# Stochastic Approximation with Delayed Updates: Asynchronous Machine Learning Algorithms Study ⋆

**Hong Phuc Nguyen** * **Aditya Mahajan** **
**Silviu-Iulian Niculescu** ***

* *University de Montréal, Montréal, Canada (e-mail: hong.phuc.nguyen@umontreal.ca).*
** *Department of Electrical and Computer Engineering, McGill University, Montréal, Canada (e-mail: aditya.mahajan@mcgill.ca)*
*** *University Paris-Saclay, CNRS, CentraleSupelec, Inria, Laboratory of Signals and Systems (L2S), Gif-sur-Yvette, France (e-mail: silviu.niculescu@centralesupelec.fr)*

**Abstract:** In this paper, we investigate the convergence dynamics of stochastic approximation algorithms under delayed gradient updates and their impact on asynchronous machine learning performance. We present a comprehensive case study comparing the asymptotic behavior of stochastic gradient descent with theoretical guarantees from stochastic approximation theory, examining both decaying and constant learning rate regimes. For decaying rates, we verify convergence but demonstrate that high delays necessitate impractical learning rates; for constant rates, we show that systematically exploiting the learning rate and delay relationship enables robust asynchronous learning. Furthermore, we identify that popular adaptive optimizers degrade under delays because of estimating curvature at current rather than delayed iterates. Finally, we propose a theoretically grounded Hessian-based correction framework using modified Newton-Raphson iterations that explicitly account for gradient staleness.

*Keywords:* Stochastic Approximation, Optimization Algorithms, Delay, Machine Learning.

## 1. INTRODUCTION

Stochastic approximation (SA) algorithms provide a fundamental framework for analyzing the convergence behavior of stochastic iterative methods. These algorithms serve as the theoretical backbone for numerous machine learning applications. Motivated by the prevalence of asynchronous computation in distributed machine learning, understanding SA algorithms with delayed updates has become increasingly important. In distributed settings, gradient updates are computed independently without global synchronization, introducing communication-induced delays that can significantly impact convergence. The general form of delayed stochastic approximation is (Adibi et al., 2024):

$$x_{k+1} = x_k + \alpha_k h(x_{k-d}, w_{k-d+1}), \quad k \geq d \tag{1}$$

where $\{x_k\}_{k\geq 0}$ denotes the parameter iterate, $\{\alpha_k\}_{k\geq 0}$ the learning rate sequence, $h$ the SA operator, $d \in [0, D] \cap \mathbb{N}$ the delay, and $w_{k-d+1}$ the evaluation noise. Recent work Mahajan et al. (2024a) shows that constant step-size SA with delay $d$ converges to the equilibrium of a delay differential equation (DDE) with effective lag $\tau = \alpha d$, establishing that asymptotic behavior depends on the product $\tau$ rather than on $\alpha$ and $d$ individually. However, extending this DDE perspective to deep neural network optimization remains challenging: while stability of low-dimensional linear DDEs can be characterized through spectral methods and by constructing simple Lyapunov-Krasovskii functionals, these approaches do not readily accommodate the high-dimensional nonlinear dynamics of deep learning, where delay interacts with complex loss geometry including saddle points and sharp minima.

Asynchronous methods such as Hogwild (Niu et al., 2011) enable agents to compute gradient updates independently, offering computational advantages on multicore architectures. Delay-Compensated Asynchronous Stochastic Gradient Descent (DC-ASGD) (Zheng et al., 2017) and related approaches (Ren et al., 2020; Wu et al., 2022) address gradient staleness through step-size adaptation and Taylor-based corrections, establishing convergence guarantees across various delay regimes. The former relies on approximation that induce bias or instability when curvature estimates are stale, while the latter is a step-size–based, first-order method that—despite asymptotic convergence guarantees—does not correct stale curvature or preconditioner errors. While these papers provide strong theoretical foundations, the emprical validation of specific stochastic approximation predictions in applied settings has been less thoroughly explored. For instance, with decaying step-sizes, theory predicts convergence to the same equilibrium as the non-delayed case (Bhatnagar, 2011; Mahajan et al., 2024b). Conversely, for constant step-sizes, the DDE framework predicts that $\tau = \alpha d$ determines the asymptotic behavior (Mahajan et al., 2024a), suggesting

⋆ Sponsor and financial support acknowledgment goes here. Paper titles should be written in uppercase and lowercase letters, not all uppercase.

that trajectories with the same $\alpha d$ should be qualitatively equivalent regardless of individual choice of $\alpha$ and $d$. The primary motivation of this work is to systematically investigate these analytical predictions and assess their alignment with empirical observations

