AUTOSGM: A UNIFIED LOWPASS REGULARIZATION FRAMEWORK FOR ACCELERATED LEARNING

Oluwasegun A. Somefun, Stefan Lee, V John Mathews

School of Electrical Engineering and Computer Science, Oregon State University. Corvallis, OR 97331 USA.

ABSTRACT

This paper unifies commonly used accelerated stochastic gradient methods (Polyak's Heavy Ball, Nesterov's Accelerated Gradient and Adaptive Moment Estimation (Adam)) as specific cases of a general lowpass regularized learning framework, the Automatic Stochastic Gradient Method (AutoSGM). For AutoSGM, we derive an optimal iteration-dependent learning rate function and realize an approximation. Adam is also an approximation of this optimal approach that replaces the iteration-dependent learning-rate with a constant. Empirical results on deep neural networks comparing the learning behavior of AutoSGM equipped with this iteration-dependent learning-rate algorithm demonstrate fast learning behavior, robustness to the initial choice of the learning rate, and can tune an initial constant learning-rate in applications where a good constant learning rate approximation is unknown.

Index Terms— stochastic gradient method, accelerated learning, learning algorithms, optimization, deep learning.

1. INTRODUCTION

Learning of parameters in the practice of deep learning is mostly driven by stochastic gradient methods (SGMs) [1, 2, 3]. Generally, the SGM updates the parameter w_t using the state equation

$$\mathrm{w}_t = \mathrm{w}_{t-1} - \alpha_t \, \mathrm{g}_t \, . \tag{1}$$

It performs an iterative optimization (typically minimization) of some differentiable scalar objective function $f(\mathrm{w})$, where $t = 0, 1, 2, \ldots, k, \ldots, k \in \mathbb{N}$ is the current iteration index, w_t is the new state value of the parameter, its input g_t is a first-order gradient of the objective function with respect to the current state of the parameter w_{t-1}, and α_t is a possibly iteration-varying step-size.

Several authors have provided convergence analyses for the SGM and explanations of its effectiveness in large-scale learning [4, 5]. Despite this, fine-tuning α_t in SGMs is still an art [6, 7, 8]. Consequently, researchers continue to search for variants or alternative algorithms with better learning characteristics or ease of tuning [9, 10]. Although the question of optimal learning algorithms was treated in [11], relatively recent works have continued to posit that stochastic gradient learning theory and practice needs more explanation and unification, especially for large-scale non-convex optimization [12, 13]. Properly tuned momentum-based learning algorithms, also known as accelerated SGMs, often exhibit faster convergence than the baseline algorithm in (1) [14, 15, 16, 17]. Relevant to this work are three classic and mainstream accelerated methods: Polyak's Heavy Ball (PHB) [18, 19], Nesterov's Accelerated Gradient (NAG) [20], and Adaptive Moment Estimation (Adam) [21]. These algorithms are typically treated as distinctly separate methods. For example, popular deep learning frameworks like Torch, provide separate implementations for each of these methods.

This work was funded in part by NSF grants 1901492 and 1901236.

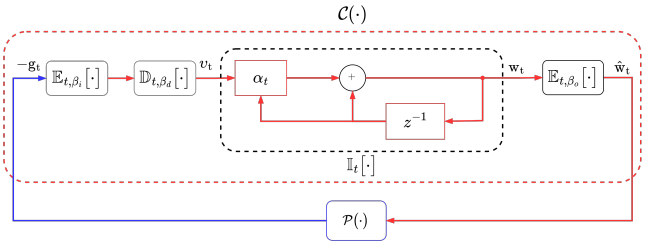

Fig. 1: A block diagram for the AutoSGM framework. The block denoted $\mathcal{P}(\cdot)$ is a first-order gradient generating system (blue lines) and z^{-1} represents a one-step delay operator. The dash-dotted area is the AutoSGM function $\mathcal{C}(\cdot)$ which contains a digital integrator $\mathbb{I}_t$ and a step-size α_t, . The input to $\mathbb{I}_t$ is a gradient g_t processed by a first-order lowpass filter $\mathbb{E}_{t,\beta_i}$ and a digital proportional plus differentiator $\mathbb{D}_{t,\beta_d}$ in series. $0 \leq \beta_i, \beta_o < 1, \beta_d \geq 0$ are digital filter parameters.

In this paper, we show that these accelerated algorithms, despite having different implementations, can be characterized as special cases of a general structure, the automatic stochastic gradient method (AutoSGM) displayed in Fig. 1. The descriptor "automatic" in the name will be made clearer later in the paper, when we derive an optimal step-size that mitigates the need for manually tuning the learning algorithm.

