}{c_\beta\epsilon^4} \right\rceil.$$

Thus a larger β permits smaller prescribed batches but requires a smaller learning rate and more iterations. Relative to $\beta = 0$ within this SGDM parameterization, the unrounded batch target is multiplied by c_β, whereas the expression inside the ceiling for T is multiplied by $1/c_\beta$. The

Table 2: Downstream performance of OLMo2 models pretrained on C4. Entries are accuracy (%), reported as mean±std across 3 seeds.

Model	Batch Size	HellaSwag	CommonsenseQA
OLMo2-100M	Small ($b = 32$)	28.98 ± 0.08	29.03 ± 0.06
	Large ($b = 512$)	27.97 ± 0.25	27.27 ± 0.01
	BG-a	29.15 ± 0.30	29.07 ± 0.46
OLMo2-600M	Small ($b = 32$)	40.43 ± 0.11	32.43 ± 0.04
	Large ($b = 512$)	43.05 ± 0.15	32.92 ± 0.11
	BG-a	43.11 ± 0.32	33.33 ± 0.34
OLMo2-1B	Small ($b = 64$)	50.90 ± 0.12	33.66 ± 0.09
	Large ($b = 2048$)	51.12 ± 0.47	34.89 ± 0.22
	BG-a	51.86 ± 0.26	36.53 ± 0.41

batch comparison is at a common query point, since different momentum choices generally produce different trajectories. In equation 76, these factors cancel in the B^2 and G^2 terms, while the minimum cost of one sample per iteration retains the factor $1/c_\beta$. Hence β controls a batch size/iteration tradeoff while preserving the ϵ exponent for fixed β and the bound does not provide a reduction in total SFO cost as momentum increases. Note that the accuracy restriction must also hold for the chosen β, and the expected SFO statement is restricted to $0 \leq a \leq 2$.

E Additional Experiments

E.1 LLM Experiments

We present the downstream accuracy scores on HellaSwag and CommonsenseQA for the three OLMo2 models (100M, 600M, 1B) in Table 2. Each model is trained with 20 tokens per parameter, corresponding to 2B, 12B, and 20B training tokens, respectively. We observe that BG-a attains the highest mean accuracy on both benchmarks, with the clearest gains over small batch training at the larger model sizes, suggesting that the improvement in validation loss carries over to downstream tasks.

E.2 Vision Experiments

Here, we provide our experiments on image classification with the ResNet50 architecture on CIFAR-10, CIFAR-100 and ImageNet-1K. We first repeat the noise growth study of Section 2 for these models, and then evaluate our adaptive batch size scheduler. Unlike LLM pretraining, these models are trained for multiple epochs over a fixed dataset, which allows us to examine how the optimization benefits of adaptive batching translate to train and test performances in the offline setting.

E.2.1 Gradient Noise Growth in Image Classification

We repeat the noise growth study of Section 2 for ResNet50 on CIFAR-10, CIFAR-100, and ImageNet-1K. Unlike the language models, we set $D = I$ for the vision models, so the perturbation direction u is uniformly distributed on the unit sphere. As in the language model study, we consider 50 logarithmically spaced radii and sample 10 independent directions at each radius. For CIFAR-10 and CIFAR-100, the reference batch contains 5000 images, and the estimate uses $M = 16$ stochastic batches, each containing $b = 128$ images. For ImageNet-1K, we use 1024 reference images and $M = 32$ stochastic batches of $b = 16$ images.

Figure 4 shows the variance measurements together with the fitted BG-a curves. The fitted exponents are approximately $a = 1.25$ for CIFAR-10, $a = 1.19$ for CIFAR-100, and $a = 0.96$ for ImageNet-1K. Similar to the language models, the variance grows subquadratically with the distance from initialization, and the BG-a model captures both the initial noise floor (G^2 term) and the distance-dependent growth (from B^2 term).

