How Memory in Optimization Algorithms Implicitly Modifies the Loss

Matias D. Cattaneo*
Princeton University
cattaneo@princeton.edu

Boris Shigida*
Princeton University
bs1624@princeton.edu

Abstract

In modern optimization methods used in deep learning, each update depends on the history of previous iterations, often referred to as *memory*, and this dependence decays fast as the iterates go further into the past. For example, gradient descent with momentum has exponentially decaying memory through exponentially averaged past gradients. We introduce a general technique for identifying a memoryless algorithm that approximates an optimization algorithm with memory. It is obtained by replacing all past iterates in the update by the current one, and then adding a correction term arising from memory (also a function of the current iterate). This correction term can be interpreted as a perturbation of the loss, and the nature of this perturbation can inform how memory implicitly (anti-)regularizes the optimization dynamics. As an application of our theory, we find that Lion does not have the kind of implicit anti-regularization induced by memory that AdamW does, providing a theory-based explanation for Lion's better generalization performance recently documented (X. Chen et al., 2023).

1 Introduction

Many optimization methods used in deep learning are first-order methods with exponentially decaying memory. For example, adding "momentum" to gradient descent (GD) is a well-established practice to make training smoother and convergence faster (e. g. Krizhevsky, Sutskever, and G. E. Hinton (2012)). Adaptive methods such as Adam (Kingma and Ba, 2017), RMSProp (Tieleman, G. Hinton, et al., 2012), AdamW (Loshchilov and Hutter, 2019), and AdaFactor (Shazeer and Stern, 2018), which are commonly used to train large language models (Grattafiori et al., 2024; DeepSeek-AI et al., 2024; Chowdhery et al., 2023), all have exponentially decaying memory. Despite the popularity of such optimization methods, there is little theoretical knowledge about the implicit regularization memory introduces to them (potentially informing what regions of the loss space the method takes the iterates to, what minima they converge to, how such minima influence the generalization of the trained model, and so on). In this article, we introduce a general framework for identifying such regularization.

We study a general class of optimization algorithms described by the following iteration

$$\boldsymbol{\theta}^{(n+1)} = \boldsymbol{\theta}^{(n)} - h\boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)}), \tag{1}$$

where $\boldsymbol{\theta}^{(n)} \in \mathbb{R}^d$ are the (evolving) parameters of the machine learning model, $\boldsymbol{\theta}^{(0)}$ is some initial condition, h is the step size or learning rate, and the functions $\boldsymbol{F}^{(n)}$ map from (some subset of) $(\mathbb{R}^d)^{n+1}$ to $\mathbb{R}^d$ and are allowed to be different at each iteration. The right-hand side in Eq. (1) depends on the whole history of previous iterates, which means the algorithm has memory.

For many algorithms used in practice, dependence on the history comes in one specific form: by using what we call "*momentum variables*", that is, exponential averages of some functions of the iterate $\boldsymbol{\theta}^{(n)}$ (usually, more specifically, functions of the loss gradient). We present five leading examples to illustrate this point.

---

*Equal contribution

Example 1.1 (Heavy-ball momentum gradient descent; Polyak (1964)). This optimizer can be written in the form (1) with

$$\boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)}) = \boldsymbol{m}_1^{(n+1)},$$
$$\text{where} \quad \boldsymbol{m}_1^{(n+1)} = \sum_{k=0}^{n} \beta^{n-k} \nabla \mathcal{L}(\boldsymbol{\theta}^{(k)}), \tag{2}$$

for some initial condition $\boldsymbol{\theta}^{(0)}$, and where $\beta \in [0, 1)$ is the momentum parameter, $\mathcal{L}$ is the loss function to be optimized, and $\nabla \mathcal{L}$ is its gradient. ∎

This optimizer in Example 1.1 is often just referred to as GD with momentum, where the exponential sum $\boldsymbol{m}_1^{(n+1)}$ in Eq. (2) is the *momentum variable*: it exponentially averages past gradients. The aforementioned optimizer is well-known and often used for training recurrent neural networks and convolutional neural networks, but it underperforms adaptive optimizers when training other architectures such as transformers (Zhang et al., 2020; Liu et al., 2020; Anil et al., 2019; Kunstner et al., 2023). The following modification is also commonly used (this formulation is taken from Choi et al. (2020) and matches the standard PyTorch implementation).

Example 1.2 (Nesterov's accelerated gradient descent; Nesterov (1983)). This optimizer can be written in the form (1) with

$$\boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)}) = \boldsymbol{m}_1^{(n+1)} + \boldsymbol{m}_2^{(n+1)},$$
$$\text{where} \quad \boldsymbol{m}_1^{(n+1)} = \beta \sum_{k=0}^{n} \beta^{n-k} \nabla \mathcal{L}(\boldsymbol{\theta}^{(k)}),$$
$$\boldsymbol{m}_2^{(n+1)} = \nabla \mathcal{L}(\boldsymbol{\theta}^{(n)}),$$

for some initial condition $\boldsymbol{\theta}^{(0)}$, and where $\beta \in [0, 1)$ is the momentum parameter, $\mathcal{L}$ is the loss function to be optimized, and $\nabla \mathcal{L}$ is its gradient. ∎

The next example presents a leading adaptive optimizer nowadays commonly used for training large language models (Grattafiori et al., 2024; DeepSeek-AI et al., 2024).