Let $\mathrm{m}_t = \mathbb{E}_{t,\beta}\{\mathrm{b}_t\}$ represent a first-order digital lowpass filter, commonly implemented as $\mathrm{m}_t = \beta \, \mathrm{m}_{t-1} + \eta \, \mathrm{b}_t$, where m_t is the lowpass system's output when b_t is its input, with stability and behaviour controlled by the pole location $0 \leq \beta < 1$, and the gain η of the system. Further, let $\mathrm{m}_t = \mathbb{D}_{t,\beta_d}\{\mathrm{b}_t\}$ represent a first-order digital proportional plus differentiator system, commonly implemented as $\mathrm{m}_t = \mathrm{b}_t + \beta_d \, (\mathrm{b}_t - \mathrm{b}_{t-1})$ [22], where m_t is the system's output when b_t is its input, with a filter constant $\beta_d \geq 0$. Then the AutoSGM weight update function is

$$\mathrm{w}_t = \mathrm{w}_{t-1} - \alpha_t \, \upsilon_t \tag{2}$$

where υ_t represents a smoothed gradient signal. υ_t is obtained as $\mathbb{D}_{t,\beta_d}\{\mathbb{E}_{t,\beta_i}\{\mathrm{g}_t\}\}$ with input g_t. Without any loss of generality, the order of the cascade combination of $\mathbb{E}_{t,\beta_i}$ and $\mathbb{D}_{t,\beta_d}$ can be reversed. Finally, AutoSGM's output w_t is optionally further smoothed by a lowpass filter to obtain $\hat{\mathrm{w}}_t = \mathbb{E}_{t,\beta_o}\{\mathrm{w}_t\}$ [23]. All vector operations with (2) are performed on a sample-by-sample basis.

The AutoSGM algorithm along with PHB, NAG and Adam are summarized in Table 1 below. We observe that AutoSGM with $\beta_d = 0$, $\eta_i = 1$, and constant α_t is PHB. AutoSGM becomes NAG for $\beta_d = \beta_i$, $\eta_i = 1$, and a constant α_t. Using an unbiased lowpass

Table 1: Three popular accelerated learning algorithms as special cases of AutoSGM.

Algorithm Name	Parameter update (one-line difference equation)
AutoSGM	$\mathrm{w}_{t+1} = \mathrm{w}_t + \beta_i\,(\mathrm{w}_t - \mathrm{w}_{t-1}) - \alpha_t\,\eta_i\Big(\mathrm{g}_t + \beta_d(\mathrm{g}_t - \mathrm{g}_{t-1})\Big)$
PHB	$\mathrm{w}_{t+1} = \mathrm{w}_t + \beta_i\,(\mathrm{w}_t - \mathrm{w}_{t-1}) - \alpha\,\mathrm{g}_t$
NAG	$\mathrm{w}_{t+1} = \mathrm{w}_t + \beta_i\,(\mathrm{w}_t - \mathrm{w}_{t-1}) - \alpha\Big(\mathrm{g}_t + \beta_i(\mathrm{g}_t - \mathrm{g}_{t-1})\Big)$
Adam	$\mathrm{w}_{t+1} = \mathrm{w}_t + \beta_i\,(\mathrm{w}_t - \mathrm{w}_{t-1}) - \alpha_t\eta_i\,\mathrm{g}_t$, where $\alpha_t \coloneqq \hat{\alpha}_0/\sqrt{\mathbb{E}_{t,\beta_e}\{\mathrm{g}_t^2\}}$

Notes: $\hat{\alpha}_0$ is a constant learning rate, β_e is the lowpass parameter used for moment estimation.

filter implementation $\beta_d = 0$, with $\eta_i = 1 - \beta_i$, and a moment-estimation step-size algorithm for α_t, AutoSGM becomes Adam. When η_i is set as 1, independent of β_i, as in common implementations of PHB and NAG, instability in the form of limit-cycle convergence has been demonstrated in [24] for certain strongly-convex functions. This implies that the lowpass filter implementation matters.

The main contributions of this paper are as follows: (1) PHB, NAG, and Adam are special cases of AutoSGM. (2) We derive an iteration-dependent optimal learning-rate function for AutoSGM and realize an approximate implementation. This approach is robust to the choice of initial learning rate. (3) We show that the lowpass filtering of the gradient function is approximately equivalent to a lowpass regularization of the objective function f during learning. This improves the odds that parameters of the learning algorithm will converge to a better local optimum of the objective function's surface. (4) We empirically compare the performance of Adam with a constant learning-rate and AutoSGM equipped with the iteration-dependent learning-rate algorithm on some commonly used neural network architectures. Our findings show that without compromising performance, the iteration-dependent learning-rate algorithm automatically tunes an initial constant learning rate. This is a desirable property in many applications where good, constant learning-rates are difficult to find.

2. REFINEMENTS AND ANALYSIS OF AUTOSGM

We derive an iteration-dependent learning rate function and introduce several properties of the AutoSGM in this section. Because of page limitation, only outlines of proofs of theorems are provided.