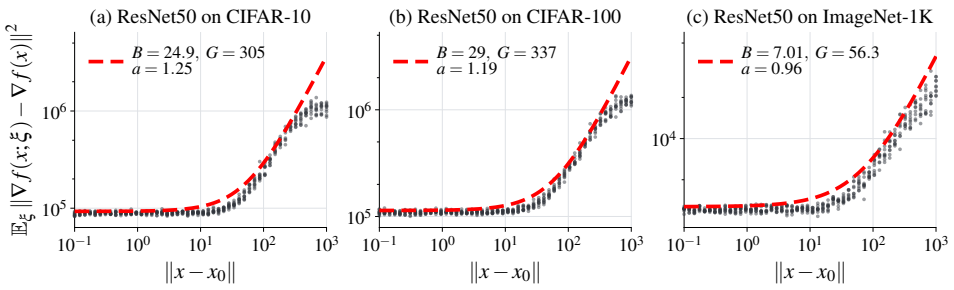


Figure 4: Empirical stochastic-gradient noise of ResNet50 as a function of displacement from initialization. For each radius r, we evaluate $x = x_0 + ru$ along ten independently sampled, isotropic Gaussian directions and plot the corresponding minibatch variance estimates. The dashed red curve in each panel is the fitted BG-a model, $(B^2 \|x - x_0\|^a + G^2)$. The fitted parameters are: **(a)** CIFAR-10: $B = 24.9$, $G = 305$, $a = 1.25$; **(b)** CIFAR-100: $B = 29$, $G = 337$, $a = 1.19$; and **(c)** ImageNet-1K: $B = 7.01$, $G = 56.3$, $a = 0.96$.

E.2.2 Experimental Results

We compare our adaptive batch size scheduler, denoted as BG-a, against a small and a large fixed batch baseline under the same number of training epochs. All runs use SGD with momentum, and the remaining training details are provided in Appendix F. Figures 5, 6 and 7 show the training loss curves, with the batch sizes selected by BG-a shown in the insets, together with the modeled variance ratio. We report the final training and test losses in Table 3.

Across all three datasets, BG-a achieves a lower training loss than small batch training, while using only 14–16% of its iterations. Compared to large batch training, BG-a attains a lower training loss on CIFAR-10 and a slightly higher training loss on CIFAR-100 and ImageNet-1K, with a similar number of iterations. The improvement over small batch training is in line with our theoretical analysis: BG-a controls the variance growth as the iterates move away from initialization, whereas small batch training does not and attains the highest training loss in all three settings.

However, unlike in LLM pretraining, where each batch consists of previously unseen tokens and the validation loss directly reflects the optimized objective, a lower training loss does not directly translate into a lower test loss in image classification. On CIFAR-10 and CIFAR-100, small batch training attains the lowest test loss despite having the highest training loss, while BG-a improves upon large batch training in test loss on both datasets. This contrast is consistent with the distinction between offline and online learning discussed in Appendix A. In multi-epoch training on small datasets such as CIFAR, the models nearly interpolate the training data, with training losses of order 10^{-4} and 10^{-3}, and the implicit regularization effect of small batches becomes the dominant factor for generalization (Keskar et al., 2017). In contrast, Vyas et al. (2024) show that this implicit bias advantage disappears in the online setting, where each sample is seen at most once and the benefit of small batches becomes purely computational. LLM pretraining falls into this online regime, where improving the training objective directly improves the validation loss, as observed in Section 5. ImageNet-1K lies between these two cases: its larger dataset keeps the gap between training and test losses small, and the ordering of test losses closely follows the ordering of training losses. We also note that, analysis of stochastic nonconvex optimization often assumes that each iteration draws fresh and independent samples, which holds in online learning (e.g., LLM pretraining) but not in multi-epoch training, where every sample is revisited after the first epoch, i.e., after only 0.5% of training on CIFAR and 1.1% on ImageNet-1K. The stochastic gradients are then no longer independent draws from the data distribution, and the noise model estimated at initialization need not describe the noise encountered later in training.

