OPT2023: 15th Annual Workshop on Optimization for Machine Learning

On the Interplay Between Stepsize Tuning and Progressive Sharpening

Vincent Roulet VROULET@GOOGLE.COM
Atish Agarwala THETISH@GOOGLE.COM
Fabian Pedregosa PEDREGOSA@GOOGLE.COM
Google DeepMind

Abstract

Recent empirical work has revealed an intriguing property of deep learning models by which the sharpness (largest eigenvalue of the Hessian) increases throughout optimization until it stabilizes around a critical value at which the optimizer operates at the edge of stability, given a *fixed* stepsize [Cohen et al., 2022b]. We investigate empirically how the sharpness evolves when using stepsize-tuners, the Armijo linesearch and Polyak stepsizes, that adapt the stepsize along the iterations to local quantities such as, implicitly, the sharpness itself. We find that the surprisingly poor performance of a classical Armijo linesearch may be well explained by its tendency to ever-increase the sharpness of the objective in the full or large batch regimes. On the other hand, we observe that Polyak stepsizes operate generally at the edge of stability or even slightly beyond, while outperforming its Armijo and constant stepsizes counterparts. We conclude with an analysis that suggests unlocking stepsize tuners requires an understanding of the joint dynamics of the step size and the sharpness.

1. Introduction

Selecting a good learning rate is a time-consuming part of practical machine learning. To alleviate this, a number of automatic stepsize tuners have been proposed in the literature. Two of the most common methods are the stochastic Armijo line-search of Vaswani et al. [2019] and the stochastic Polyak stepsize of Berrada et al. [2020], Loizou et al. [2021]. These methods enjoy strong theoretical guarantees that, as we will see, do not always translate into fast empirical performance.

A different line of work has highlighted a number of important phenomena during training of deep neural networks with large learning rates [Foret et al., 2022, Ghorbani et al., 2019, Gilmer et al., 2022, Neyshabur et al., 2017]. In particular, many experiments have shown the tendency for the loss's sharpness (largest Hessian eigenvalue) to increase until it reaches the *edge of stability* (EOS), in which the maximum eigenvalue increases to and stabilizes at $2/\gamma$ for stepsize γ, corresponding to the largest eigenvalue that converges for gradient descent with stepsize γ in a quadratic objective [Cohen et al., 2022a,b, Giladi et al., 2020, Wu et al., 2018].

While there are some studies of sharpness dynamics with preconditioners [Cohen et al., 2022b], little is known about sharpness dynamics with stepsize tuners that condition on local properties of the loss landscape. This work takes a first step to fill this void by providing a systematic study of loss and sharpness dynamics in Armijo and Polyak stepsize tuners.

We find the following. In the *deterministic* (full batch) setting, Armijo performs worse than constant stepsize while Polyak performs well. Armijo exhibits an ever-increasing sharpness while

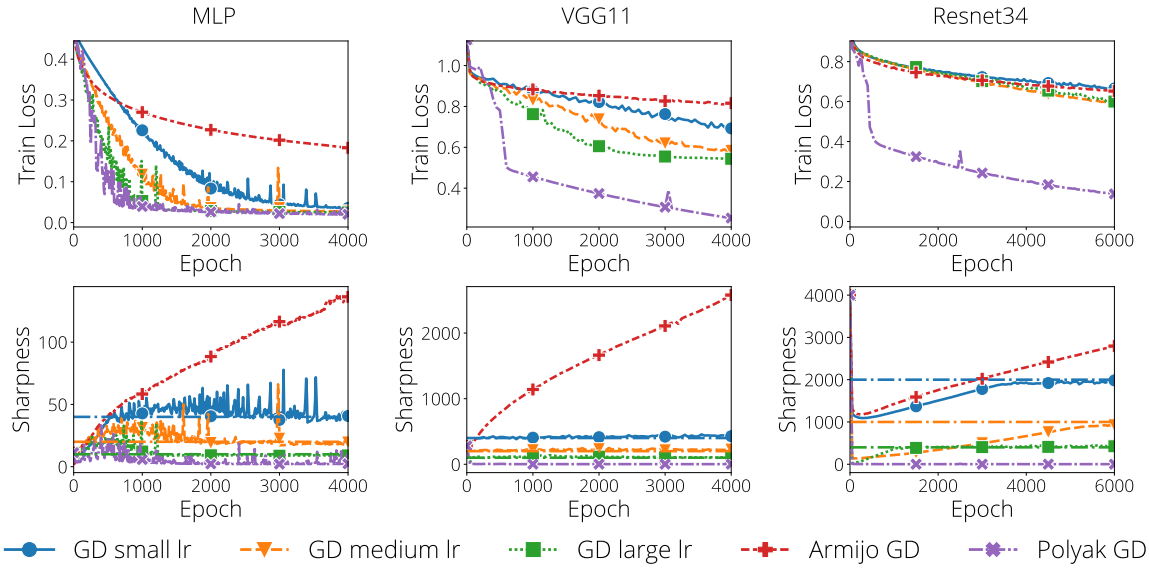

Figure 1: **Interplay between stepsize tuners and sharpness.** We plot the training loss (top) and sharpness (bottom) for various architectures on a subset of CIFAR10, for the different stepsize tuners (Armijo-GD, Polyak-GD) as well as GD with fixed stepsizes γ, all in the full-batch setting. The sharpness of GD stabilizes around the value $2/\gamma$ (dashed line). While Armijo-GD decreases the objective monotonically, the sharpness climbs further above any other method. In contrast, the train loss of Polyak-GD is not monotonically decreasing but the sharpness plateaus at low values.