The main contributions of this paper are threefold. First, we present a controlled deep-learning case study on Modified National Institute of Standards and Technology (MNIST) classification (LeCun et al., 1998) in which we inject controlled delays into SGD and compare the empirical performance with stochastic approximation guarantees for both decaying and constant learning rates. Decaying rates satisfying Robbins-Monro condition behave consistently with theoretical asymptotic guarantees, but large delays require impractically small learning rates under realistic iteration budgets. For constant $\alpha$, matching $\tau = \alpha d$ across $(\alpha, d)$ configurations aligns training trajectories, in line with the DDE perspective of (Mahajan et al., 2024a). Second, we benchmark popular adaptive optimizers (Adam (Kingma and Ba, 2015) and Adagrad (Duchi et al., 2011)) and approximate second-order methods (L-BFGS (Liu and Nocedal, 1989) and conjugate gradient). Matching SGD-type accuracy requires substantially smaller nominal step sizes (roughly an order of magnitude in our setting).

Finally, we provide a theoretical justification for evaluating Hessian information at *delayed* iterates, i.e., at the point where the stale gradient was computed, to align the curvature and gradient references. Based on this justification, we propose a Delay-Tolerant Conjugate Gradient procedure and show experimentally that it mitigates the Hessian delay shift and improves robustness to large delays relative to standard adaptive and quasi-Newton baselines.

## 2. BACKGROUND ON STOCHASTIC APPROXIMATION WITH DELAYED UPDATES

### *2.1 Stochastic Approximation*

Classical stochastic approximation considers an iterative scheme for updating a parameter $\{x_k\}_{k\geq 0}$, $x_k \in \mathbb{R}^p$, which is initialized at some value $x_0 \in \mathbb{R}^p$, and updated as

$$x_{k+1} = x_k + \alpha_k\big[h(x_k) + \xi_{k+1}\big], \quad k \geq 0 \tag{2}$$

where $\{\alpha_k\}_{k\geq 0}$ is the learning rate sequence, $h\colon \mathbb{R}^p \to \mathbb{R}^p$, and $\{\xi_k\}_{k\geq 1}$, $\xi_k \in \mathbb{R}^p$, is a stochastic noise sequence. Let $\mathcal{H}_k = \sigma(x_{0:k}, \xi_{1:k})$ denote the sigma-algebra of all the information available until iteration $k$. Observe that $\{\xi_k\}_{k\geq 1}$ is a martingale difference sequence with respect to $\{\mathcal{H}_k\}_{k\geq 0}$.

The following assumptions are imposed on the model.

**(F1)** The function $h$ has a unique zero denoted by $x^*$.

**(F2)** The function $h$ is $L$-Lipschitz for some $L > 0$:
$$\|h(x_1) - h(x_2)\|_2 \leq L\|x_1 - x_2\|_2, \quad x_1, x_2 \in \mathbb{R}^p.$$

**(F3)** The point $x^*$ is a globally asymptotically stable equilibrium point of the ODE $\dot{x} = h(x)$.

In addition, the noise $\{\xi_k\}_{k\geq 1}$ is assumed to satisfy:

**(N1)** There exists a constant $\sigma^2 < \infty$ such that
$$\mathbb{E}[\|\xi_{k+1}\|^2 \mid \mathcal{H}_k] \leq \sigma^2(1 + \|x_k - x^*\|^2), \quad \forall k \geq 0.$$

Furthermore, the learning rate is assumed to satisfy one of the following conditions:

**(R1)** The learning rate sequence satisfies $\sum_{k=0}^{\infty} \alpha_k = \infty$, and $\sum_{k=0}^{\infty} \alpha_k^2 < \infty$, a.s.

**(R2)** The learning rate sequence is constant, i.e., $\alpha_k = \alpha$ for some $\alpha$.

The main convergence results are as follows (Borkar and Meyn, 2000):

*Theorem 1.* Suppose conditions (F1), (F2), (N1), (R1) and (R2) hold. In addition, the following conditions hold.

- There exists a limit function $h_\infty\colon \mathbb{R}^p \to \mathbb{R}^p$ such that
$$\lim_{r\to\infty} \frac{h(rx)}{r} = h_\infty(x).$$
- Origin is globally asymptotically stable equilibrium point of the ODE $\dot{x} = h_\infty(x)$.