F Experimental Details

Models and computational resources. For all vision experiments, we use the ResNet50 architecture (He et al., 2016). For ImageNet-1K (Deng et al., 2009), we use the standard PyTorch implementation without architectural modifications. For CIFAR-10 and CIFAR-100 (Krizhevsky,

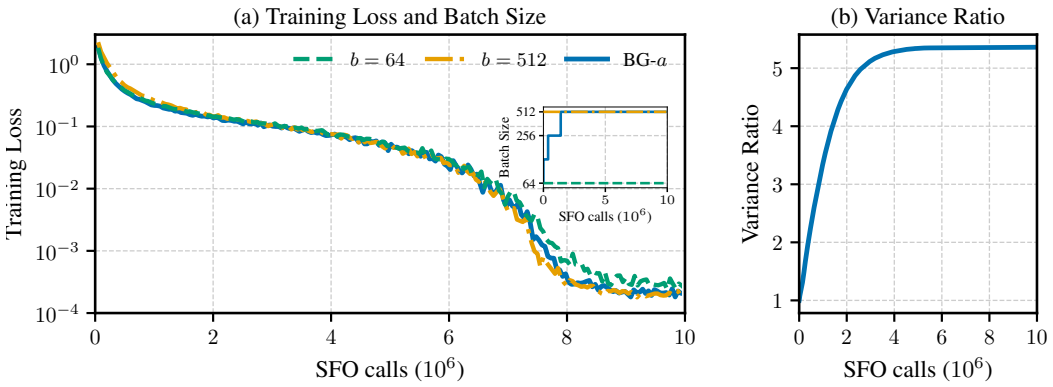


Figure 5: Results for ResNet50 trained on CIFAR-10. (a) Training loss for fixed batches $b = 64$, $b = 512$, and BG-a, with batch size across iterations shown in the inset. (b) Modeled variance ratio for BG-a, with parameters $a = 1.25, B = 24.9, G = 305$

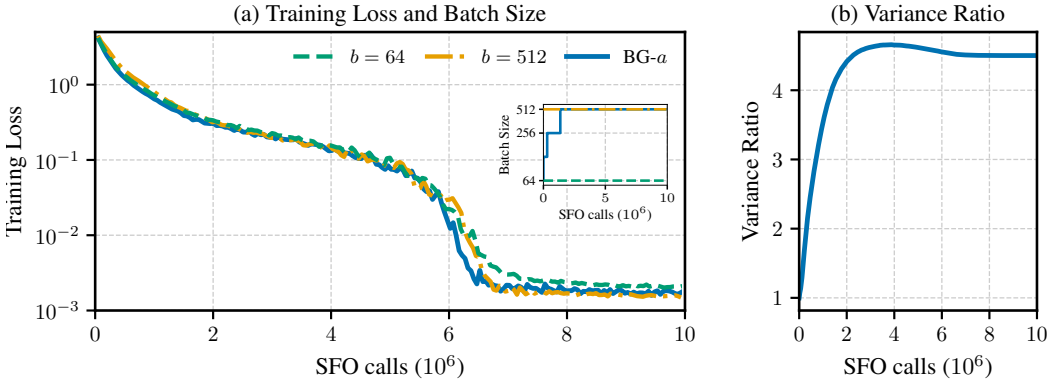


Figure 6: Results for ResNet50 trained on CIFAR-100. (a) Training loss for fixed batches $b = 64$, $b = 512$, and BG-a, with batch size across iterations shown in the inset. (b) Modeled variance ratio for BG-a, with parameters $a = 1.19, B = 29, G = 337$

Table 3: Training and test performance of the ResNet50 model trained on CIFAR-10, CIFAR-100 and ImageNet-1K. Entries are loss, reported as mean±std across 3 seeds. Iterations are averaged across seeds and given as total iterations and percentage relative to the small-batch baseline within each dataset.