Polyak operates almost systematically at EOS or above. In the *stochastic* (minibatch) setting, the performance of Armijo depends highly on the mini-batch size, while Polyak performs slightly less well than in the full batch setting and operates mostly again at EOS. Our theoretical insights show that the *joint* dynamics of loss and stepsize tuning itself is necessary to explain these phenomena.

2. Methods

We consider minimizing of the average f of n functions $f_1, \dots, f_n$, representing typically the average loss of a deep network on a set of samples, that is, we aim to solve

$$w^\star \in \arg\min_{w \in \mathbb{R}^p} \left\{ f(w) \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} f_i(w) \right\} . \tag{1}$$

In that purpose, we consider gradient or stochastic gradient descent algorithms of the form

$$w_{t+1} = w_t - \gamma_t \nabla \phi(w_t), \qquad \text{with } \phi = f \text{ (deterministic regime) or } \phi = f_i \text{ (stochastic regime)}\,, \tag{2}$$

for i sampled uniformly at random in $[n]$ in the stochastic regime. Here rather fixing the stepsize (a.k.a. learning rate) γ_t to a constant, we consider stepsize-tuners that automatically adjust γ_t given local information on the objective. We consider the two following stepsize-tuners.

Armijo line-search. The Armijo line-search is a standard method for setting the stepsize of gradient-based optimizers in the deterministic setting [Nocedal and Wright, 1999]. Vaswani et al.

[2019] adapted this classical method to the stochastic regime and provided convergence guarantees for smooth convex functions in the interpolation regime. The Armijo condition consists in selecting the stepsize γ_t to ensure decrease of the objective or its mini-batch counterpart as

$$\phi(w_t - \gamma_t \nabla \phi(w_t)) \leq \phi(w_t) - c\gamma_t \|\nabla \phi(w_t)\|^2, \quad \text{with } \phi = f \text{ (deterministic) or } \phi = f_i \text{ (stochastic)}, \tag{3}$$

for $c > 0$ a hyperparameter of the method, set to 10^{-4} in our experiments as in usual implementations. To ensure that the above is satisfied, one starts from a maximum stepsize $\gamma_{\max}$, set to $\gamma_{\max} = 1$ *for each step* in the experiments, and decreases the stepsize by a constant factor of 0.8 until the above equation is satisfied.

Polyak stepsize. A recent breakthrough in optimization was the realization that the venerable Polyak stepsize, [Polyak, 1969] originally developed for deterministic/full gradient optimization, extends naturally to stochastic optimization [Berrada et al., 2020, Jiang and Stich, 2023, Loizou et al., 2021]. Many variants of this stepsize have been proposed, see, e.g., [Gower et al., 2022, Li et al., 2022]. In this manuscript we focus on the $\text{SPS}_{\max}$ variant of [Loizou et al., 2021] that sets the stepsize based on the following formula:

$$\gamma_t = \min \left\{ \frac{\phi(w_t) - \phi^\star}{\|\nabla \phi(w_t)\|^2}, \gamma_{\max} \right\} \quad \text{with } \phi = f \text{ (deterministic) or } \phi = f_i \text{ (stochastic)}, \tag{4}$$

for $\gamma_{\max}$ a hyperparameter of the method that restricts Polyak from taking a too large step. In our experiments, this parameter is set to $\gamma_{\max} = 1$ and we consider $\phi^\star = 0$, that is, the model can overfit the training regime.

3. Experiments and Discussion

We study stepsize tuners on the CIFAR10 image classification dataset [Krizhevsky and Hinton, 2009] with a squared loss and weight decay with parameter 10^{-4}. We will distinguish between the *deterministic* setting – full batch gradient descent (GD) – and the *stochastic* setting – minibatch gradient descent (SGD). We selected GD stepsizes close to the maximal stepsize that does not lead to divergence.

3.1. Deterministic regime

Setting. We first consider training deep networks on a reduced subset of 4096 samples of CIFAR10 training on all the samples each step. We trained on a 6-layer fully connected network, VGG, and with ResNet. Architectural details can be found in Appendix A.1.