Then,

(1) Under (R1), $\{x_k\}_{k\geq 1}$ is almost surely bounded. If, in addition, (F3) holds, then $x_k \to x^*$, almost surely.
(2) Under (R2) and (F3), there exist a $\alpha^* > 0$ and $C$ such that if $\alpha < \alpha^*$, then
$$\limsup_{k\to\infty} \mathbb{E}[\|x_k - x^*\|^2] \leq C.$$

### *2.2 Stochastic Approximation with Delayed Updates*

Stochastic approximation with delayed updates refers to iteration of the form (1), where $\{d_k\}_{k\geq 0}$, $d_k \in \mathbb{N}$, is a $\{\mathcal{H}_k\}_{k\geq 0}$ measurable delay sequence and the rest of the terms have the same meaning as Sec. 2.1. We assume that the delay satisfies

**(D)** The delays satisfy $d_k \leq D < \infty$ almost surely.

In addition, instead of (N1), the noise sequence satisfies:

**(N1')** There exists a constant $\sigma^2$ such that,
$$\mathbb{E}[\|\xi_{k+1}\|^2 \mid \mathcal{H}_k] \leq \sigma^2(1 + \|x_{k-d_k} - x^*\|^2),$$

For some of the results, we assume that the following holds instead of (F3).

**(F3')** The point $x^*$ is a globally asymptotically stable equilibrium point of the delay differential equation
$$\dot{x}(t) = h(x(t-\tau)), \quad \tau = \alpha d. \tag{3}$$

The main asymptotic convergence results in this setting are as follows Mahajan et al. (2024b,a):

*Theorem 2.* Suppose (F1), (F2), (N1') and (D) hold. Then we have the following.

(1) Under (R1) and (F3), the iterates $\{x_k\}_{k\geq 0}$ converge to $x^*$ almost surely on the set $\{\|x_k\| < \infty\}$.
(2) Under (R2) and (F3') and additional moment assumptions on the iterates, there exists a $\alpha^*$ and $C$ (which depend on the maximum delay $D$) such that for all $\alpha < \alpha^*$
$$\limsup_{k\to\infty} \mathbb{E}[\|x_k - x^*\|^2] \leq C.$$

Theorem 2 shows that in the regime of constant learning rates, stochastic approximation with delayed updates exhibits fundamentally different qualitative behavior compared to the non-delayed setting (Mahajan et al., 2024a). Sufficient conditions for the stability of iterates for decaying step sizes are presented in Bhatnagar (2011). In

the non-delayed setting, the iterates concentrate in a ball around the equilibrium point on an ODE; in the delayed setting, the iterates concentrate in a ball around the equilibrium point of a DDE.

Both the convergence rate and the variance of the asymptotic noise ball degrade proportionally to the maximum delay $D$. This bound holds in the stable regime; when $\alpha D$ exceeds a threshold, the DDE becomes unstable and the iterates may diverge, as is illustrated in the empirical observations in Section 3. When the step size is held constant, $\alpha_k \equiv \alpha$, stability is preserved for sufficiently small $\alpha$, but convergence occurs only to a stationary distribution centered around $x^*$. For constant step-size delayed stochastic approximation, the iterates converge exponentially fast to a steady-state noise ball with asymptotic mean squared error (Adibi et al., 2024, Theorem. 2):

$$\mathbb{E}[\|x_k - x^*\|^2] \leq C_\tau e^{-\frac{1}{2}\alpha\mu k} + \mathcal{O}\left(\frac{\sigma^2 \alpha L^2 D}{\mu}\right) \qquad (4)$$

where $C_\tau$ depends on the delay, $\mu$ is the strong monotonicity constant, $L$ is the Lipschitz constant, $\sigma^2$ bounds the noise variance, and $D$ is the maximum delay.

## 3. EMPIRICAL STUDY OF STOCHASTIC APPROXIMATION WITH DELAYED UPDATES

### *3.1 Deep Learning Experimental setup*

In this section, we systematically investigate whether the asymptotic behavior predicted by Theorem. 2 aligns with empirical observations in realistic learning settings, directly addressing the gap between theoretical convergence guarantees and practical behavior. We train a fully connected neural network with ReLU activation using stochastic gradient descent (SGD) on MNIST classification (LeCun et al., 1998) under controlled delayed-gradient conditions. The dataset is partitioned into training and test sets and to ensure distinct stochastic trajectories, the training set is randomly shuffled before each iteration.