In the gradient generating system $\mathcal{P}$, we consider the empirical minimization of a scalar-valued function $f\big(\mathrm{w}\big)$ given by

$$\min_{\mathrm{w}} f(\mathrm{w}) = \frac{1}{n}\sum_{i=1}^{n} \ell_i(\mathrm{w}) + \mu\,\|\mathrm{w}\|^2 \tag{3}$$

We assume that f has a log-likelihood interpretation and is a composition of one or more smooth (differentiable) functions, denoted $\ell_i(\mathrm{w})$, averaged over a training set of n examples from a data generating function $\mathcal{S}_n = \{(x_i, y_i); i = 1, 2, \ldots, n\}$. It is common to impose a weight constraint that forces the weights to be as small as possible. The regularization constant μ in (3) attempts to achieve this objective.

2.1. An Iteration-Dependent Learning Rate Function

Let $\mathcal{H}_r$ be the Hilbert space of random variables defined on a probability space with an unknown probability measure $\mathbb{P}$. For any $u, z \in \mathcal{H}_r$ we define inner-product $\langle u, z\rangle \coloneqq \mathbb{E}[uz]$, where $\mathbb{E}[\cdot]$ is the statistical expectation. For each $\mathrm{w}, \mathrm{g} \in \mathcal{H}_r$, the gradient of a log-likelihood objective function f has an expected value of zero at all locations on its surface [25]. Since $\mathbb{E}[\mathrm{g}_t] = 0$, it follows immediately that $\mathbb{E}[\upsilon_t] = 0$.

We assume that AutoSGM minimizes the mean-squared error risk function $\mathcal{R}_t \coloneqq \mathbb{E}[\varepsilon_{\alpha_t}^2]$, where ε_{α_t} is the per-parameter estimation error given by $\varepsilon_{\alpha_t} \coloneqq \mathrm{w}_t - \mathrm{w}^\star$, and $\mathrm{w}^\star$ is the unknown optimum solution to w_t. The optimization problem is then to discover the value of α_t that minimizes the risk $\mathcal{R}_t$.

By differentiating $\mathcal{R}_t$ with respect to α_t and setting it to zero, we get: $\mathbb{E}[\varepsilon_{\alpha_t}\upsilon_t] = \mathbb{E}[\big(\varepsilon_{\alpha_{t-1}} - \alpha_t\upsilon_t\big)\,\upsilon_t] = 0$. Solving for α_t gives

$$\alpha_t = \frac{\mathbb{E}[\varepsilon_{\alpha_{t-1}}\,\upsilon_t]}{\mathbb{E}[\upsilon_t^2]} \tag{4}$$

2.2. Stability

We now show, under certain conditions, that the AutoSGM system equipped with (4) is uniformly stable.

Theorem 1. *For $t \geq 0$, α_t in (4), and risk $\mathcal{R}$ for each tuple* (w, g), *the system is uniformly stable.*

Theorem 1 can be obtained by noting that the error state equation is $\varepsilon_{\alpha_t} = \varepsilon_{\alpha_{t-1}} - \alpha_t\upsilon_t$. Substitute in (4) for α_t and expand $\mathcal{R}_t \coloneqq \mathbb{E}[\varepsilon_{\alpha_t}^2]$ to get, $\mathcal{R}_t = \mathcal{R}_{t-1}(1 - \varsigma_t{}^2)$ where $\varsigma_t = \mathbb{E}[\varepsilon_{\alpha_{t-1}}\,\upsilon_t]/\sqrt{\mathbb{E}[\varepsilon_{\alpha_{t-1}}^2]\,\mathbb{E}[\upsilon_t^2]}$, satsifies the bound $|\varsigma_t| \leq 1$. Iterating the state transition for the risk t times and bounding each ς_t with a finite constant $\hat{\varsigma} \in [0, 1)$, we obtain the bound $\mathcal{R}_t \leq \lambda^t\,\mathcal{R}_0$, where $\lambda = 1 - \hat{\varsigma}^2$. For all t, if $\hat{\varsigma} = 0$, then $\lambda = 1$, the system has converged to a local minima. Therefore, the system is uniformly stable.

2.3. A Realizable Approximation for the Optimal Learning Rate

In practice, we do not have access to the optimal α_t in (4) and w^* in $\varepsilon_{\alpha_{t-1}}$ is not known *a priori*. Without any loss of generality, we can rewrite (4) in normalized form:

$$\alpha_t = \hat{\alpha}_t/\sqrt{\mathbb{E}[\upsilon_t^2]} \tag{5}$$

where $\hat{\alpha}_t \coloneqq \mathbb{E}[\varepsilon_{\alpha_{t-1}}\,\bar{\upsilon}_t]$ is an iteration-dependent correlation function that defines the optimal learning-rate, and $\bar{\upsilon}_t = \upsilon_t/\sqrt{\mathbb{E}[\upsilon_t^2]}$ is a normalized gradient. To obtain a practical iteration-dependent realization for (5), we approximate the output of the two statistical expectations $\mathbb{E}[\,\cdot\,]$ in (5), $\mathbb{E}[\upsilon_t^2]$ and $\mathbb{E}[\varepsilon_{\alpha_{t-1}}\,\bar{\upsilon}_t]$ with outputs of an appropriate lowpass filters of the form $\mathbb{E}_{t,\beta}\{\,\cdot\,\}$. We can use Chernoff inequality [26] and ergodic assumptions on the input to an exponentially-weighted lowpass filter to provide a probably approximately correct bound on how close we expect the output of the lowpass filter $\mathbb{E}_{t,\beta}\{u\}$ to deviate from $\mathbb{E}[u]$.