Results. As promised, Armijo-GD (linesearch, deterministic setting) gives monotonically decreasing loss for all architectures (Fig. 1, top). This is in contrast with the extremely non-monotonic loss trajectories for fixed stepsize gradient descent (GD). Note that this non-monotonicity comes from non-linear discrete effects, and not sampling noise. However, Armijo-GD greatly underperforms constant stepsize GD. This observation comes somewhat as a surprise given that stepsize tuners such as Armijo have been the workhorse of non-convex optimization for decades [Nocedal and Wright, 1999]. This inefficiency seems to be related to the strong progressive sharpness (increasing sharpness) of Armijo-GD (Fig. 1, bottom). The cost of the non-monotonicity in the face of increasing sharpness is a decrease in stepsize (Fig 2, top). This can be further elucidated by

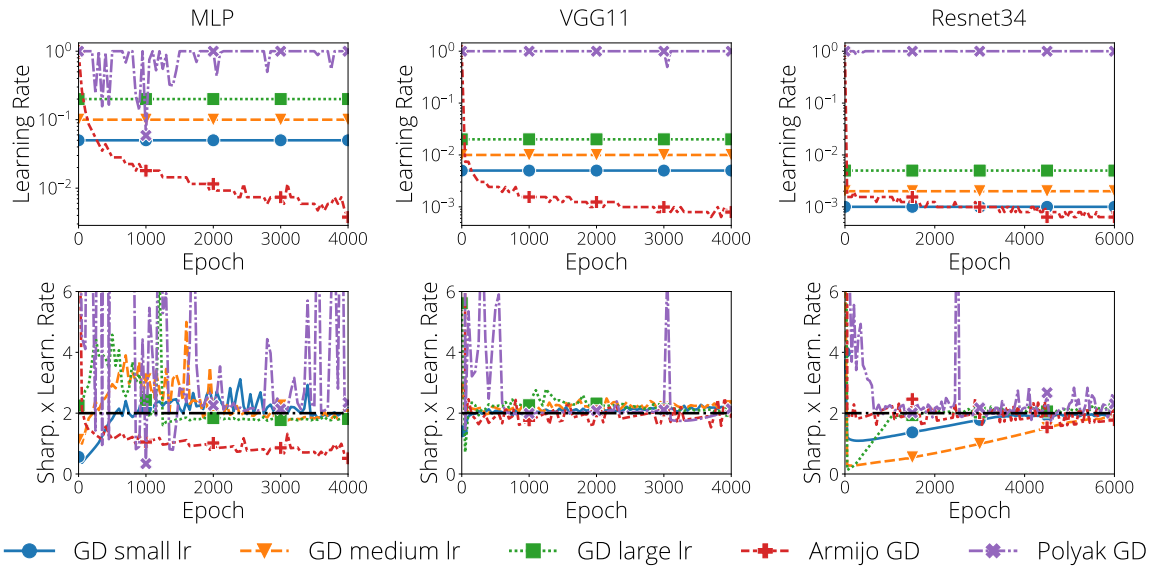

Figure 2: **A closer look at the learning rate dynamics.** In the top plot, we show the learning rate dynamics for the different stepsize tuners and GD. The Armijo backtracking line-search (red) decreases the learning rate monotonically, while the Polyak stepsize (violet) oscillates around the maximal acceptable value from (4), $\gamma_{\max} = 1$. In the bottom plot, we show the product of the learning rate and the sharpness of the Hessian. For constant stepsize GD, this product stabilizes at the critical EOS value 2 (black dashed line). For Armijo-GD the value either stays well below 2 or lingers around it from the start. For Polyak-GD, the product oscillates around the critical value without stabilizing like GD.

measuring the stepsize times the sharpness (Fig. 1, bottom). For fixed stepsize this oscillated above and below the EOS at value 2, leading to sharpness regularization; for Armijo-GD, the value is very much below 2 for the MLP (left, bottom), and near but often below 2 for the other architectures.

In contrast, Polyak-GD gives non-monotonic loss trajectories but with a faster decrease than Armijo, and leads to low final values of the sharpness (Fig 2). The stepsizes selected by Polyak's method tend to select the maximal acceptable value from (4), $\gamma_{\max} = 1$, and leads the normalized spectral norm to oscillate about (and sometimes well above) the critical value of 2 (Fig 2). Zooming in the first iterations of the method and recording at each iteration as done in Fig. 6 in Appendix B shows that Polyak's method only selects large stepsizes (around the maximal one) once the sharpness has sufficiently decreased.

3.2. Stochastic regime

Setting. We repeat the experiments, this time in the stochastic (minibatch) setting. We now train on all of CIFAR10, with minibatch sizes $\{256, 1024, 4096\}$. Results in Fig. 3 are in the MLP architecture while results on VGG and ResNet are presented in Appendix B.

Results. The performance of the stepsize tuners varies greatly with the batch size. For the largest batch size of 4096 (Fig. 3, top), we observe essentially the same behavior as in the deterministic (full batch) setting: Armijo-SGD displays progressive sharpening to long times, the stepsize decreases, and optimization is poor. Polyak once again performs well.