We systematically vary the learning rates $\{\alpha_k\}$ and *constant* delay $d$ across multiple configurations. To ensure fair comparison across different learning rates, the total iteration count is adjusted inversely proportional to the step size: $T(\alpha) = 10^3/\alpha$, maintaining a constant effective training budget in terms of parameter displacement. Each configuration is evaluated over 20 independent runs with different random seeds, with $10^6$ iterations per run. Performance metrics are recorded every 0.5% iterations, and the Interquartile Mean (IQM) is computed across runs to eliminate outliers.

### *3.2 Constant Learning Rate*

Let $F : \mathbb{R}^p \to \mathbb{R}$ denote the empirical training objective. The simplest instantiation of SGD employs a fixed learning rate $\alpha$ throughout training, yielding the update rule

$$x_{k+1} = x_k + \alpha \nabla F(x_k, w_{k+1}).$$

We run multiple experiments training the neural network using SGD with fixed learning rates $\alpha$ and delays $d$ ranging from $10^0$ to $10^4$. The performance plots are shown in Fig. 1.

In the case of delay update theorem 2 states that during training, the iterates (i.e., the weights of the neural networks) should asymptotically become close to the trajectory of the delay-differential equation $\dot{x}(t) = h(x(t-\tau))$, where the effective delay $\tau = \alpha d$ depends on the learning rate and delay. The "time" of this DDE at iteration $k$ is equal to $k\alpha$. Thus, there is a fundamental trade-off between learning rate and delay, as observed in Fig. 1. Smaller learning rates lead to smaller effective delay $\tau = \alpha d$ but convergence is slow because $k$ must be large for the "DDE time" $k\alpha$ to become large. In contrast, larger learning rates (below an appropriate threshold) lead to faster convergence (as "DDE time" $k\alpha$ grows faster) but the effective delay $\tau = \alpha d$ is large, making performance more sensitive to delay magnitude.

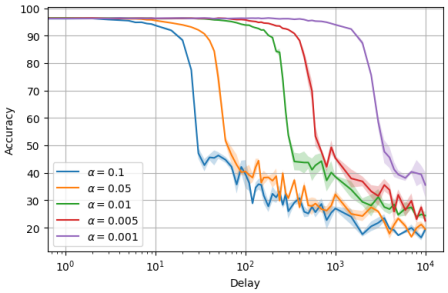


Fig. 1. Performance as a function of delay, SGD training on MNIST: rates $\alpha \in \{0.001, 0.005, 0.01, 0.05, 0.1\}$ and iterations $N \in \{1e6, 5e5, 1e5, 5e4, 1e4\}$.

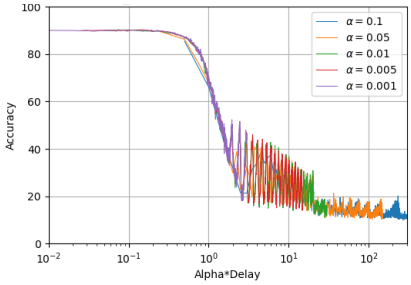


Fig. 2. Performance as a function of the effective delay parameter $\tau = \alpha d$ for different constant learning rates under SGD training on MNIST.

DDEs typically become unstable once the effective delay $\tau$ exceeds a critical delay threshold (Michiels and Niculescu, 2014). For a constant $\alpha$, this corresponds to a critical threshold for the discrete delay $d$ beyond which performance degrades, consistent with the empirical behavior in Fig. 1. Similar behavior has been observed in the literature, where it is argued that delayed gradient updates effectively introduce an implicit momentum term that compounds gradient misalignment, leading to oscillatory or divergent behavior (Mitliagkas et al., 2016). Plotting performance as a function of the effective delay parameter $\tau = \alpha d$ (Fig. 2) reveals that equivalent performance is achieved across different parameter combinations, confirming that performance depends critically on $\tau$, as predicted by Theorem 2. This provides strong empirical evidence that in the presence of delays, the iterates $\{x_k\}_{k\geq 0}$ asymptotically track the trajectory of the DDE in (F3') rather than the ODE in (F3), as occurs in the delay-free case.

### *3.3 Decaying Learning Rate*

We run multiple experiments training the neural network using SGD with decaying learning rates $\alpha_n = A/n^q$ for

some constant $A$ and $q \in \{0.6, 0.7, 0.8, 0.9, 1.0\}$ and delays $d$ ranging from $10^0$ to $10^4$. Note that the learning rates satisfy (R1) for all $q \in (0.5, 1]$. The performance plots for two different values of $A$ are shown in Fig. 3.