Example 1.3 (AdamW; Loshchilov and Hutter (2019)). The optimizer can be written in the form (1) with

$$\boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)}) = \frac{\boldsymbol{m}_1^{(n+1)}}{\sqrt{\boldsymbol{m}_2^{(n+1)} + \varepsilon}} + \boldsymbol{m}_3^{(n+1)},$$
$$\text{where} \quad \boldsymbol{m}_1^{(n+1)} = \frac{1 - \beta_1}{1 - \beta_1^{n+1}} \sum_{k=0}^{n} \beta_1^{n-k} \nabla \mathcal{L}(\boldsymbol{\theta}^{(k)}),$$
$$\boldsymbol{m}_2^{(n+1)} = \frac{1 - \beta_2}{1 - \beta_2^{n+1}} \sum_{k=0}^{n} \beta_2^{n-k} \big(\nabla \mathcal{L}(\boldsymbol{\theta}^{(k)})\big)^2,$$
$$\boldsymbol{m}_3^{(n+1)} = \lambda \boldsymbol{\theta}^{(n)},$$

for some initial condition $\boldsymbol{\theta}^{(0)}$, and where $0 \leqslant \beta_1, \beta_2 < 1$ are momentum parameters, $\varepsilon > 0$ is a numerical stability parameter, $0 < \lambda < 1$ is a weight decay parameter, and the squares and square roots are taken component-wise. ∎

In Example 1.3, $\boldsymbol{m}_1^{(n+1)}$ and $\boldsymbol{m}_2^{(n+1)}$ are also *momentum variables*: exponentially averaged gradients and exponentially averaged squared gradient components respectively, with coefficients in front of the sum, such as $(1 - \beta_1)(1 - \beta_1^{n+1})^{-1}$, providing "bias correction" (Kingma and Ba, 2017). The variable $\boldsymbol{m}_3^{(n+1)}$ here is a degenerate momentum variable, with memory decaying infinitely fast.

The following modification incorporates Nesterov's momentum into AdamW. This formulation is taken from Choi et al. (2020) (except here ε is inside the square root in the denominator).

Example 1.4 (NAdam with decoupled weight decay; Dozat (2016)). The optimizer can be written in the form (1) with

$$\boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)},\ldots,\boldsymbol{\theta}^{(0)}) = \frac{\beta_1 \boldsymbol{m}_1^{(n+1)} + (1-\beta_1)\boldsymbol{m}_4^{(n+1)}}{\sqrt{\boldsymbol{m}_2^{(n+1)} + \varepsilon}} + \boldsymbol{m}_3^{(n+1)},$$

$$\text{where} \quad \boldsymbol{m}_1^{(n+1)} = \frac{1-\beta_1}{1-\beta_1^{n+1}}\sum_{k=0}^{n}\beta_1^{n-k}\nabla\mathcal{L}(\boldsymbol{\theta}^{(k)}),$$

$$\boldsymbol{m}_2^{(n+1)} = \frac{1-\beta_2}{1-\beta_2^{n+1}}\sum_{k=0}^{n}\beta_2^{n-k}\big(\nabla\mathcal{L}(\boldsymbol{\theta}^{(k)})\big)^2,$$

$$\boldsymbol{m}_3^{(n+1)} = \lambda\boldsymbol{\theta}^{(n)},$$

$$\boldsymbol{m}_4^{(n+1)} = \nabla\mathcal{L}(\boldsymbol{\theta}^{(n)}),$$

for some initial condition $\boldsymbol{\theta}^{(0)}$, and where $0 \leqslant \beta_1, \beta_2 < 1$ are momentum parameters, $\varepsilon > 0$ is a numerical stability parameter, $0 < \lambda < 1$ is a weight decay parameter, and the squares and square roots are taken component-wise. ∎

As a final example, consider a new optimizer called Lion (Evo**L**ved S**i**gn M**o**me**n**tum), which was recently discovered by an evolutionary search, and then verified to generalize better than AdamW on a variety of tasks (X. Chen et al., 2023). We consider a generalized version of the Lion algorithm.

Example 1.5 (Lion-$\mathcal{K}$; L. Chen et al. (2024)). The optimizer can be written in the form of (1) with

$$\boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)},\ldots,\boldsymbol{\theta}^{(0)}) = -\nabla\mathcal{K}(\boldsymbol{m}_1^{(n+1)} + \boldsymbol{m}_2^{(n+1)}) + \boldsymbol{m}_3^{(n+1)},$$

$$\text{where} \quad \boldsymbol{m}_1^{(n+1)} = -(1-\rho_2)\frac{\rho_1}{\rho_2}\sum_{k=0}^{n}\rho_2^{n-k}\nabla\mathcal{L}(\boldsymbol{\theta}^{(k)}),$$

$$\boldsymbol{m}_2^{(n+1)} = -\left(1 - \frac{\rho_1}{\rho_2}\right)\nabla\mathcal{L}(\boldsymbol{\theta}^{(n)}),$$

$$\boldsymbol{m}_3^{(n+1)} = \lambda\boldsymbol{\theta}^{(n)}, \tag{3}$$

for some initial condition $\boldsymbol{\theta}^{(0)}$, and where $0 \leqslant \rho_1, \rho_2 < 1$ are Lion's momentum parameters (Section 3 makes clear why we choose the letter ρ rather than β for them), $\lambda > 0$ is a weight decay parameter, $\mathcal{K}: \mathbb{R}^d \to \mathbb{R}$ is some convex function, and $\nabla\mathcal{K}$ is its subgradient. ∎

Ordinary Lion corresponds to $\mathcal{K}(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$ and $\nabla\mathcal{K}(\boldsymbol{x}) = \text{sign}(\boldsymbol{x})$ in Example 1.5, where the sign function is understood component-wise. We consider the generalized Lion-$\mathcal{K}$ algorithm because it covers a few known algorithms as special cases: see Table 1 and Section 3.1 in L. Chen et al. (2024). In fact, it also includes Example 1.1 as a special case by taking $\mathcal{K}(\boldsymbol{x}) = \|\boldsymbol{x}\|^2/2$, $\rho_1 = \rho_2$, and $\lambda = 0$, but we will deal with that important specific example separately for clarity.