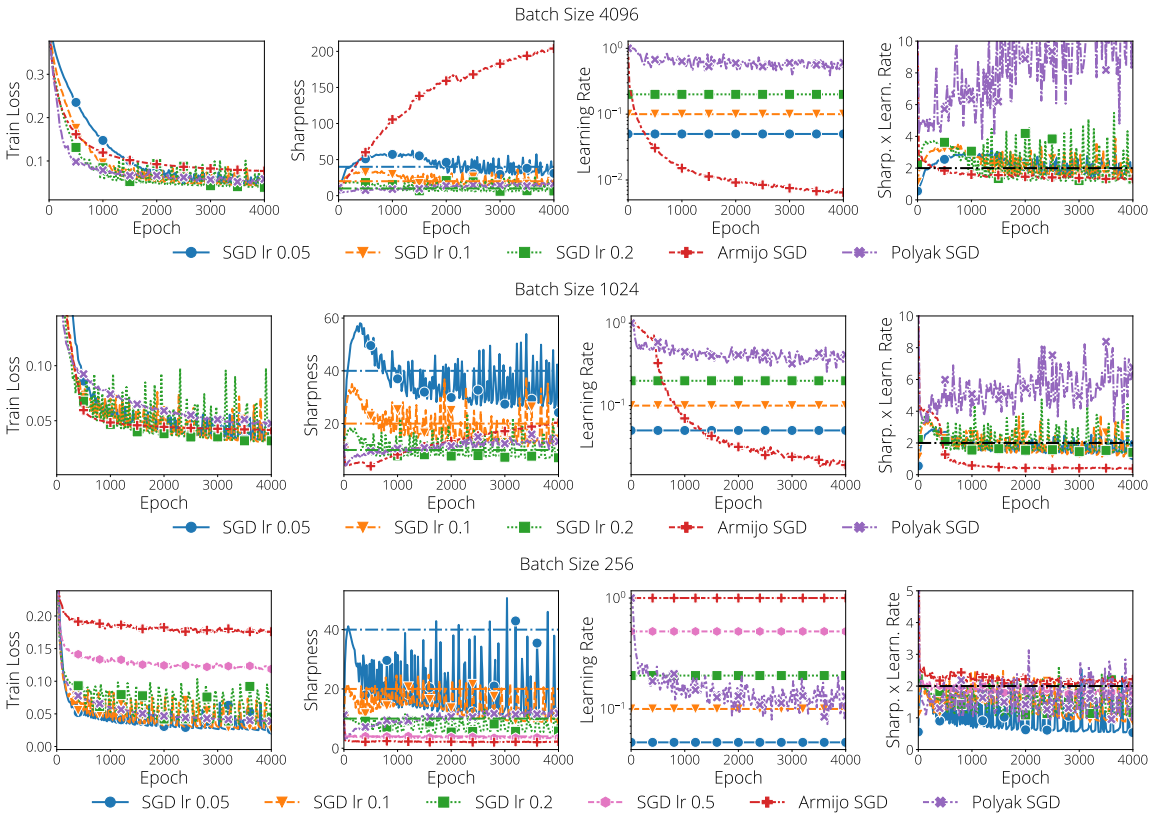

Figure 3: **Batch sizes impact the behavior of stepsize tuners in the stochastic regime.** We consider training an MLP (see Appendix for VGG and ResNet) on the full CIFAR10 dataset with various batch-sizes. In this stochastic regime, Armijo's performance varies greatly with the mini-batch considered, with a good performance at medium scale, and poor performance otherwise. Progressive sharpening of Armijo is only observed at medium and large scales. Polyak performs reasonably well in all settings, while operating potentially above the edge of stability regime in, e.g., the medium scale.

For the intermediate batch size of 1024, (Fig. 3, middle), Armijo-SGD optimizes well while maintaining relative smoothness of the loss trajectory. There is sharpening but less than some of the GD settings. Armijo-SGD still stays below the EOS. In contrast, Polyak-SGD now optimizes similarly to Armijo and fixed step size SGD (slower at the start but faster later) to similar values of the loss. Polyak-SGD seems to stabilize above the EOS.

For the smallest batch size of 256 (Fig. 3, bottom), Armijo-SGD induces small sharpness, but dramatically underperforms fixed stepsize SGD. In this case, Armijo-SGD often selects the maximal stepsize[1] and remains above EOS. This is in contrast to fixed stepsize SGD which remains below the EOS. This is similar to Polyak-SGD which performs poorly here compared to fixed stepsize while staying above the EOS.

The varying behavior of Armijo-SGD in the stochastic regime may be explained by the discrepancy between the maximal acceptable stepsize on a mini-batch and on the full loss. As shown by Mutschler and Zell [2020, Figure 7], the best stepsize decreasing the value on a mini-batch tends to

1. The plot is in epochs, at the scale of each steps, we observed stepsizes sometimes of the order of 0.5 rather than 1.

be larger than the stepsize that would have been selected on the full loss. The improved performance of Armijo in the stochastic setting may then be explained by its inability to satisfy the criterion that would be used in the full-batch regime.