Dataset	Batch Size	Train Loss	Test Loss	Iterations (T)	Iterations (%)
CIFAR-10	Small ($b = 64$)	0.000262 ± 0.000020	0.2253 ± 0.0086	156,200	100.00
	Large ($b = 512$)	0.000204 ± 0.000021	0.2508 ± 0.0028	19,400	12.42
	BG-a	0.000200 ± 0.000009	0.2412 ± 0.0028	23,800	15.24
CIFAR-100	Small ($b = 64$)	0.002215 ± 0.000128	0.9522 ± 0.0052	156,200	100.00
	Large ($b = 512$)	0.001496 ± 0.000009	0.9951 ± 0.0244	19,400	12.42
	BG-a	0.001746 ± 0.000193	0.9609 ± 0.0160	23,440	15.00
ImageNet-1K	Small ($b = 128$)	0.9445 ± 0.0011	0.9260 ± 0.0032	900,810	100.00
	Large ($b = 1024$)	0.8227 ± 0.0009	0.9141 ± 0.0017	112,590	12.50
	BG-a	0.8316 ± 0.0008	0.9255 ± 0.0067	127,100	14.11

2009), we replace the initial 7×7 convolution with a 3×3 convolution to accommodate for their 32×32 image sizes. For language model pretraining, we use the OLMo2 family and the OLMo-core repository (Team OLMo et al., 2024) at commit `92870a3`. Model architectures are reported in

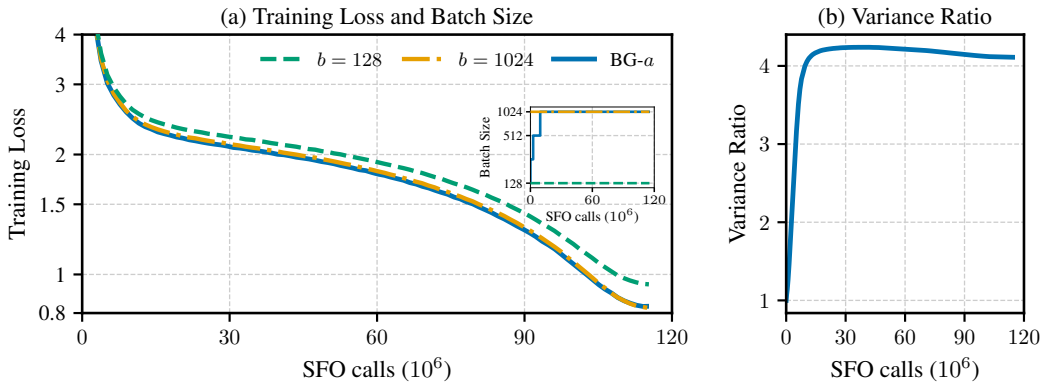


Figure 7: Results for ResNet50 trained on ImageNet-1K. (a) Training loss for fixed batches $b = 128$, $b = 1024$, and BG-a, with batch size across iterations shown in the inset. (b) Modeled variance ratio for BG-a, with parameters $a = 0.96, B = 7.01, G = 56.3$

Table 4: Vision training configurations and mean wall-clock times, using four NVIDIA A100 GPUs. For BG-a, the batch size and base learning rate refer to the initial batch size and its associated base learning rate. $(*)$ means that the given values are the initial starting values.

Dataset	Method	Epochs	Batch size	Base LR	Time (min)
CIFAR-10	Small batch	200	64	0.05	338
	Large batch	200	512	0.4	206
	BG-a	200	64*	0.05*	240
CIFAR-100	Small batch	200	64	0.05	337
	Large batch	200	512	0.4	206
	BG-a	200	64*	0.05*	240
ImageNet-1K	Small batch	90	128	0.05	761
	Large batch	90	1024	0.4	489
	BG-a	90	128*	0.05*	542

Table 7. Main vision experiments use four NVIDIA A100 GPUs, while main language modeling experiments use four NVIDIA H100 GPUs. We use distributed data parallelism (DDP) for all vision models and for OLMo2-100M and OLMo2-600M, and fully sharded data parallelism (FSDP) for OLMo2-1B.