It is reasonable to expect that adding exponentially decaying memory to an algorithm in such a way as described above (for example, replacing the gradient with exponentially averaged past gradients) changes the optimization dynamics, thereby affecting the performance of the trained model. The technique we introduce identifies *how* the iteration evolution changes when memory is added. This technique starts with an iteration having memory, and replaces it by a memoryless iteration that approximates the original one, provided a correction term is added. Specifically, we start with algorithm (1), and then construct a corresponding new memoryless iteration of the form

$$\tilde{\boldsymbol{\theta}}^{(n+1)} = \tilde{\boldsymbol{\theta}}^{(n)} - h\big[\boldsymbol{F}^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}) + \boldsymbol{M}^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)})\big], \tag{4}$$

where we slightly abuse notation and put

$$\boldsymbol{F}^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}) \equiv \boldsymbol{F}^{(n)}(\underbrace{\tilde{\boldsymbol{\theta}}^{(n)},\ldots,\tilde{\boldsymbol{\theta}}^{(n)}}_{n+1\text{ times}}),$$

and where the function $\boldsymbol{M}^{(n)}(\boldsymbol{\theta})$ captures a correction due to the presence of memory. We then prove an explicit bound on the approximation error $\|\boldsymbol{\theta}^{(n)} - \tilde{\boldsymbol{\theta}}^{(n)}\|$, as a function of the learning rate h. Interpreting

the correction term can sometimes generate predictions on whether memory helps or hurts generalization of first-order methods with momentum.

Our general theory only relies on memory decaying sufficiently fast, not necessarily in the form of momentum variables, and thus covers all the examples listed above and many others, while also allowing for both full-batch and mini-batch training. Section 2 first presents a heuristic discussion of our proposed technique focusing on the simplest possible case for clarity: GD with momentum (Example 1.1). Then, Section 3 presents our main theoretical contribution, which is more cumbersome in terms of notation and technicalities due to its generality.

Depending on specific optimization algorithm considered, our general result can lead to different theoretical and practical conclusions. As a substantive application, Section 4 studies AdamW (Example 1.3) and Lion-$\mathcal{K}$ (Example 1.5), and demonstrates that *Lion does not suffer from the anti-regularization effect that AdamW's memory has*, which may be a partial explanation for Lion's better generalization on many tasks. Section 5 discusses further implications of our main theoretical result: modified equations, and mini-batch training.

1.1 Related Literature

Approximating a memoryful iteration with a memoryless one is closely connected with the method of modified equations (sometimes called *backward error analysis*), where a discrete algorithm like (1) is approximated by a continuous solution of an ordinary differential equation or stochastic differential equation. Typically, this method can only be applied to an algorithm with no memory, in a possibly enlarged phase space as opposed to $\mathbb{R}^d$; for example, heavy-ball momentum GD has no memory when viewed as a discrete iteration $(\boldsymbol{\theta}^{(n)}, \boldsymbol{m}^{(n)})$ in $\mathbb{R}^{2d}$, where $\boldsymbol{m}^{(n)}$ is the momentum variable. The general technique introduced in this paper can be used to derive a memoryless discrete iteration which can then be approximated by a continuous trajectory. Background on the method of modified equations can be found in Ernst Hairer and Wanner (2006) and Q. Li, Tai, and E (2017).

Works deriving modified equations for (S)GD with or without momentum include Barrett and Dherin (2021), Smith et al. (2021), Ghosh et al. (2023), Farazmand (2020), Kovachki and Stuart (2021), Miyagawa (2022), Rosca et al. (2023), and Q. Li, Tai, and E (2017). In particular, Ghosh et al. (2023) identified that momentum strongly regularizes the loss function in the case of GD, though their error bounds both have a different focus (continuous approximation rather than discrete one), and follow a different approach which appears hard to generalize to other algorithms. Our approach works for a wide class of algorithms, and we recover their main results in Section 5. Works approximating adaptive methods with continuous trajectories include C. Ma, Wu, and E (2022), Malladi et al. (2022), and Barakat and Bianchi (2021). More recently, Cattaneo, Klusowski, and Shigida (2024) studied the special case of Adam / RMSProp. Their focus is not on memory but on continuous approximations; in particular, they do not have approximation bounds between two discrete iterations like we do. In addition, their arguments are highly specialized to Adam, and they do not incorporate weight decay. Although we also discuss Adam (with weight decay) extensively, it is only because of its importance in practice, and our results cover a much broader class of optimizers.

This paper is also connected to the strand of the literature studying the implicit bias of optimization algorithms. For example, Xie and Z. Li (2024) and L. Chen et al. (2024) prove that weight decay causes AdamW and Lion to solve an ℓ_∞-norm constrained optimization problem. In addition, a large body of work is devoted to the bias of optimization algorithms towards the direction of the max-margin vector (Soudry, Hoffer, and Srebro, 2018; Nacson, J. Lee, et al., 2019; Nacson, Srebro, and Soudry, 2019; Q. Qian and X. Qian, 2019; B. Wang et al., 2022; Gunasekar, J. Lee, et al., 2018; Ji and Telgarsky, 2018; Ji and Telgarsky, 2019). Similarly, Damian, T. Ma, and Jason D Lee (2021), Z. Li, T. Wang, and Arora (2022), Arora, Z. Li, and Panigrahi (2022), Wen, T. Ma, and Z. Li (2022), and Damian, Nichani, and Jason D. Lee (2023) explore the sharpness of regions SGD converges to. Gunasekar, Woodworth, et al. (2017) and Arora, Cohen, et al. (2019) study implicit regularization in matrix factorization. Y. Li, Wei, and T. Ma (2019) prove in a certain setting that a larger learning rate leads to better generalization.

1.2 Notation

We use standard notations for the ℓ_p norm of a vector $\|\boldsymbol{v}\|_p = \left(\sum_i |v_i|^p\right)^{1/p}$; the infinity-norm is defined as $\|\boldsymbol{v}\|_\infty = \max_i |v_i|$; finally, the norm without indices is by default Euclidean: $\|\boldsymbol{v}\| \equiv \|\boldsymbol{v}\|_2$. When we write $\boldsymbol{u}_{n,h} = O(g(h))$, where $g(h)$ is some fixed function of h and $\boldsymbol{u}_{n,h}$ is some sequence of vectors possibly

depending on h, we mean that there is a constant C not depending on h or n such that $\|\boldsymbol{u}_{n,h}\| \leqslant Cg(h)$. For a vector-valued function $\boldsymbol{f}(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k) \in \mathbb{R}^d$, the notation $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}_\ell}$, where $\boldsymbol{u}_\ell \in \mathbb{R}^m$, means a $d \times m$ matrix whose (i,j)-th element is $\frac{\partial f_i}{\partial u_{\ell;j}}$. Similarly, $\frac{\partial^2 \boldsymbol{f}}{\partial \boldsymbol{u}_\ell \partial \boldsymbol{u}_p}$ is a tensor of order 3 whose (i,j,k)-th coordinate is $\frac{\partial^2 f_i}{\partial u_{\ell;j} \partial u_{p;k}}$. (S)GD is an abbreviation for (stochastic) gradient descent, O(S)DE for ordinary (stochastic) differential equation. We will contract repeating arguments when convenient, e.g. instead of $\boldsymbol{F}^{(n)}(\boldsymbol{\theta}, \ldots, \boldsymbol{\theta})$ we will write just $\boldsymbol{F}^{(n)}(\boldsymbol{\theta})$. We will use notation $\mathcal{L}(\cdot)$ for the loss and $\nabla\mathcal{L}(\cdot)$ for its gradient, h for the learning rate.