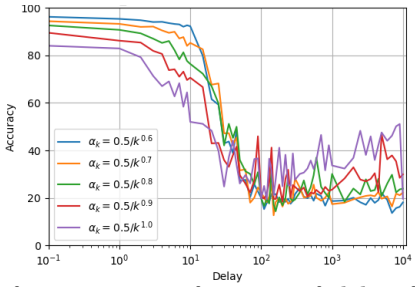


Fig. 3. Performance as a function of delay for different decaying learning rates of the form $0.5/k^q$ for $q \in \{0.6, 0.7, 0.8, 0.9, 1.0\}$ under SGD training on MNIST.

Despite the theoretical predictions of Thm. 2 that the iterates will converge asymptotically to the same equilibrium point for uniformly bounded delays, the empirical performance shown in Fig. 3 suggests that in practice performance still degrades once delay is above a critical threshold. Interestingly, the behavior as a function of delay depends on the constant $A$ rather than the exponent $q$ of the learning rate. Similar behavior has been observed in finite-time analysis of SGD without delays (Bach and Moulines, 2011). Crucially, Eq. (4) shows that both the convergence rate (through the exponential decay factor) and the asymptotic error floor scale linearly with the maximum delay $D$, explaining why larger delays require proportionally longer training horizons to approach comparable accuracy. The experiments confirm this theoretical prediction: achieving performance comparable to the zero-delay regime necessitates reducing the step size $\alpha$ (which slows convergence) extending the iteration budget to accommodate the degraded convergence rate.

## 4. ADAPTIVE LEARNING RATES

SDG is a first-order method and can have slow convergence in ill-conditioned optimization landscapes. For this reason, adaptive learning rate methods such as Adam (Adaptive Moment Estimation) (Kingma and Ba, 2015) and AgaGrad (Adaptive Gradient) (Duchi et al., 2011), which precondition using estimated curvature to mimic a second order method, have become standard in deep learning due to their ability to automatically adjust the learning rates.

In AdaGrad, the component-wise learning rates are computed suing cumulative gradient statistics $G_k = \sum_{i=1}^{k} \nabla F(x_i, w_i) \odot \nabla F(x_i, w_i)$ (where $\odot$ denotes the Hadamard product). Adam combines exponential moving averages of the gradients and their second moments with the parameter update

$$x_{k+1} = x_k - \frac{\alpha}{(\sqrt{\hat{v}_k} + \varepsilon)} \hat{m}_k,$$

where $\hat{m}_k = m_k/(1-\beta_1^k)$, $\hat{v}_k = v_k/(1-\beta_2^k)$ are the bias-corrected moments (Kingma and Ba, 2015) and $\beta_1^k, \beta_2^k$ are appropriate hyperparameters. Linearizing momentum-type updates on a quadratic with Hessian eigenvalues yields a second-order recurrence per eigenmode. Adam's dynamics are more intricate because of the time-varying denominator and the coupled $(m_k, v_k)$ state; for rigorous instability and non-convergence examples for Adam (and fixes such as AMSGrad) see, e.g., Reddi et al. (2018).

One might anticipate that Adam's adaptive per-coordinate learning rates, through the denominator's curvature approximation, would compensate for the delayed gradient information by effectively rescaling the step size to match the underlying geometry, thereby yielding a more accurate effective learning rate than the nominal stepsize $\alpha$ employed in the optimization process. Under this hypothesis, Adam should exhibit improved convergence properties in delay settings compared to non-adaptive methods. However, our empirical findings do not support this expectation: the experiments reveal that Adam's adaptive mechanism does not sufficiently ameliorate the convergence degradation induced by gradient delays.

### *4.1 Empirical Study of Adaptive Learning Rates*

We run multiple experiments training the neural network using Adam and AdaGrad optimizers with initial learning rates $\alpha \in \{10^{-3}, 10^{-4}\}$ for Adam and $\alpha \in \{10^{-2}, 10^{-3}, 10^{-4}\}$ for AdaGrad and delays $d$ ranging from $10^0$ to $10^4$. The performance plots are shown in Fig. 4.

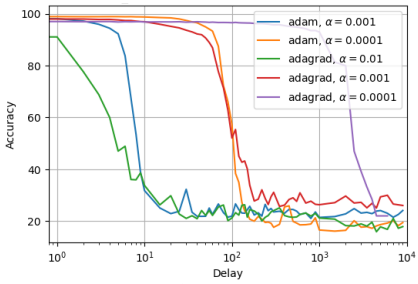


Fig. 4. Performance of Adam and Adagrad optimizers under different delays on MNIST.