4. Simple models fail to capture stepsize tuner dynamics

In their empirical study, Cohen et al. [2022b] uncovered two distinct phenomena: increasing largest Hessian eigenvalue at early times (progressive sharpening) and stabilization of said eigenvalue around a critical value (EOS). One natural hypothesis is that good stepsize tuners will quickly reach values near the EOS and stay there during training. We will analyze the stepsize tuners using a simple model of EOS behavior and see that EOS alone is not sufficient to understand our experiments - we must eventually understand the sharpening dynamics themselves.

The edge of stability dynamics is well represented by the projection onto the top eigenmode of the Hessian/NTK jointly with the dynamics of the eigenvalue itself [Agarwala et al., 2023, Damian et al., 2022]. We ask a basic question: can this simplified model be used to derive the stable values for the step sizes achieved by Armijo-GD and Polyak-GD observed in practice?

More concretely, suppose w_t is close to a minimum $w^\star$. Suppose that $w_t - w^\star$ is given by $a_t v_{\max,t}$, where a_t is a scalar and $v_{\max,t}$ is the eigenvector associated with the largest eigenvalue $\lambda_{\max,t}$ of $\nabla^2 f(w_t)$. The edge of stability dynamics corresponds to $\lambda_{\max,t}$ staying near $2/\gamma_t$ and a_t oscillating about 0. Under these assumptions, the question is: what is the behavior of the constraints on γ_t induced by the stepsize tuners?

We will make the further approximation that the loss is well approximated by the second order expansion around $w^\star$, and that v_t and $\lambda_{\max,t}$ are approximately constant. The loss and gradients are then approximated as

$$f(w_t) \approx \frac{1}{2}(w_t - w^\star)^\top \nabla^2 f(w^\star)(w_t - w^\star) = \frac{1}{2}\lambda_{\max} a_t^2, \quad \nabla f(w_t) \approx \lambda_{\max} a_t v_{\max}.$$

The Armijo-GD condition (3) becomes: $\frac{1}{2}\lambda_{\max}(1 - \lambda_{\max}\gamma_t)^2 a_t^2 \leq \lambda_{\max}(1 - c\gamma_t\lambda_{\max})a_t^2$, which constrains the stepsize selected by Armijo to satisfy

$$\gamma_t \leq \frac{2(1-c)}{\lambda_{\max}}.$$

For slowly changing $\lambda_{\max,t} \approx \lambda_{\max}$, Armijo-SGD stabilizes below the EOS. For Polyak-GD we have, for $\gamma_{\max} = 1$,

$$\gamma_t = \min\left\{\frac{f(w_t) - f^\star}{\|\nabla f(w_t)\|^2}, 1\right\} = \min\left\{\frac{1}{2\lambda_{\max}}, 1\right\} = \begin{cases} 1/(2\lambda_{\max}) & \text{if } \lambda_{\max} < 1/2 \\ 1 & \text{otherwise} \end{cases}.$$

This threshold is also below the EOS, and in fact smaller than the threshold for Armijo-GD.

However, in practice Polyak-GD seems to give rise to larger learning rates than Armijo-GD, while in the SGD setting things are highly senstitive to batch size. Therefore this simple model does not capture the true dynamics. Understanding stepsize tuners requires analysis of the *joint* dynamics of γ_t and $\lambda_{\max,t}$; in other words, progressive sharpening must be considered as well as EOS.

References

Atish Agarwala, Fabian Pedregosa, and Jeffrey Pennington. Second-order regression models exhibit progressive sharpening to the edge of stability. In *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pages 169–195. PMLR, 2023.

Leonard Berrada, Andrew Zisserman, and M Pawan Kumar. Training neural networks for and by interpolation. In *International conference on machine learning*. PMLR, 2020.

Jeremy Cohen, Simran Kaur, Yuanzhi Li, J. Zico Kolter, and Ameet Talwalkar. Gradient Descent on Neural Networks Typically Occurs at the Edge of Stability. In *International Conference on Learning Representations*, February 2022a.

Jeremy M. Cohen, Behrooz Ghorbani, Shankar Krishnan, Naman Agarwal, Sourabh Medapati, Michal Badura, Daniel Suo, David Cardoze, Zachary Nado, George E. Dahl, and Justin Gilmer. Adaptive Gradient Methods at the Edge of Stability, July 2022b.

Alex Damian, Eshaan Nichani, and Jason D. Lee. Self-Stabilization: The Implicit Bias of Gradient Descent at the Edge of Stability, September 2022.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware Minimization for Efficiently Improving Generalization. In *International Conference on Learning Representations*, April 2022.

Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An Investigation into Neural Net Optimization via Hessian Eigenvalue Density. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2232–2241. PMLR, May 2019.

Niv Giladi, Mor Shpigel Nacson, Elad Hoffer, and Daniel Soudry. At Stability's Edge: How to Adjust Hyperparameters to Preserve Minima Selection in Asynchronous Training of Neural Networks? In *Eighth International Conference on Learning Representations*, April 2020.