2 Developing Intuition: Memory Regularizes GD with Momentum

We provide a heuristic explanation of our technique, considering the simplest algorithm with exponentially decaying memory: heavy-ball momentum GD (Example 1.1). As explained above, we would like to remove the dependence of the right-hand side in

$$\boldsymbol{\theta}^{(n+1)} = \boldsymbol{\theta}^{(n)} - h \sum_{k=0}^{n} \beta^{n-k} \nabla\mathcal{L}(\boldsymbol{\theta}^{(k)}) \tag{5}$$

on the "past" iterates $\boldsymbol{\theta}^{(n-1)}, \ldots, \boldsymbol{\theta}^{(0)}$, leaving only the dependence on the "current" iterate $\boldsymbol{\theta}^{(n)}$. Let us represent "past" iterates through the "current" one. First, write

$$\boldsymbol{\theta}^{(n-1)} = \boldsymbol{\theta}^{(n)} + h \sum_{b=0}^{n-1} \beta^b \nabla\mathcal{L}(\boldsymbol{\theta}^{(n-1-b)})$$

$$= \boldsymbol{\theta}^{(n)} + h \sum_{b=0}^{n-1} \beta^b \nabla\mathcal{L}(\boldsymbol{\theta}^{(n)}) + O(h^2),$$

where the second equality relies on exponential averaging to replace historical iterates with $\boldsymbol{\theta}^{(n)}$, influencing only higher-order terms. Similarly,

$$\boldsymbol{\theta}^{(n-2)} = \boldsymbol{\theta}^{(n-1)} + h \sum_{b=0}^{n-2} \beta^b \nabla\mathcal{L}(\boldsymbol{\theta}^{(n-2-b)})$$

$$= \boldsymbol{\theta}^{(n-1)} + h \sum_{b=0}^{n-2} \beta^b \nabla\mathcal{L}(\boldsymbol{\theta}^{(n)}) + O(h^2),$$

$$= \boldsymbol{\theta}^{(n)} + h \left\{ \sum_{b=0}^{n-1} \beta^b + \sum_{b=0}^{n-2} \beta^b \right\} \nabla\mathcal{L}(\boldsymbol{\theta}^{(n)}) + O(h^2),$$

where the last equality follows by inserting the expression for $\boldsymbol{\theta}^{(n-1)}$. Continue like this up to

$$\boldsymbol{\theta}^{(n-k)} = \boldsymbol{\theta}^{(n)} + h \sum_{l=1}^{k} \sum_{b=0}^{n-l} \beta^b \nabla\mathcal{L}(\boldsymbol{\theta}^{(n)}) + O(k^2 h^2),$$

where the k^2 provides an estimate on the accumulation of error terms of order h^2.

We have now represented all the historical iterates through the current one. Combining it with Taylor expansion around $\boldsymbol{\theta}^{(n)}$ in Eq. (5), we obtain

$$\boldsymbol{\theta}^{(n+1)} = \boldsymbol{\theta}^{(n)} - h \sum_{k=0}^{n} \beta^k \left\{ \nabla\mathcal{L}(\boldsymbol{\theta}^{(n)}) + h\nabla^2\mathcal{L}(\boldsymbol{\theta}^{(n)}) \sum_{l=1}^{k} \sum_{b=0}^{n-l} \beta^b \nabla\mathcal{L}(\boldsymbol{\theta}^{(n)}) + O(k^2 h^2) \right\}$$

$$= \boldsymbol{\theta}^{(n)} - h \frac{1 - \beta^{n+1}}{1 - \beta} \nabla\mathcal{L}(\boldsymbol{\theta}^{(n)})$$

$$- h^2 \frac{\beta[1 - (2n+1)\beta^n(1-\beta) - \beta^{2n+1}]}{(1-\beta)^3} \nabla^2\mathcal{L}(\boldsymbol{\theta}^{(n)})\nabla\mathcal{L}(\boldsymbol{\theta}^{(n)}) + O(h^3).$$

5

Now using $\nabla^2\mathcal{L}(\boldsymbol{\theta})\nabla\mathcal{L}(\boldsymbol{\theta}) = (1/2)\nabla\|\nabla\mathcal{L}(\boldsymbol{\theta})\|^2$ and neglecting coefficients that go exponentially fast to zero as $n \to \infty$, we obtain that heavy-ball momentum GD is close to ordinary GD (no momentum) with a different step size and different loss, given by

$$\boldsymbol{\theta}^{(n+1)} = \boldsymbol{\theta}^{(n)} - \frac{h}{1-\beta}\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}), \quad \text{where}$$

$$\tilde{\mathcal{L}}(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) + \frac{h\beta}{2(1-\beta)^2}\|\nabla\mathcal{L}(\boldsymbol{\theta})\|^2. \tag{6}$$

The term $\frac{h\beta}{2(1-\beta)^2}\|\nabla\mathcal{L}(\boldsymbol{\theta})\|^2$ that is added implicitly to the loss by the momentum can be interpreted as implicit regularization. Since β is usually taken to be close to one, the term strongly penalizes the squared norm of the gradient. There is empirical evidence that such penalization improves generalization (Barrett and Dherin, 2021; Smith et al., 2021; Ghosh et al., 2023). In fact, this term (up to coefficients) can be interpreted as a first-order approximation of ℓ_2 sharpness (Ghosh et al., 2023), which suggests that it moves the trajectory towards "flatter" minima; this is often thought to improve generalization (Foret et al., 2021), although in general the evidence on the connection between sharpness of minima and generalization is mixed (Andriushchenko et al., 2023).