Algorithm 1 An AutoSGM implementation with iteration-dependent learning-rate function.

State: w, **Input:** g, **Output:** $\hat{\mathrm{w}}$
Params: $0 < \hat{\alpha}_0 < 1$ (need this for initialization only), $0 \leq \beta_i < 1, \beta_o \cong 0, \beta_d \geq 0, 0.9 < \beta_e < 1, \hat{\epsilon} > 0$.

$t \leftarrow 0, k > 0$
for $t = 1 : k$ **do**
  $\upsilon_t \leftarrow \mathbb{D}_{t,\beta_d}\{\mathbb{E}_{t,\beta_i}\{\mathrm{g}_t\}\}$ (*smoothing*)
  $\mathrm{s}_t \leftarrow \mathbb{E}_{t,\beta_e}\{\mathbb{D}_{t,\beta_d}\{\mathrm{g}_t\}^2\}$ (*averaging*)
  $\bar{\upsilon}_t \leftarrow \upsilon_t/(\sqrt{\mathrm{s}_t} + \hat{\epsilon})$ (*normalization*)
  $\hat{\alpha}_t \leftarrow |\mathbb{E}_{t,\beta_e}\{\mathrm{w}_{t-1}\,\bar{\upsilon}_t\}|$ (*averaging*)
  $\mathrm{w}_t \leftarrow \mathrm{w}_{t-1} - \hat{\alpha}_t\,\bar{\upsilon}_t$ (*integration*)
  $\hat{\mathrm{w}}_t \leftarrow \mathbb{E}_{t,\beta_o}\{\mathrm{w}_t\}$ (*smoothing*)

Theorem 2. *For $0 \leq \beta < 1$, the steady-state value of the probability that $\mathbb{E}_{t,\beta}\{\cdot\}$ deviates within a small error $\epsilon > 0$ from the true function $\mathbb{E}[\cdot]$ is*

$$\lim_{t\to\infty} \mathbb{P}\big(\xi_t \geq \epsilon\big) \leq \exp\Big(-\frac{\epsilon^2\, c_\infty}{2\,\mathcal{B}}\Big) \tag{6}$$

where $\xi_t \coloneqq \mathbb{E}_{t,\beta}\{u\} - \mathbb{E}\{u\}$, $\mathcal{B} \coloneqq \mathbb{E}[(u - \mathbb{E}\{u\})^2]$, and $c_\infty \coloneqq (1+\beta)/(1-\beta)$.

Remark 1. Since c_∞ is a function of β, then this result intuitively suggests that for averaging with $\mathbb{E}_{t,\beta}\{\,\cdot\,\}$, we should choose β close to 1 to keep the probability of deviation low. A common setting in practice is to choose $0.9 < \beta < 1$.

To avoid singularities in the update equation, we approximate the smooth normalized gradient as $\bar{\upsilon}_t = \upsilon_t/\big(\sqrt{\mathbb{E}_{t,\beta_e}\{\upsilon_t{}^2\}} + \hat{\epsilon}\big)$ where $\hat{\epsilon} > 0$ is a real valued constant.

We still need to find a realizable iteration-dependent approximation for $\hat{\alpha}_t \coloneqq \mathbb{E}[\varepsilon_{\alpha_{t-1}}\,\bar{\upsilon}_t]$ which contains w^* in $\varepsilon_{\alpha_{t-1}}$. The final step to acheive this, is to ignore the w^*. That is, we replace $\varepsilon_{\alpha_{t-1}}$ with w_{t-1} leading to $\mathbb{E}[\mathrm{w}_{t-1}\,\bar{\upsilon}_t]$.This approximation becomes more accurate as $t \to \infty$, since in the steady state, we have $\mathbb{E}\big[\mathrm{w}^*\,\bar{\upsilon}_t\big] = 0$. The correlation function $\hat{\alpha}_t \coloneqq \mathbb{E}[\varepsilon_{\alpha_{t-1}}\,\bar{\upsilon}_t]$ is non-negative, but its approximation as $\mathbb{E}_{t,\beta_e}\{\mathrm{w}_{t-1}\,\bar{\upsilon}_t\}$ may not be. When this approximation becomes negative, we may set it to zero and not update the parameters. Empirically, we have found that replacing the negative estimates of the approximation $\hat{\alpha}_t \approx \mathbb{E}_{t,\beta_e}\{\mathrm{w}_{t-1}\,\bar{\upsilon}_t\}$ with absolute values, and restarting the filter when $\hat{\alpha}_t$ drops below a small threshold provided better overall learning behavior. Experimental results reported in Section 3 were obtained using this approach (we used 10^{-5} in our experiments). The AutoSGM equipped with this iteration-dependent learning-rate algorithm is described by the pseudo-code in Algorithm 1.