Justin Gilmer, Behrooz Ghorbani, Ankush Garg, Sneha Kudugunta, Behnam Neyshabur, David Cardoze, George Edward Dahl, Zachary Nado, and Orhan Firat. A Loss Curvature Perspective on Training Instabilities of Deep Learning Models. In *International Conference on Learning Representations*, March 2022.

Robert M Gower, Mathieu Blondel, Nidham Gazagnadou, and Fabian Pedregosa. Cutting some slack for sgd with adaptive polyak stepsizes. *arXiv preprint arXiv:2202.12328*, 2022.

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*, pages 770–778, 2016.

Xiaowen Jiang and Sebastian U Stich. Adaptive sgd with polyak stepsize and line-search: Robust convergence and variance reduction. *arXiv preprint arXiv:2308.06058*, 2023.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Tech. Rep. 2009-005, Dept. Comput. Sci., Univ. Toronto*, 2009.

Shuang Li, William J Swartworth, Martin Takáč, Deanna Needell, and Robert M Gower. Sp2: A second order stochastic polyak method. *arXiv preprint arXiv:2207.08171*, 2022.

Nicolas Loizou, Sharan Vaswani, Issam Hadj Laradji, and Simon Lacoste-Julien. Stochastic polyak step-size for sgd: An adaptive learning rate for fast convergence. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021.

Maximus Mutschler and Andreas Zell. Parabolic approximation line search for dnns. *Advances in Neural Information Processing Systems*, 33:5405–5416, 2020.

Behnam Neyshabur, Srinadh Bhojanapalli, David Mcallester, and Nati Srebro. Exploring Generalization in Deep Learning. In *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 2017.

Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.

Boris Teodorovich Polyak. Minimization of unsmooth functionals. *USSR Computational Mathematics and Mathematical Physics*, 1969.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Sharan Vaswani, Aaron Mishkin, Issam Laradji, Mark Schmidt, Gauthier Gidel, and Simon Lacoste-Julien. Painless stochastic gradient: Interpolation, line-search, and convergence rates. *Advances in neural information processing systems*, 32, 2019.

Lei Wu, Chao Ma, and Weinan E. How SGD Selects the Global Minima in Over-parameterized Learning: A Dynamical Stability Perspective. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Appendix A. Experimental details

A.1. Architectures

Here are some additional details on the architectures considered

1. *MLP*: We considered 6 layers of output size $(128, 128, 64, 64, 32, 32)$ followed by a last classification layer of output dimension 10. All layers are without batch normalization or dropout. Each layer consists in a linear transformation followed by the ReLU activation function except for the classification layer, which has no activation.

2. *VGG11*: We follow the implementation of the VGG11 architecture [Simonyan and Zisserman, 2014] except that we removed dropout layers and the batch normalization layers.

3. *ResNet34*: We follow the implementation of the Resnet34 architecture [He et al., 2016] except that we removed the batch normalization layers.

A.2. Evaluation

1. *Sharpness computation*: To compute the sharpness, that is, the spectral norm of the Hessian, we used a power iteration method, stopped once changes in the residual go below a threshold of 10^{-3} or after 1000 iterations.

2. *Visualization*: For the stochastic experiments we average the evaluations of each quantity around 20 steps.

Appendix B. Additional experiments

B.1. VGG and ResNet in stochastic regime

On Fig. 4 and 5, we display the training dynamics of SGD, ArmijoSGD and PolyakSGD, when considering the VGG and ResNet architectures. Compared to the MLP setting of Fig 3, we still observe the same sharpening Armijo at large and medium scales, though less starkly displayed for the ResNet architecture. In this regime, we oberve that the best regime for which Armijo works is the smallest one, hinting that the ideal batch-size for Armijo naturally depends on the architecture considered. On the other hand, Polyak SGD performs very well in all settings. Note that the constant stepsizes for SGD were chosen as in the full batch regime and do not illustrate the best possible performance of SGD in these settings. In other words, the stark contrast of performance between PolyakSGD and the constant size variants may be an effect of the loose search of constant stepsizes of SGD rather than a true qualitative difference. On the other hand, this illustrates the benefit of Polyak SGD in this context which alleviates the search for a stepsize.

B.2. Zoom into Polyak's behavior

In Fig. 2 we plotted several metrics every 50 epochs. In Fig. 6, we show the train loss and the learning rate at every epoch (that is every step as we are in full batch) for the first 200 epochs. We observe that the learning rate is not just constant to 1 but oscillates between 0 and 1 except at the start where it stays at a low value. Polyak starts taking large stepsizes (between 0 and 1) only once the sharpness has significantly decreased.