3 General Theory: The Effect of Memory

The general form of an optimization algorithm with memory is given by Eq. (1). The only property of memory we use is that it (uniformly in n) decays exponentially fast, as made precise by Assumption 3.1 below. Openness and convexity of the domain of optimization $\mathcal{D}$, that is, where all $\{\boldsymbol{\theta}^{(n)}\}$ will be assumed to lie, are innocuous assumptions (e.g., $\mathbb{R}^d$ is open and convex); we impose them to avoid technicalities with differentiation and Taylor expansion.

Assumption 3.1 (Memory Decay). *Let $\mathcal{D}$ be an open convex domain in $\mathbb{R}^d$. Let $\{\boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)})\}_{n=0}^{\infty}$ be a family of functions $\mathcal{D}^{n+1} \to \mathbb{R}^d$, two times continuously differentiable on their respective domains, such that for any $n \in \mathbb{Z}_{\geq 0}$, $k_1, k_2 \in \{0, \ldots, n\}$, $r, i, j \in \{1, \ldots, d\}$,*

$$|F_r^{(n)}| \leqslant \gamma_{-1}, \tag{7}$$

$$\left|\frac{\partial F_r^{(n)}}{\partial \theta_i^{(n-k_1)}}\right| \leqslant \gamma_{k_1}, \tag{8}$$

$$\left|\frac{\partial^2 F_r^{(n)}}{\partial \theta_i^{(n-k_1)} \partial \theta_j^{(n-k_2)}}\right| \leqslant \gamma_{k_1,k_2}, \tag{9}$$

where $\boldsymbol{F}^{(n)} = (F_1^{(n)}, \ldots, F_d^{(n)})^{\mathsf{T}}$, and γ_{-1}, γ_{k_1} and γ_{k_1,k_2} are a families of positive reals (not depending on n) satisfying $\sum_{k_1=1}^{\infty}\gamma_{k_1}k_1^2 + \sum_{k_1,k_2=1}^{\infty}\gamma_{k_1,k_2}k_1k_2 < \infty$.

In the examples listed in the introduction, $\boldsymbol{F}^{(n)}$ satisfies a more specific form that can be used to give more primitive conditions for Assumption 3.1. The following remark discusses this case.

Remark 3.2 ($\boldsymbol{F}^{(n)}$ as a function of momentum variables). Let $\{\boldsymbol{g}_\ell^{(n)}\}_{\ell=1}^L$ be L two times continuously differentiable functions $\mathcal{D} \to \mathbb{R}^d$, uniformly bounded along with two derivatives, and let $\boldsymbol{\Phi}(\boldsymbol{m}_1, \ldots, \boldsymbol{m}_L) \colon (\mathbb{R}^d)^L \to \mathbb{R}^d$ be a fixed two times continuously differentiable function, uniformly bounded along with two derivatives. Let $\{\beta_\ell\}_{\ell=1}^L$ be fixed reals in $[0, 1)$, and $\{b_\ell^{(n+1)}\}_{\ell=1}^L$ be L bounded nonnegative sequences of reals (for $n \in \mathbb{Z}_{\geq 0}$). If the function $\boldsymbol{F}^{(n)}$ has the specific form

$$\boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)}) := \boldsymbol{\Phi}(\boldsymbol{m}_1^{(n+1)}, \ldots, \boldsymbol{m}_L^{(n+1)}),$$

$$\text{where } \boldsymbol{m}_\ell^{(n+1)} := b_\ell^{(n+1)} \sum_{k=0}^{n} \beta_\ell^k \boldsymbol{g}_\ell^{(n-k)}(\boldsymbol{\theta}^{(n-k)}), \tag{10}$$

then it satisfies Assumption 3.1 (Lemma D.1). In the full-batch case, the same holds true except $\boldsymbol{g}_\ell^{(n)} \equiv \boldsymbol{g}_\ell$ are not allowed to depend on n.

Remark 3.2 considers the special case of optimization methods based on momentum variables. For instance, in the case of AdamW (Example 1.3), Remark 3.2 applies with $L = 3$,

$$\boldsymbol{g}_1(\boldsymbol{\theta}) = \nabla\mathcal{L}(\boldsymbol{\theta}), \quad \boldsymbol{g}_2(\boldsymbol{\theta}) = \left(\nabla\mathcal{L}(\boldsymbol{\theta})\right)^2, \quad \boldsymbol{g}_3(\boldsymbol{\theta}) = \boldsymbol{\theta},$$

$$b_1^{(n+1)} = \frac{1-\beta_1}{1-\beta_1^{n+1}} \to b_1 = 1 - \beta_1,$$

$$b_2^{(n+1)} = \frac{1-\beta_2}{1-\beta_2^{n+1}} \to b_2 = 1 - \beta_2,$$

$$b_3^{(n+1)} \equiv b_3 = \lambda, \quad \beta_3 = 0.$$

In the case of Lion-$\mathcal{K}$ (Example 1.5), the remark applies with $L = 3$,

$$\boldsymbol{g}_1(\boldsymbol{\theta}) = -\nabla\mathcal{L}(\boldsymbol{\theta}), \quad \boldsymbol{g}_2(\boldsymbol{\theta}) = -\nabla\mathcal{L}(\boldsymbol{\theta}), \quad \boldsymbol{g}_3(\boldsymbol{\theta}) = \boldsymbol{\theta},$$

$$\beta_1 = \rho_2, \quad \beta_2 = 0, \quad \beta_3 = 0,$$

$$b_1^{(n+1)} \equiv b_1 = (1-\rho_2)\frac{\rho_1}{\rho_2},$$

$$b_2^{(n+1)} \equiv b_2 = 1 - \frac{\rho_1}{\rho_2},$$

$$b_3^{(n+1)} \equiv b_3 = \lambda.$$

We used the letter ρ when defining the Lion iteration to avoid confusion with the β in the definition of momentum variables.