Vision training. We train ResNet50 for 200 epochs on CIFAR-10 and CIFAR-100 and for 90 epochs on ImageNet-1K, using standard data augmentation with random resized cropping. All runs use SGD with momentum 0.9 (SGDM) and weight decay 10^{-4}. We select the base learning rate through a grid search at the small fixed batch size and scale it linearly for larger fixed batch sizes. The learning rate follows a linear warmup over the first five epochs, followed by cosine decay to zero. Table 4 reports the configurations and mean training times.

Language model pretraining. We pretrain OLMo2-100M, OLMo2-600M, and OLMo2-1B on C4 using Chinchilla optimal (Hoffmann et al., 2022) training budgets of approximately 20 tokens per parameter, corresponding to 2B, 12B, and 20B tokens, respectively. All runs use Adam with decoupled weight decay (AdamW), with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and weight decay 0.1, following the repository defaults. We set the sequence length to 2048. We select the base learning rate through a grid search at the small fixed batch size and apply square root scaling for larger fixed batch sizes. The learning rate increases linearly to its base value over the first 5% of training and subsequently follows a cosine schedule to 10% of that value. Table 5 reports the learning rates used, including the initial base learning rates for BG-a.

Table 5: Language model pretraining configurations on C4 and mean wall-clock times, using four NVIDIA H100 GPUs. For BG-a, the batch size and base learning rate refer to the initial batch size and its associated base learning rate. $(*)$ means that the given values are the initial starting values.

Model	Method	Tokens	Batch size	Base LR	Time
OLMo2-100M	Small batch	2B	32	0.004	28m 37s
	Large batch	2B	512	0.016	16m 13s
	BG-a	2B	32^*	0.004^*	17m 20s
OLMo2-600M	Small batch	12B	32	0.001	12h 58m
	Large batch	12B	512	0.004	9h 03m
	BG-a	12B	32^*	0.001^*	9h 17m
OLMo2-1B	Small batch	20B	64	0.001	36h 12m
	Large batch	20B	2048	0.0057	27h 35m
	BG-a	20B	64^*	0.001^*	29h 05m

Table 6: Computational resources and wall-clock times for the BG-a noise growth ablation studies.

Model	Dataset	H100 GPUs	Time
ResNet50	CIFAR-10	1	2m 44s
ResNet50	CIFAR-100	1	2m 38s
ResNet50	ImageNet-1K	1	11m 49s
OLMo2-100M	C4	1	50m 08s
OLMo2-600M	C4	4	2h 34m 28s
OLMo2-1B	C4	4	2h 56m 39s

Selection of the initial batch size and learning rate. For each setting, we tune the base learning rate through a grid search over $\{0.005, 0.01, 0.025, 0.05, 0.1\}$ for vision models and $\{0.0001, 0.0005, 0.001, 0.0025, 0.004, 0.005\}$ for language models, for each candidate small batch size in $\{32, 64, 128\}$ for vision models and $\{16, 32, 64\}$ for language models. We select the smallest batch size that trains stably and attains a competitive final loss. The selected batch size and its tuned learning rate are used both by the small batch baseline and as the initial batch size b_0 and base learning rate of BG-a. The large batch baseline uses the same base learning rate, scaled linearly for SGDM and with the square root rule for AdamW, without additional tuning. The resulting values are reported in Tables 4 and 5.

Adaptive batching and learning rate scaling. BG-a uses the noise parameters estimated by the procedure in Section 2, with fitted values reported in Figure 1. Following Algorithm 1, the scheduler selects $b_t = k_t b_0$, where k_t is the power-of-two multiplier defined in equation 12. The warmup and cosine schedule defines η_t at the initial batch size b_0. At each iteration, we apply the batch dependent adjustment in equation 13, hence, the effective learning rate is $k_t\eta_t$ for the vision experiments that utilize SGDM and $\sqrt{k_t}\eta_t$ for language model pretraining which utilize AdamW. Thus, the learning rate schedule and the batch dependent scaling are applied jointly throughout training.