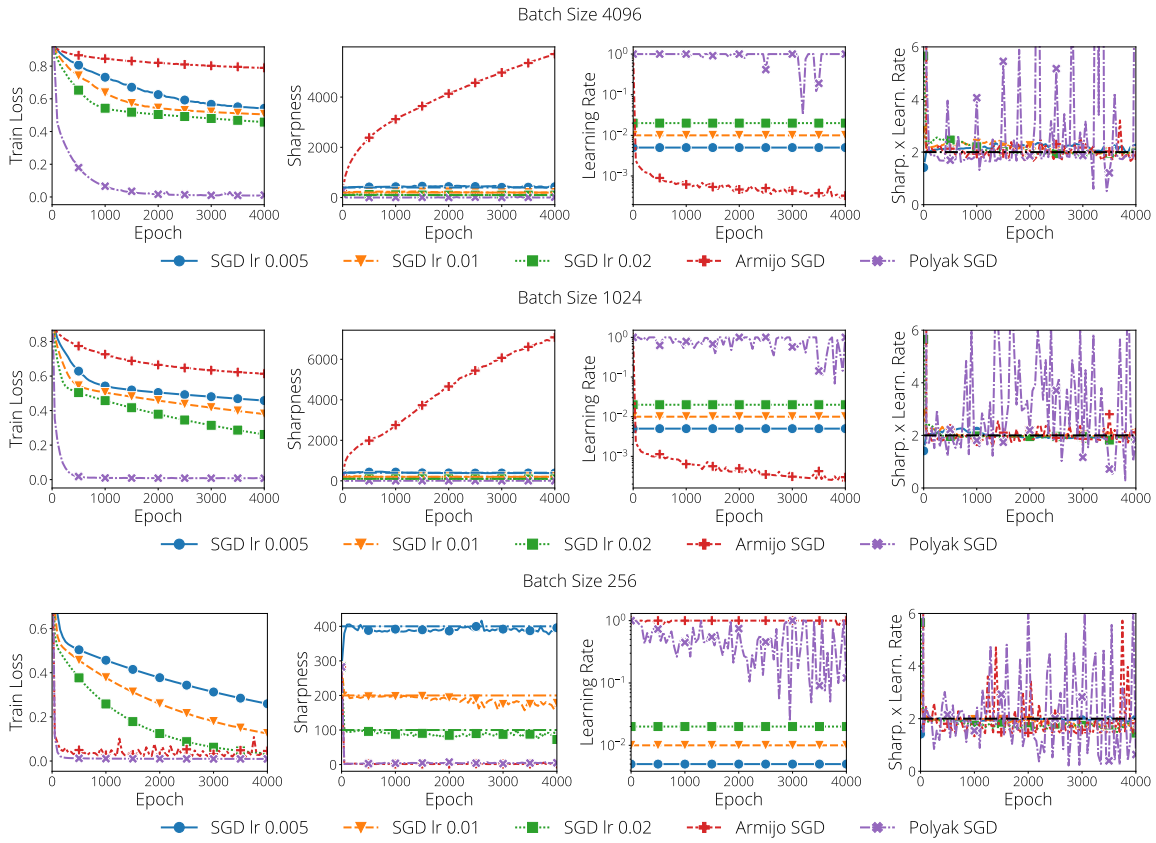

Figure 4: Training dynamics in stochastic regime for the VGG11 architecture.

We also present on Fig. 7, how the maximal stepsize for the Polyak method affects the increasing sharpness. Unsurprisingly by setting a small maximal stepsize we retrieve a comportment akin to the one displayed by constant stepsize SGD. On the other hand, in this experiment, fixing a maximal stepsize even relatively large does not lead to divergence, though the algorithm is less stable at the start.