3.1 Deriving the Memoryless Approximation

By Taylor expansion with the Lagrange remainder,

$$F_r^{(n)}(\boldsymbol{\theta}^{(n)}, \dots, \boldsymbol{\theta}^{(0)}) - F_r^{(n)}(\boldsymbol{\theta}^{(n)}, \dots, \boldsymbol{\theta}^{(n)})$$

$$= \sum_{k=1}^{n} \left(\boldsymbol{\theta}^{(n-k)} - \boldsymbol{\theta}^{(n)}\right)^{\mathsf{T}} \frac{\partial F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k)}}(\boldsymbol{\theta}^{(n)}, \dots, \boldsymbol{\theta}^{(n)})$$

$$+ \frac{1}{2} \sum_{k_1,k_2=1}^{n} \left(\boldsymbol{\theta}^{(n-k_1)} - \boldsymbol{\theta}^{(n)}\right)^{\mathsf{T}} \frac{\partial^2 F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k_1)} \partial \boldsymbol{\theta}^{(n-k_2)}}(\boldsymbol{\zeta})\left(\boldsymbol{\theta}^{(n-k_2)} - \boldsymbol{\theta}^{(n)}\right)$$

$$= \sum_{k=1}^{n} \left(\boldsymbol{\theta}^{(n-k)} - \boldsymbol{\theta}^{(n)}\right)^{\mathsf{T}} \frac{\partial F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k)}}(\boldsymbol{\theta}^{(n)}, \dots, \boldsymbol{\theta}^{(n)}) + O(h^2), \tag{11}$$

where $\boldsymbol{\zeta}$ is some point on the segment between $\left(\boldsymbol{\theta}^{(n)}, \dots, \boldsymbol{\theta}^{(0)}\right)$ and $\left(\boldsymbol{\theta}^{(n)}, \dots, \boldsymbol{\theta}^{(n)}\right)$; in the last step we used Eq. (9), $\boldsymbol{\theta}^{(n-k_1)} - \boldsymbol{\theta}^{(n)} = O(k_1 h)$, and $\boldsymbol{\theta}^{(n-k_2)} - \boldsymbol{\theta}^{(n)} = O(k_2 h)$.

Next, write

$$\boldsymbol{\theta}^{(n-k)} - \boldsymbol{\theta}^{(n)}$$

$$= \sum_{s=n-k}^{n-1} \left(\boldsymbol{\theta}^{(s)} - \boldsymbol{\theta}^{(s+1)}\right)$$

$$= h \sum_{s=n-k}^{n-1} \boldsymbol{F}^{(s)}(\boldsymbol{\theta}^{(s)}, \dots, \boldsymbol{\theta}^{(0)})$$

$$= h \sum_{s=n-k}^{n-1} \boldsymbol{F}^{(s)}(\boldsymbol{\theta}^{(n)}, \dots, \boldsymbol{\theta}^{(n)}) + O(k^2 h^2),$$

where in the last step we used $\boldsymbol{F}^{(s)}(\boldsymbol{\theta}^{(s)}, \dots, \boldsymbol{\theta}^{(0)}) - \boldsymbol{F}^{(s)}(\boldsymbol{\theta}^{(n)}, \dots, \boldsymbol{\theta}^{(n)}) = O((n-s)h)$, which follows from Taylor expansion and Eq. (8). Insert this into Eq. (11) and use Eq. (8) again to continue:

$$F_r^{(n)}(\boldsymbol{\theta}^{(n)}, \dots, \boldsymbol{\theta}^{(0)}) - F_r^{(n)}(\boldsymbol{\theta}^{(n)}, \dots, \boldsymbol{\theta}^{(n)})$$

$$= h \sum_{k=1}^{n} \frac{\partial F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k)}} (\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(n)})^{\mathsf{T}} \sum_{s=n-k}^{n-1} \boldsymbol{F}^{(s)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(n)}) + O(h^2).$$

We conclude that the original numerical iteration can be rewritten as

$$\boldsymbol{\theta}^{(n+1)} = \boldsymbol{\theta}^{(n)} - h\big[\boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)}) + \boldsymbol{M}^{(n)}(\boldsymbol{\theta}^{(n)})\big] + O(h^3),$$

where the linear in h correction function is defined as $\boldsymbol{M}^{(n)} = (M_1^{(n)}, \ldots, M_d^{(n)})^{\mathsf{T}}$ with

$$M_r^{(n)}(\boldsymbol{\theta}) := h \sum_{k=1}^{n} \frac{\partial F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k)}}(\boldsymbol{\theta})^{\mathsf{T}} \sum_{s=n-k}^{n-1} \boldsymbol{F}^{(s)}(\boldsymbol{\theta}). \tag{12}$$

The derivation of the memoryless iteration is now complete. To make it easier to apply it to our examples, we can write it more specifically as the following section shows.

3.2 $\boldsymbol{F}^{(n)}$ as a Function of Momentum Variables

Specializing to the setup of Remark 3.2, and for any $s, n \in \mathbb{Z}_{\geq 0}$, $\boldsymbol{F}^{(s)}(\boldsymbol{\theta}) = \boldsymbol{\Phi}\big(\bar{g}_1^{(s+1)}(\boldsymbol{\theta}), \ldots, \bar{g}_L^{(s+1)}(\boldsymbol{\theta})\big)$, where $\bar{g}_\ell^{(s+1)}(\boldsymbol{\theta}) := b_\ell^{(n+1)} \sum_{k=0}^{s} \beta_\ell^k g_\ell^{(n-k)}(\boldsymbol{\theta})$, and

$$\frac{\partial F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k)}}(\boldsymbol{\theta})$$
$$= \sum_{\ell=1}^{L} \sum_i b_\ell^{(n+1)} \beta_\ell^k \frac{\partial \Phi_r}{\partial m_{\ell;i}} \big(\bar{g}_1^{(n+1)}(\boldsymbol{\theta}), \ldots, \bar{g}_L^{(n+1)}(\boldsymbol{\theta})\big)^{\mathsf{T}} \nabla g_{\ell;i}^{(n-k)}(\boldsymbol{\theta}).$$