BG-a noise growth ablations. We perform the ablation studies using the same repositories as their corresponding training experiments. As described in Section 2, these studies evaluate stochastic gradient variance at randomly perturbed parameter states without performing training. Each study uses 50 logarithmically spaced radii and 10 independently sampled directions per radius. For language models, we multiply the Gaussian coordinates corresponding to the LM head by two, leave the remaining coordinates unchanged, and then normalize the full direction to unit Euclidean norm. For vision models, we normalize the unscaled Gaussian direction. In both cases, the perturbation radius equals the total Euclidean displacement from initialization. OLMo2-600M and OLMo2-1B ablations use four NVIDIA H100 GPUs, and all other ablations use one NVIDIA H100 GPU. Table 6 reports the wall-clock cost of each study.

Table 7: Architectural details of the OLMo2 models used in our experiments. All models use a decoder-only Transformer architecture with SwiGLU feed-forward layers, RMSNorm, QK-normalized attention, and rotary positional embeddings.

	OLMo2-100M	OLMo2-600M	OLMo2-1B
Parameters	100M	600M	1B
Layers	12	16	16
Hidden dimension	512	1344	2048
Attention heads	8	16	16
Head dimension	64	84	128
FFN hidden dimension	2048	5376	8192

G Discussion

Our analysis concerns the optimization of the training objective, namely finding approximate stationary points, and our experiments support this view in both language modeling and image classification. Whether an optimization improvement carries over to the test performance depends on the training regime. In LLM pretraining, each token is seen at most once, so the training and validation losses estimate the same population loss, and the implicit bias advantage of small batches disappears (Vyas et al., 2024). Accordingly, adaptive batching attains a lower validation loss than both small and large batch training. In multi-epoch image classification, adaptive batching consistently achieves a lower training loss than small batch training under matched training budgets, but the models revisit the same samples many times, and small batches may attain a lower test loss despite their higher training loss, consistent with their implicit regularization effect (Keskar et al., 2017) (Appendix E.2). Hence, our results suggest that adaptive batching is particularly well suited for the online regime, which matches the setting of modern LLM pretraining.

Our measurements in Section 2 show that the growth exponent a increases with model size, from $a = 0.98$ for OLMo2-100M to $a = 1.49$ for OLMo2-1B. Under our analysis, a faster variance growth increases the oracle complexity and makes controlling the variance growth more important, which is consistent with small batch training falling behind both large batch training and BG-a for OLMo2-600M and OLMo2-1B. Studying whether this trend continues at even larger scales, and how it relates to the critical batch size (Zhang et al., 2025; Merrill et al., 2026), is an interesting direction for future work. Furthermore, Zhang et al. (2025) empirically show that CBS grows primarily with the training data size. Since we train with 20 tokens per parameter, a larger model requires larger data. Our analysis and empirical results also suggest that larger models eventually require larger batch sizes during training, which is in accordance with the results of Zhang et al. (2025).

Our scheduler follows the batch size rule of our analysis, while also incorporating common practices of large scale training. In particular, the scheduler scales the learning rate with the batch size, whereas our theorems use a fixed step size, and our language model experiments use AdamW, whereas our analysis covers SGD and SGDM. Extending the BG-a analysis to adaptive methods and to include learning rate schedulers is a natural next step, and would further clarify the interaction between the batch size, the learning rate, and distance-dependent noise. Similarly, we select the initial batch size empirically, since the target noise level σ^2 is not known a priori, which is also a common practice for batch size schedulers based on GNS (McCandlish et al., 2018; Naganuma et al., 2026).

Limitations. Our experiments cover OLMo2 models of up to 1B parameters trained on C4 with Chinchilla optimal budgets. Evaluating the scheduler at larger scales and on other pretraining corpora would further clarify its practical benefits. In addition, estimating the noise parameters requires a one time measurement before training, whose cost we report in Table 6. Since this measurement does not involve training and depends only on the model and dataset, it can be reused across training runs. Finally, our upper bounds are worst-case guarantees that hold for $a \leq 2$, which covers all the exponents we observe in practice, while extending them to $a > 2$ requires controlling higher moments of the displacement, as discussed in Appendix C.