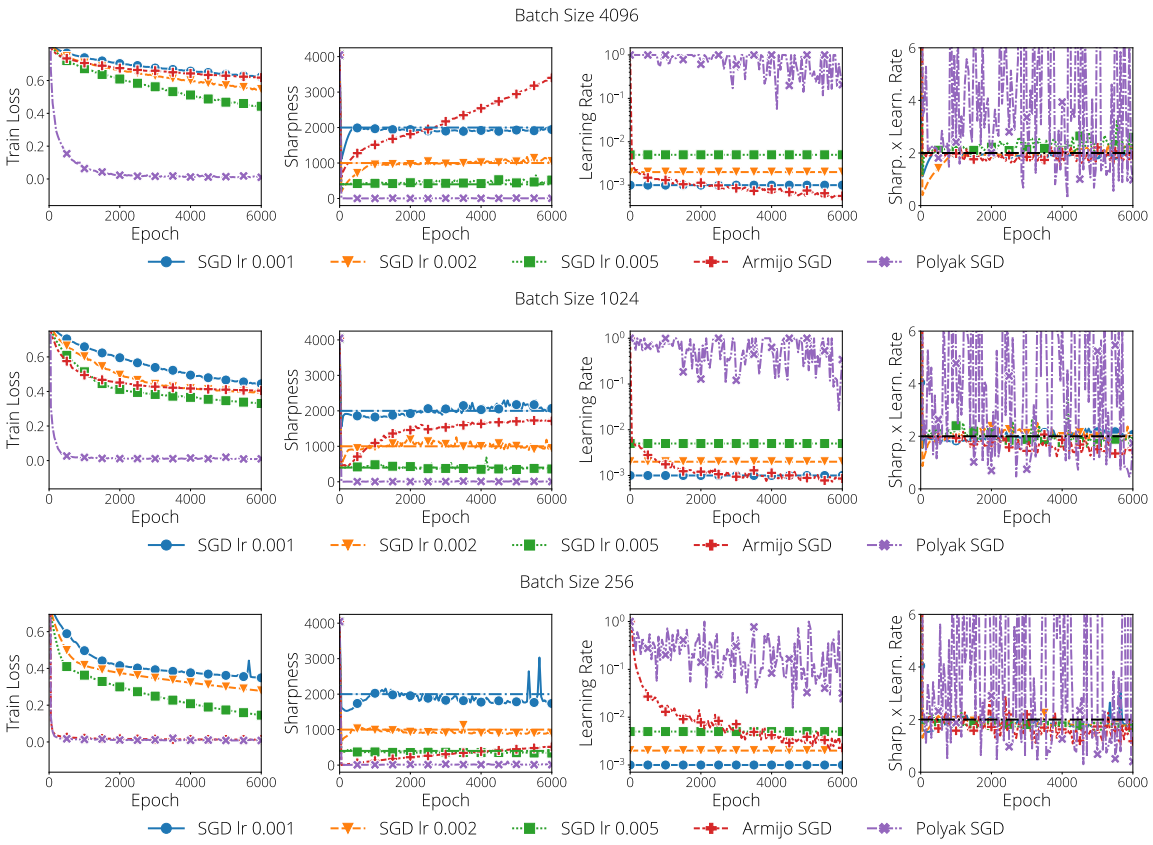

Figure 5: Training dynamics in stochastic regime for the ResNet34 architecture.

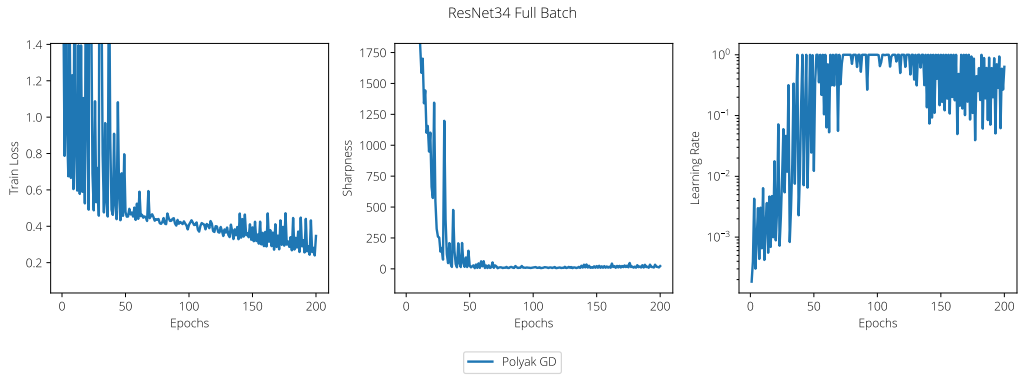

Figure 6: Polyak GD with maximal stepsize of 1. on a ResNet34 for the first 200 epochs. We observe that at the start, the learning rate remains small until the sharpness has sufficiently decrease for the optimizer to take large stepsizes.

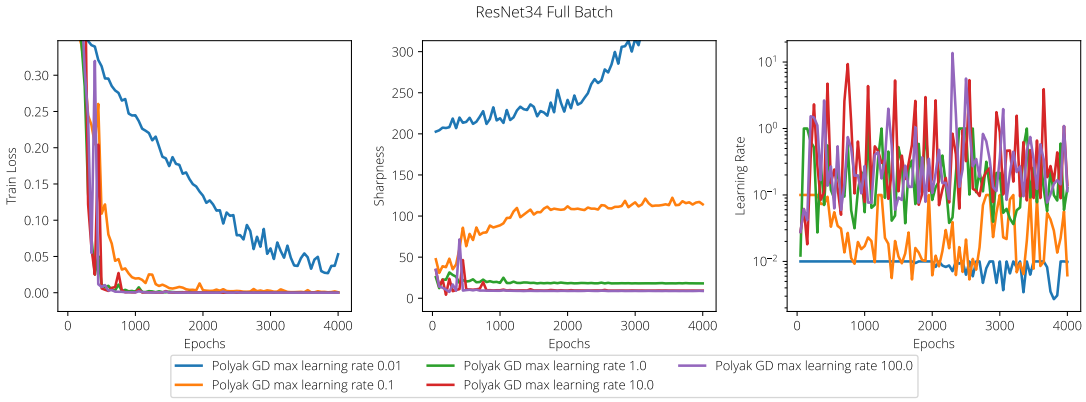

Figure 7: Varying the maximal stepsize of Polyak. Even for large maximal stepsizes Polyak converges efficiently, though it is less stable at the start. Small maximum stepsizes lead to a similar behavior as constant stepsizes.