Therefore, in this special case, the correction term in the memoryless iteration (4) is given by, for $r = 1, \ldots, d$,

$$M_r^{(n)}(\boldsymbol{\theta})$$
$$= h \sum_{\ell=1}^{L} \sum_i b_\ell^{(n+1)} \frac{\partial \Phi_r}{\partial m_{\ell;i}}\big(\bar{g}_1^{(n+1)}(\boldsymbol{\theta}), \ldots, \bar{g}_L^{(n+1)}(\boldsymbol{\theta})\big)$$
$$\times \sum_{k=1}^{n} \beta_\ell^k \nabla g_{\ell;i}^{(n-k)}(\boldsymbol{\theta})^{\mathsf{T}} \sum_{s=n-k}^{n-1} \boldsymbol{\Phi}\big(\bar{g}_1^{(s+1)}(\boldsymbol{\theta}), \ldots, \bar{g}_L^{(s+1)}(\boldsymbol{\theta})\big).$$

In the full-batch case $g_\ell^{(n)}(\boldsymbol{\theta}) \equiv g_\ell(\boldsymbol{\theta})$, this can be simplified further. Let us assume $b_\ell^{(n+1)} \xrightarrow[n \to \infty]{} b_\ell$, where b_ℓ is constant in n. Then $\bar{g}_\ell^{(n+1)}(\boldsymbol{\theta})$ also become constant in n: specifically, they settle to $\bar{g}_\ell(\boldsymbol{\theta}) := b_\ell(1 - \beta_\ell)^{-1} g_\ell(\boldsymbol{\theta})$. Lemma D.2 then implies that the iteration becomes close to

$$\boldsymbol{\theta}^{(n+1)} = \boldsymbol{\theta}^{(n)} - h\big[\boldsymbol{\Phi}(\bar{g}_1(\boldsymbol{\theta}^{(n)}), \ldots, \bar{g}_L(\boldsymbol{\theta}^{(n)})) + \boldsymbol{M}^{(n)}(\boldsymbol{\theta})\big]$$

with

$$M_r^{(n)}(\boldsymbol{\theta})$$
$$= h \sum_{\ell=1}^{L} \sum_i \frac{b_\ell \beta_\ell}{(1 - \beta_\ell)^2} \frac{\partial \Phi_r}{\partial m_{\ell;i}}\big(\bar{g}_1(\boldsymbol{\theta}^{(n)}), \ldots, \bar{g}_L(\boldsymbol{\theta}^{(n)})\big)$$
$$\times \nabla g_{\ell;i}(\boldsymbol{\theta}^{(n)})^{\mathsf{T}} \boldsymbol{\Phi}\big(\bar{g}_1(\boldsymbol{\theta}^{(n)}), \ldots, \bar{g}_L(\boldsymbol{\theta}^{(n)})\big).$$

Correction terms for all examples are provided in Appendix C.

3.3 Approximation Bound

An argument similar to the derivation in Section 3.1 can be made to obtain the following result.

Theorem 3.3 (Memoryless approximation: 1-step error bound). *Under Assumption 3.1, there exists a discrete memoryless iteration $\{\tilde{\boldsymbol{\theta}}^{(n)}\}_{n=0}^{\infty}$ satisfying (4) with initial condition $\tilde{\boldsymbol{\theta}}^{(0)} = \boldsymbol{\theta}^{(0)}$, correction function defined in Eq. (12), and a constant C_1 not depending on h, such that*

$$\sup_{n \in \mathbb{Z}_{\geq 0}} \left\| \tilde{\boldsymbol{\theta}}^{(n+1)} - \tilde{\boldsymbol{\theta}}^{(n)} + h\boldsymbol{F}^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(0)}) \right\|_{\infty} \leqslant C_1 h^3.$$

The proof is available in Appendix A.

The importance of this one-step approximation result is that it allows to bound the global error between the memoryfull iteration $\boldsymbol{\theta}^{(n)}$ and memoryless iteration $\tilde{\boldsymbol{\theta}}^{(n)}$ on a finite time horizon.

Corollary 3.4 (General momentum methods: global error bound on a finite time horizon). *In the setting of Theorem 3.3, let $\{\boldsymbol{\theta}^{(n)}\}_{n \in \mathbb{Z}_{\geq 0}}$ be the sequence of vectors generated by the iteration in Eq. (1) with initial condition $\boldsymbol{\theta}^{(0)}$. Let $T \geq 0$ be a fixed time horizon. Then there exists a constant C_2, depending on T but independent of h, such that $\max_{n \in [0:\lfloor T/h \rfloor]} \left\| \boldsymbol{\theta}^{(n)} - \tilde{\boldsymbol{\theta}}^{(n)} \right\|_{\infty} \leqslant C_2 h^2$.*

The proof is in Appendix B.

4 Application: Memory Anti-Regularizes AdamW but not Lion

We first study AdamW with memory by an application of Theorem 3.3 and Corollary 3.4. Neglecting coefficients decaying to zero exponentially fast, we have

$$\boldsymbol{\theta}^{(n+1)} = (1 - h\lambda)\boldsymbol{\theta}^{(n)} - h\left(\underbrace{\frac{\nabla\mathcal{L}(\boldsymbol{\theta}^{(n)})}{\sqrt{\left(\nabla\mathcal{L}(\boldsymbol{\theta}^{(n)})\right)^2 + \varepsilon}}}_{\approx \operatorname{sign}(\nabla\mathcal{L}(\boldsymbol{\theta}^{(n)}))} + \boldsymbol{M}^{(n)}(\boldsymbol{\theta}^{(n)}) \right),$$

where $\boldsymbol{M}^{(n)}(\boldsymbol{\theta})$ is given by

$$h\left(\frac{\beta_1(1-\beta_1)^{-1} - \beta_2(1-\beta_2)^{-1}}{\left(\left|\nabla\mathcal{L}(\boldsymbol{\theta})\right|^2 + \varepsilon\right)^{1/2}} + \frac{\varepsilon\beta_2(1-\beta_2)^{-1}}{\left(\left|\nabla\mathcal{L}(\boldsymbol{\theta})\right|^2 + \varepsilon\right)^{3/2}} \right) \left(\nabla\|\nabla\mathcal{L}(\boldsymbol{\theta})\|_{1,\varepsilon} + \lambda\nabla^2\mathcal{L}(\boldsymbol{\theta})\boldsymbol{\theta} \right).$$