The performance of all variations of the adaptive optimizers is comparable to SGD, provided that the initial learning rates are smaller than that of SGD by approximately an order of magnitude (as evidenced in Fig. 1 and Fig. 4, where comparable Adagrad performance occurs at $\alpha = 10^{-4}$ compared to SGD's $\alpha = 10^{-3}$). As specified in the experimental setup, this rescaling directly necessitates a 10-fold increase in the total iteration count—from $T(\alpha) = 10^3/\alpha$—to maintain a constant effective training budget in terms of parameter displacement. Moreover, the best performance is attained only in the final 60–90% of these extended iteration counts across the 20 random seeds, meaning that the full 10-fold computational overhead must be incurred to realize any benefit.

Consequently, training with first-order adaptive step-size optimizers incurs a prohibitively high computational cost in high-delay regimes, requiring ten times as many iterations as SGD to achieve comparable accuracy. This finding suggests that while momentum accumulation and adaptive scaling mechanisms are not entirely incompatible with delayed gradient information, they fundamentally alter the convergence time scales in a manner that negates their practical advantage over simpler methods like SGD.

### *4.2 Emperical Study of Second-order Methods*

Second-order optimization methods such as Newton's method leverage curvature information to address chal-

lenges in ill-conditioned loss landscapes (Martens, 2010). Storing the full Hessian as a $p \times p$ matrix, where $p$ is the number of parameters, is infeasible for large-scale models. Quasi-Newton methods and conjugate gradients provide methods to estimate the Hessian efficiently. For our empirical study, we choose the limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm (Liu and Nocedal, 1989) and Conjugate Gradient-Hessian Vector Product (CG-HVP) (Martens, 2010) to perform Hessian updates. L-BFGS maintains a compact representation of the inverse Hessian approximation using a fixed number of recent gradient differences and parameter updates. CG–HVP, by contrast, avoids storing any Hessian representation entirely, instead applying conjugate-gradient (CG) solvers to compute Hessian–vector products rather than storing full second-derivative matrices.

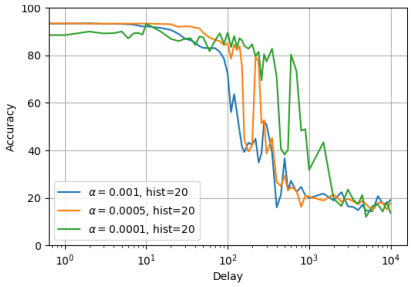


Fig. 5. Performance of L-BFGS approximate Newton method under different delays on MNIST.

We evaluate both L-BFGS and the conjugate gradient method across various algorithm parameters and delays $d \in [10^0, 10^4]$. The results for these experiments are presented in Fig. 5 and Fig. 6, respectively. The results show that despite incorporating second-order information, L-BFGS performance degrades under delay similarly to SGD and adaptive methods. In contrast, the conjugate gradient method is more robust, exhibiting noticeably less deterioration than L-BFGS in the high-delay regime.

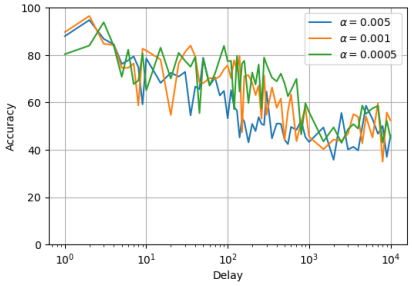


Fig. 6. Performance of original conjugate gradient method under different delays on MNIST.

Quasi-Newton methods such as L-BFGS rely on curvature pairs $(s_k = x_{k+1} - x_k, y_k = \nabla F(x_{k+1}) - \nabla F(x_k))$ that implicitly assume temporal coherence between iterates (Liu and Nocedal, 1989; Byrd et al., 2016). Delayed gradients violate the underlying secant condition, degrading inverse-Hessian approximations and requiring careful synchronization or variance control (Mokhtari et al., 2018).

## 5. DELAY-TOLERANT NEWTON'S METHOD

### *5.1 Theoretical analysis of Newton's method delayed updates*

In this section, we consider the convergence properties of Newton's method with delayed updates for strongly convex optimization problems. Under appropriate delay bounds, we establish global convergence guarantees and provide explicit convergence rates. The derivation extends the recursive argument of Polyak (1987); however, due to the delay term, the resulting update behaves as a perturbed fixed-point iteration instead of exhibiting pure quadratic convergence.