2.4. Connection to Adam

AutoSGM becomes Adam if in (5) the gradient g_t is not smoothed, $\beta_d = 0$, and the correlation function $\hat{\alpha}_t$ is approximated as a single constant $\hat{\alpha}_0$. Thus we get Adam, if (5) is replaced by

$$\alpha_t = \hat{\alpha}_0/\Big(\sqrt{\mathbb{E}_{t,\beta_e}\{\mathrm{g}_t^2\}} + \hat{\epsilon}\Big) \tag{7}$$

2.5. Lowpass Regularization

Assuming slow evolution of w_t, we can prove the following.

Theorem 3. *Lowpass filtering the gradient g is approximately equivalent to lowpass smoothing the surface of the objective function f.*

Theorem 3 can be proved by recognizing that lowpass filtering involves convolution of the input with the filter's unit impulse response signal h. As $t \to \infty$, for slowly varying w_t, we have:

$$\upsilon_t = \sum_{j=0}^{t-1} h_j \frac{\partial f(\mathrm{w}_{t-1-j})}{\partial \mathrm{w}_{t-1-j}} \approx \frac{\partial}{\partial \mathrm{w}_{t-1}} \sum_{j=0}^{t-1} h_j\, f(\mathrm{w}_{t-1-j}) \tag{8}$$

Remark 2. Theorem 3 suggests that AutoSGM algorithms are lowpass regularizers of the objective function in the gradient generating system. This implies that, at each learning iteration, an AutoSGM algorithm with an appropriate $\beta_i > 0$ implicity solves a smoother, unbiased and bounded local approximation of f, enabling convergence of the optimizer to better local minima of the loss surface [17].

2.6. Local Convergence Behavior

Assume that f is smooth with a Lipschitz constant L, *i.e.*, $\|\mathrm{g}_t - \mathrm{g}^*\| \leq L\,\|\mathrm{w}_t - \mathrm{w}^*\|$, and that it satisfies the Polyak-Lojasiewicz (PL) inequality $\|\mathrm{g}_t\|^2 \geq 2\iota\big(f_t - f^\star\big)$ [27]. In this case, we say f is $L-$smooth and $\iota-$PL where $0 < \iota < L$. Define $\varepsilon_t \coloneqq \mathrm{w}_t - \mathrm{w}^*$ and note that $\varepsilon_t - \varepsilon_{t-1} = \mathrm{w}_t - \mathrm{w}_{t-1}$. The behavior of f near the optimum point w^* can be locally approximated with a candidate Lyapunov function $\hat{f}$ [28, 29], such as $\hat{f}_t = \frac{1}{2}\varepsilon_t^T\,\mathbf{\Sigma}\,\varepsilon_t$, where $\mathbf{\Sigma} \in \mathbb{R}^{d\times d}$ is a matrix dependent on the finite training data distribution $\mathcal{S}_n$ of the gradient-generating system $\mathcal{P}$ such that $\iota\mathbf{I} \leq \mathbf{\Sigma} \leq L\mathbf{I}$ and $\kappa = L/\iota \geq 1$ is the condition number of $\hat{f}$. For this approximation, the gradient becomes $\mathrm{g}_t \coloneqq \mathbf{\Sigma}\,\varepsilon_t$, and $(\mathrm{g}_t - \mathrm{g}_{t-1}) = \mathbf{\Sigma}\,(\mathrm{w}_t - \mathrm{w}_{t-1})$. For $0 < \beta_i < 1$, subtract w^* from both sides of the parameter update for AutoSGM in Table 1 to obtain $\varepsilon_{t+1} = (\mathbf{I} - \alpha_t\eta_i\mathbf{\Sigma})\,\varepsilon_t + (\beta_i - \alpha_t\eta_i\beta_d\mathbf{\Sigma})\,(\varepsilon_t - \varepsilon_{t-1})$. With this, we can state the following theorem.

Theorem 4. *For any small $\epsilon > 0$, the worst-case number of iterations for an AutoSGM sequence $\{\mathrm{w}_t\}$ to converge to a local optimum w^* is $\tau \sim O\big(\sqrt{\kappa}\log\big(\epsilon^{-1}\big)\big)$.*

To prove Theorem 4, let $\Phi_t(\mathbf{\Sigma})$ be a polynomial function of degree t dependent on $\mathbf{\Sigma}, \alpha_t, \beta_i, \beta_d, \eta_i$, with $\Phi_0(\mathbf{\Sigma}) = \mathbf{I}$. Now assume $\varepsilon_s = \Phi_s(\mathbf{\Sigma})\,\varepsilon_0$ for $0 \leq s \leq t$. It follows immediately that $\varepsilon_{t+1} = \Phi_{t+1}(\mathbf{\Sigma})\,\varepsilon_0$, where $\Phi_{t+1}(\mathbf{\Sigma}) = \big[(\mathbf{I} - \alpha_t\eta_i\mathbf{\Sigma})\,\Phi_t(\mathbf{\Sigma}) + \big(\Phi_t(\mathbf{\Sigma}) - \Phi_{t-1}(\mathbf{\Sigma})\big)(\beta_i - \alpha_t\eta_i\beta_d\mathbf{\Sigma})\big]$. Then, it follows by induction that $\varepsilon_t = \Phi_t(\mathbf{\Sigma})\,\varepsilon_0, \forall\, t$. Next, take the norm of $\varepsilon_t = \Phi_t(\mathbf{\Sigma})\,\varepsilon_0$. Bound with the Cauchy-Schwarz inequality to obtain $\|\varepsilon_t\| \leq \|\Phi_t(\mathbf{\Sigma})\|\,\|\varepsilon_0\|$. By eigendecomposition, $\mathbf{\Sigma} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\intercal$, where $\mathbf{Q}$ is an orthonormal matrix and $\mathbf{\Lambda}$ is a diagonal matrix with diagonal elements on the interval $\sigma \in [\iota, L]$. Using this, the error bound becomes $\|\varepsilon_t\| \leq \max_\sigma|\Phi_t(\sigma)|\,\|\varepsilon_0\|$. The worst-case convergence rate bound problem then becomes a min-max problem: $\|\varepsilon_t\| \leq \min_{\Phi_t}\max_\sigma|\Phi_t(\sigma)|\,\|\varepsilon_0\|$, *i.e.*, finding a $\Phi_t(\sigma)$ with the smallest maximum absolute value over the $\sigma \in [\iota, L]$ interval. This problem is known to be minimized by a Chebyshev polynomial [30, 31]. Using this Chebyshev polynomial leads to $\|\varepsilon_t\| \leq \big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\big)^t\|\varepsilon_0\|$. Then for any small $\epsilon > 0$, such that $\big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\big)^t \leq \epsilon$, as t increases, the worst-case number of iterations τ can be found.

3. EXPERIMENTS

Most empirical comparisons among the special cases of AutoSGM are biased, probably because many aspects of the neural network

architectures being used were evolved to be well suited to Adam. Also, the choice of hyperparameters $(\alpha_t, \beta_i, \beta_d, \beta_o, \beta_e)$ and the filter implementations used in these special cases might differ. Effort required to scan through the search space of all hyperparameters can be costly and mostly unjustified, especially if they can be turned off or manually set to some known good fixed value [6, 17]. In this section, while keeping all other hyperparameters in AutoSGM constant, with $\beta_o = 0, \beta_d = 0$, we compare the learning behavior of Algorithm 1 and Adam (7).

We consider a common image classification task on the CIFAR-10 dataset [32], with two different deep learning models: LeNet (a convolutional neural network without normalization layers), and ResNet50 (a residually connected convolutional neural network (CNN) with 50 layers), trained on four Nvidia Quadro RTX GPUs on a shared HPC cluster. Training for 200 epochs, and averaging over five runs with a weight-decay constant of 10^{-5}, we compare training loss and test accuracy performance across different batch-sizes $\{128, 256, 1024\}$ and initial constant learning-rates $\{3 \times 10^{-3}, 10^{-3}, 10^{-4}\}$. Due to page constraints, other performance plots are reported in our code repository[1]. Training loss distribution is shown and discussed for LeNet in Figure 2 and corresponding training curves are shown in Figure 3. For ResNet50, the test-accuracy distribution is shown and discussed in Figure 4 and corresponding training curves are shown in Figure 5. Generally, we find that AutoSGM with the iteration-dependent learning rate, appears to be on par with Adam, and can help tune initial constant learning rates without compromising performance as illustrated in Figure 6.

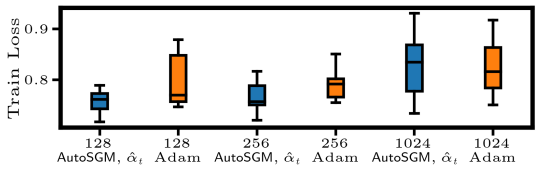

Fig. 2: AutoSGM, with $\hat{\alpha}_t$, (blue box) and Adam, $\hat{\alpha}_0$, (orange box). A summary of the training loss distributions for the three different initial learning rates $\{3 \times 10^{-3}, 10^{-3}, 10^{-4}\}$ for batch-size of $\{128, 256, \text{ and } 1024\}$ for 5 training runs. Here, while the spread is generally comparable, average performance of the iteration-dependent $\hat{\alpha}_t$ appears to be better for the smaller batch-sizes, while, Adam appears to provide a slightly better average for the larger batch-size of 1024. Model: LeNet.