Let $f\colon \mathbb{R}^p \to \mathbb{R}$ be a twice differentiable function that satisfies the following conditions:

**(C1)** $F$ is strongly convex with constant $\ell > 0$, i.e.,

$$F(y) \geq F(x) + \nabla F(x)^\top (y - x) + \frac{\ell}{2}\|y - x\|^2, \quad \forall x, y \in \mathbb{R}^p.$$

**(C2)** $\nabla^2 F$ is Lipschitz continuous with constant $L$, i.e.,

$$\|\nabla^2 F(x) - \nabla^2 F(y)\| \leq L\|x - y\|, \quad \forall x, y \in \mathbb{R}^p.$$

In the Newton's method with delayed updates, given initial condition $\{x_{-2d}, \ldots, x_{-1}, x_0\}$, we update the values $\{x_k\}_{k\geq 0}$ as follows:

$$x_{k+1} = x_k - H_{k-d}^{-1} \nabla F(x_{k-d}), \quad k \geq d, \tag{5}$$

where $H_{k-d} = \nabla^2 F(x_{k-d})$. Define:

$$q_k := \frac{L}{2\ell^2}\|\nabla F(x_k)\|, \quad Q_n := \max_{k \in [n-2d, n]} q_k$$

We assume that the initial conditions satisfy the following assumption:

**(C3)** Let $Q_0 := \max_{k\in[0,2d]} \frac{L}{2\ell^2}\|\nabla F(x_k)\|$, where $\ell$ and $L$ are as defined in (C1) and (C2). If $Q_0 \leq (1-a)/b$ and $Q_k \leq Q_)$. Then from $Q_{k+1} \leq aQ_k + bQ_k^2$ , there exists a $\rho \in (0,1)$ such that for all $k \in \{0, \ldots, 2d-1\}$, we have that $q_k \leq \rho q_{k-1}$ and $Q_0 \geq aQ_0 + bQ_0^2$ where

$$a = \frac{L\rho^{d+1}(1-\rho^d)}{\ell(1-\rho)}, \quad b = \frac{2\rho^{d+1}(1-\rho^d)}{1-\rho} + 1$$

*Theorem 3.* Under (C1), (C2) and (C3), Assume the recurrence

$$Q_{k+1} \leq aQ_k + bQ_k^2, \quad a < 1,\ b > 0.$$

If the initial condition satisfies

$$aQ_0 + bQ_0^2 \leq Q_0 \quad \left(\text{i.e., } Q_0 \leq \frac{1-a}{b}\right),$$

then $Q_k \to 0$ and the Newton's method with delayed updates (5) converges to $x^*$.

**Proof.** A detailed proof is omitted due to space constraints. The high-level idea of the proof is to start with a Taylor series expansion of $\nabla F(x_{k+1})$, simplify the expression using properties (C1) and (C2), and use it to bound the drift $\|x_k - x_{k-d}\|$, and use (C3) to argue convergence.

### *5.2 Delay-Tolerant Conjugate Gradient Newton Method*

This section introduces the Delay-Tolerant Conjugate Gradient Newton method. From the theoretical analysis of Theorem. 3, we can modify the Conjugate Gradient CG-HVP algorithm to handle gradient delays in asynchronous distributed optimization. Unlike classical Newton methods that compute second-order updates using the current model state, our approach decouples Hessian computation from update application by storing precomputed Newton directions in a delay queue. Specifically, at iteration $j$, we compute the Newton direction $v_j$ by solving $(H_j + \lambda I)v_j =$

$-g_j$ via conjugate gradient (where $g_j$ is the gradient and $H_j$ is the Hessian at the current state), but apply the delayed direction $v_{j-d}$ (where $d$ is the delay) to update parameters: $\theta_{j+1} = \theta_j + \alpha v_{j-d}$. This asynchronous update mechanism critically requires recomputing loss graphs at each HVP evaluation within the CG solver to avoid capturing stale computational graphs.

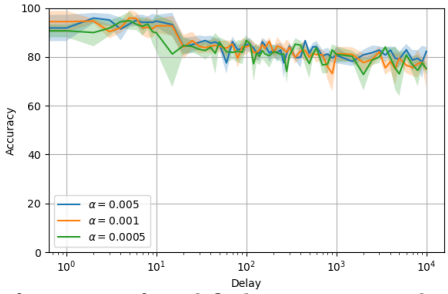


Fig. 7. Performance of modified conjugate gradient method under different delays on MNIST.

The experiment presents the performance of the modified CG-HVP in which the gradient and the Hessian estimates are corrected to account for parameter staleness Fig. 7. It demonstrates the strongest delay tolerance of all methods considered—accuracy remains consistently above 80% across the entire delay range $d \in [10^0, 10^4]$, with only marginal deterioration even at the largest delays tested. Moreover, despite employing smaller step sizes, second-order methods typically require only modest iteration counts—often on the order of twice that of SGD—to achieve comparable convergence. This contrast reinforces the principle established in the preceding section: robustness under asynchronous updates depends critically on ensuring that curvature information is evaluated and applied at a consistent iterate. The residual sensitivity of modified L-BFGS to large delays suggests that its historical buffer, even when corrected, introduces a lag that matrix-free methods avoid entirely. Together, these results validate the delay-aware correction strategy and identify modified CG–HVP as the most practically robust second-order method in high-delay regimes.

## 6. CONCLUSIONS

This paper systematically investigates the gap between theoretical convergence guarantees and empirical behavior in delayed stochastic approximation. Our analysis establishes the fundamental relationship between learning rates and delay tolerance through the effective lag parameter $\tau = \alpha d$, enabling enhanced delay margin without prohibitively small learning rates, while revealing how delay stabilization theory diverges from practical behavior in realistic settings. We further identify the Hessian delay shift as a fundamental limitation shared by adaptive optimizers and second-order methods alike, and propose a Hessian-based correction framework using modified Conjugate Gradient iterations with efficient Hessian-vector products to address this deficiency. Our theoretical analysis establishes convergence guarantees for stochastic approximation with delayed updates, demonstrating that second-order techniques can be effectively adapted for delay-robust optimization. The empirical validation confirms that these theoretical advances translate into practical improvements, providing a principled approach for asynchronous optimization in distributed learning settings.

## ACKNOWLEDGEMENTS

The authors gratefully acknowledge Professor Pierre L'Ecuyer (University de Montréal, Canada) for valuable feedback and insights.

## REFERENCES

Adibi, A. et al. (2024). Stochastic approximation with delayed updates: Finite-time rates under markovian sampling. In *AISTATS*, volume 238, 2746–2754.

Bach, F. and Moulines, E. (2011). Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *NeurIPS 24*, 451–459.

Bhatnagar, S. (2011). The Borkar–Meyn theorem for asynchronous stochastic approximations with delays. *Systems & Control Letters*, 60(7), 472–478.

Borkar, V.S. and Meyn, S.P. (2000). The O.D.E. method for convergence of stochastic approximation and reinforcement learning. *SIAM J Optim.*, 38(2), 447–469.

Byrd, R.H., Hansen, S.L., et al. (2016). A stochastic quasi-Newton method for large-scale optimization. *SIAM J, Optim.*, 26(2), 1008–1031.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12, 2121–2159.

Kingma, D.P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *ICLR*.

LeCun, Y., Bottou, et al. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11), 2278–2324.

Liu, D.C. and Nocedal, J. (1989). On the limited memory BFGS method for large scale optimization. *Math. Prog.*, 45(1–3), 503–528.

Mahajan, A., Niculescu, S.I., and Vidyasagar, M. (2024a). Constant step-size stochastic approximation with delayed updates. In *Conf Decision and Control*.

Mahajan, A., Niculescu, S.I., and Vidyasagar, M. (2024b). A vector almost-supermartingale convergence theorem and its applications. In *Conf Decision and Control*.

Martens, J. (2010). Deep learning via hessian-free optimization. In *ICML*, 735–742.

Michiels, W. and Niculescu, S.I. (2014). *Stability, Control, and Computation for Time-Delay Systems: An Eigenvalue-Based Approach*. SIAM.

Mitliagkas, I., Zhang, C., Hadjis, S., and Ré, C. (2016). Asynchrony begets momentum. In *NeurIPS*, 1037–1045.

Mokhtari, A. et al. (2018). IQN: An incremental quasi-Newton method with local superlinear convergence rate. In *SIAM J. Optim.*, volume 28, 1670–1698. SIAM.

Niu, F., Recht, B., Ré, C., and Wright, S.J. (2011). Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. In *NeurIPS 24*, 693–701.

Polyak, B.T. (1987). *Introduction to Optimization*. Optimization Software, Publications Division, New York.

Reddi, S.J., Kale, S., and Kumar, S. (2018). On the convergence of Adam and beyond. In *ICLR*.

Ren, Z., Zhou, Z., et al. (2020). Delay-adaptive distributed stochastic optimization. In *AAAI Conf*, 5503–5510.

Wu, X., Magnusson, S., et al. (2022). Delay-adaptive step-sizes for asynchronous learning. In *ICML*, 24093–24113.

Zheng, S., Meng, Q., et al. (2017). Asynchronous stochastic gradient descent with delay compensation. In *ICML*, 4120–4129.