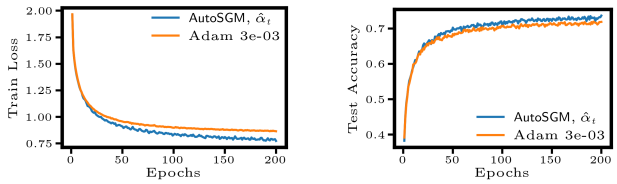

Fig. 3: Average training loss (left) and test accuracy (right) over 5 training runs for LeNet on CIFAR10, with a batch-size of 128 and initial learning-rate of $\hat{\alpha}_0 = 3 \times 10^{-3}$. AutoSGM, with $\hat{\alpha}_t$, (blue line) and Adam, $\hat{\alpha}_0$, (orange line).

[1] https://github.com/somefunAgba/autosgm

4. CONCLUSIONS

AutoSGM is a general framework for learning algorithms that have a lowpass regularized structure as illustrated in Figure 1. Commonly used accelerated SGMs such as PHB, NAG, and Adam are special cases. Choosing the filter parameters $\beta_i, \beta_o, \beta_d$ in the range [0,1) guarantees stability of the algorithm. AutoSGM may lead to the development of new learning rate algorithms for setting α_t. The iteration-dependent learning rate algoirthm tunes an initial constant learning-rate using the current state of the gradient-generating system, and leads to acceptably good solutions. In some applications, the process of fine-tuning a constant learning-rate might be difficult, costly or unsafe and involve much trial and error. The AutoSGM algorithm framework with the iteration-dependent learning-rate may simplify the tuning process in such cases. Improving on this iteration-dependent learning rate realization is a potential topic for future work.

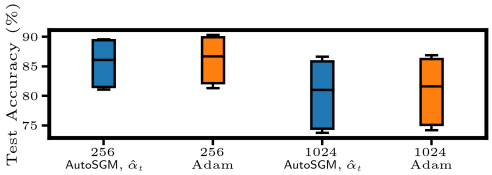

Fig. 4: AutoSGM, with $\hat{\alpha}_t$, (blue box) and Adam, $\hat{\alpha}_0$, (orange box). A summary of the test accuracy distributions for the learning rates $\{10^{-3}, 10^{-4}\}$ over 5 training runs for batch-size of $\{256, 1024\}$. Training for 200 epochs, the spread of both algorithms appears to be similar, with Adam providing slightly better average test-accuracies. Model: ResNet50, a 50-layer CNN.

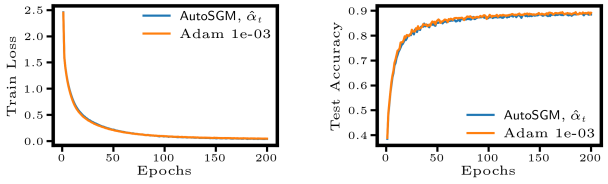

Fig. 5: Average Training loss (left), Test accuracy (right) over 5 training runs for ResNet50 on CIFAR10, with a batch-size of 256 and initial learning-rate of 10^{-3}. AutoSGM, with $\hat{\alpha}_t$, (blue line) and Adam, $\hat{\alpha}_0$, (orange line).

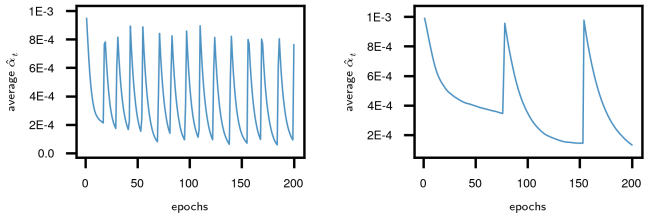

Fig. 6: Evolution of $\hat{\alpha}_t$ for a batch-size of 256 (left plot), and 1024 (right plot) and an initial learning rate of 10^{-3} for a ResNet50 weight layer trained on the CIFAR10 dataset. $\hat{\alpha}_t$ depends on the gradient which depends on the batch-size. Due to the restarts discussed in section 2.3, it evolves cyclically.

5. REFERENCES

[1] H. Robbins and S. Monro, "A stochastic approximation method," *The Annals of Mathematical Statistics*, vol. 22, pp. 400–407, Sept. 1951.

[2] B. Widrow, "Pattern Recognition and Adaptive Control," *IEEE Transactions on Applications and Industry*, vol. 83, pp. 269–277, Sept. 1964.

[3] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Review*, vol. 60, no. 2, pp. 223–311, 2018.

[4] Y. Lei, T. Hu, G. Li, and K. Tang, "Stochastic gradient descent for nonconvex learning without bounded gradient assumptions," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, pp. 4394–4400, Oct. 2020.

[5] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT*, (Paris, France), pp. 177–186, Springer, 2010.

[6] V. Godbole, G. E. Dahl, J. Gilmer, C. J. Shallue, and Z. Nado, "Deep learning tuning playbook." http://github.com/google-research/tuning_playbook, 2023.

[7] T. S. Prabhu, F. Mai, T. Vogels, M. Jaggi, and F. Fleuret, "Optimizer benchmarking needs to account for hyperparameter tuning," in *Proceedings of the 37th International Conference on Machine Learning*, vol. 119, (Vienna, Austria), pp. 9036–9045, JMLR.org, July 2020.

[8] K. Nar and S. Sastry, "Step size matters in deep learning," in *Advances in Neural Information Processing Systems*, vol. 31, (Montréal, Canada), Curran Associates, Inc., 2018.

[9] P. Chiang, R. Ni, D. Y. Miller, A. Bansal, J. Geiping, M. Goldblum, and T. Goldstein, "Loss landscapes are all you need: Neural network generalization can be explained without the implicit bias of gradient descent," in *The Eleventh International Conference on Learning Representations*, (Kigali, Rwanda), Feb. 2023.

[10] F. Orabona and T. Tommasi, "Training deep networks without learning rates through coin betting," in *Advances in Neural Information Processing Systems*, vol. 30, (Long Beach, California, USA.), Curran Associates, Inc., 2017.

[11] Y. Z. Tsypkin, *Adaptation and Learning in Automatic Systems*. New York: Academic Press, 1971.

[12] A. Khaled and P. Richtárik, "Better theory for SGD in the nonconvex world," *Transactions on Machine Learning Research*, Mar. 2023.

[13] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Communications of the ACM*, vol. 64, pp. 107–115, Feb. 2021.

[14] B. T. Polyak, "Accelerated gradient methods: History and properties," in *7th International Conference on Control and Optimization with Industrial Applications*, vol. 1, (Baku, Azerbaijan), pp. 23–25, IAM, 2020.

[15] L. Lessard and P. Seiler, "Direct synthesis of iterative algorithms with bounds on achievable worst-case convergence rate," in *2020 American Control Conference*, (Denver), pp. 119–125, IEEE, 2020.

[16] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on Machine Learning*, (Atlanta, Georgia, USA), pp. 1139–1147, May 2013.

[17] Y. Bengio, "Practical Recommendations for Gradient-Based Training of Deep Architectures," in *Neural Networks: Tricks of the Trade: Second Edition*, pp. 437–478, Berlin, Heidelberg: Springer, 2012.

[18] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Computational Mathematics and Mathematical Physics*, vol. 4, pp. 1–17, Jan. 1964.

[19] B. T. Polyak, "The conjugate gradient method in extremal problems," *USSR Computational Mathematics and Mathematical Physics*, vol. 9, pp. 94–112, Jan. 1969.

[20] Y. Nesterov, "A method for solving the convex programming problem with convergence rate $O(1/k^2)$," *Doklady Akademii Nauk*, vol. 269, pp. 543–547, 1983.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR* (Y. Bengio and Y. LeCun, eds.), (San Diego, CA, USA,), 2015.

[22] G. F. Franklin, J. D. Powell, and A. Emami-Naeini, *Feedback Control of Dynamic Systems*. USA: Pearson, 8th ed., 2019.

[23] L. Bottou, "Stochastic gradient descent tricks," in *Neural Networks: Tricks of the Trade*, pp. 421–436, Springer, 2012.

[24] L. Lessard, B. Recht, and A. Packard, "Analysis and design of optimization algorithms via integral quadratic constraints," *SIAM Journal on Optimization*, vol. 26, pp. 57–95, Jan. 2016.

[25] H. L. Van Trees, K. L. Bell, and Z. Tian, *Detection Estimation and Modulation Theory, Detection, Estimation, and Filtering Theory, Part I*. New Jersey, USA: Wiley, 2nd ed., 2013.

[26] H. Chernoff, "A note on an inequality involving the normal distribution," *The Annals of Probability*, vol. 9, pp. 533–535, June 1981.

[27] H. Karimi, J. Nutini, and M. Schmidt, "Linear convergence of gradient and proximal-gradient methods under the Polyak-Lojasiewicz condition," in *Machine Learning and Knowledge Discovery in Databases*, (Cham), pp. 795–811, Springer, 2016.

[28] K. Najim, E. Ikonen, and A.-K. Daoud, "Analysis of Recursive Algorithms," in *Stochastic Processes*, pp. 223–314, Oxford: Kogan Page Science, Jan. 2004.

[29] M. S. Fadali and A. Visioli, "Elements of nonlinear digital control systems," in *Digital Control Engineering (Third Edition)* (M. S. Fadali and A. Visioli, eds.), pp. 507–565, Academic Press, Jan. 2020.

[30] B. Goujaud, D. Scieur, A. Dieuleveut, A. B. Taylor, and F. Pedregosa, "Super-acceleration with cyclical step-sizes," in *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, (Virtual), pp. 3028–3065, PMLR, 2022.

[31] D. A. Flanders and G. Shortley, "Numerical determination of fundamental modes," *Journal of Applied Physics*, vol. 21, pp. 1326–1332, Apr. 2004.

[32] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images." https://www.cs.toronto.edu/~kriz/cifar.html, 2009.