Here $\|\cdot\|_{1,\varepsilon}$ is the perturbed one-norm defined as $\|\boldsymbol{v}\|_{1,\varepsilon} := \sum_{i=1}^{d} \sqrt{v_i^2 + \varepsilon}$. Taking ε to zero, we can write this in the form of preconditioned gradient descent (with decoupled weight decay):

$$\boldsymbol{\theta}^{(n+1)} = (1 - \lambda h)\boldsymbol{\theta}^{(n)} - h\frac{\nabla\tilde{\mathcal{L}}(\boldsymbol{\theta}^{(n)})}{\left|\left(\nabla\mathcal{L}(\boldsymbol{\theta}^{(n)})\right)\right|},$$

where

$$\tilde{\mathcal{L}}(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) + h\left(\frac{\beta_1}{1-\beta_1} - \frac{\beta_2}{1-\beta_2} \right)\|\nabla\mathcal{L}(\boldsymbol{\theta})\|_1 + h\lambda\left(\frac{\beta_1}{1-\beta_1} - \frac{\beta_2}{1-\beta_2} \right)\left(\nabla\mathcal{L}(\boldsymbol{\theta})^\mathsf{T}\boldsymbol{\theta} - \mathcal{L}(\boldsymbol{\theta}) \right)$$

is the modified loss. We see that the correction term $\beta_1/(1-\beta_1)\|\nabla\mathcal{L}(\boldsymbol{\theta})\|_1$, coming from exponentially averaging the gradients, penalizes the one-norm of the $\nabla\mathcal{L}$, acting as a regularizer, but the correction term $-\beta_2/(1-\beta_2)\|\nabla\mathcal{L}(\boldsymbol{\theta})\|_1$, coming from exponentially averaging the squared gradient components, anti-penalizes this one-norm and acts as an anti-regularizer. Moreover, if $\beta_1 < \beta_2$ (essentially always in practice), the anti-regularization is stronger than the regularization. Thus, *memory anti-regularizes AdamW*.

Consider now Lion-$\mathcal{K}$ (Example 1.5). Neglecting terms going to zero exponentially fast as $n \to \infty$, the memoryless iteration is

$$\boldsymbol{\theta}^{(n+1)} = (1 - h\lambda)\boldsymbol{\theta}^{(n)} - h\left[-\nabla\mathcal{K}\left(-\nabla\mathcal{L}(\boldsymbol{\theta}^{(n)})\right) + \boldsymbol{M}^{(n)}(\boldsymbol{\theta}^{(n)}) \right],$$

where

$$\boldsymbol{M}^{(n)}(\boldsymbol{\theta}) = -h\frac{\rho_1}{1-\rho_2}\nabla^2\mathcal{K}(-\nabla\mathcal{L}(\boldsymbol{\theta}))\nabla^2\mathcal{L}(\boldsymbol{\theta})\left[\nabla\mathcal{K}(-\nabla\mathcal{L}(\boldsymbol{\theta})) - \lambda\boldsymbol{\theta} \right].$$

9

As mentioned above, ordinary Lion is recovered by setting $\mathcal{K}(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$. This function is not differentiable, so let us replace it with the smooth convex approximation $\|\boldsymbol{x}\|_{1,\varepsilon}$, where ε is a small positive constant. The results of Section 3 can be applied, and the memoryless iteration is

$$\boldsymbol{\theta}^{(n+1)} = (1 - \lambda h)\boldsymbol{\theta}^{(n)} - h\left[\frac{\nabla\mathcal{L}(\boldsymbol{\theta}^{(n)})}{\left(|\nabla\mathcal{L}(\boldsymbol{\theta}^{(n)})|^2 + \varepsilon\right)^{1/2}} + \boldsymbol{M}^{(n)}(\boldsymbol{\theta}^{(n)})\right],$$

where

$$M_r^{(n)}(\boldsymbol{\theta}) = h\frac{\rho_1}{1 - \rho_2}\frac{\varepsilon}{\left(|\nabla_r\mathcal{L}(\boldsymbol{\theta})|^2 + \varepsilon\right)^{3/2}}\nabla_r\left[\|\nabla\mathcal{L}(\boldsymbol{\theta})\|_{1,\varepsilon} + \lambda(\nabla\mathcal{L}(\boldsymbol{\theta})^{\mathsf{T}}\boldsymbol{\theta} - \mathcal{L}(\boldsymbol{\theta}))\right].$$

This term is small as long as ε is small. Therefore, better generalization of Lion on a number of tasks (X. Chen et al., 2023) may be attributed to the fact that memory does *not* anti-regularize Lion (while it does anti-regularize AdamW).

5 Further Implications

5.1 Modified Equations

We have taken a very general algorithm (1) and, assuming smoothness and that memory decays sufficiently fast, converted it to a memoryless iteration $\boldsymbol{\theta}^{(n+1)} = \boldsymbol{\theta}^{(n)} - h\left[\boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)}) + \boldsymbol{M}^{(n)}(\boldsymbol{\theta}^{(n)})\right] + O(h^3)$. In this section we find an ODE in the form $\dot{\boldsymbol{\theta}} = \boldsymbol{G}_h(\boldsymbol{\theta})$ whose continuous solution, with initial condition $\boldsymbol{\theta}^{(0)}$, will approximate the memoryless iteration established in Section 3. Let us derive $\boldsymbol{G}_h(\boldsymbol{\theta})$ in the form of a power series $\boldsymbol{G}_1(\boldsymbol{\theta}) + h\boldsymbol{G}_2(h) + O(h^2)$, where $O(h^2)$ means "terms of order at least two in h". Relating the iteration number n of a discrete iteration and the time point $t = nh$ on a continuous trajectory, we would like the continuous trajectory to satisfy the same one-step relation as the discrete iteration, up to $O(h^3)$:

$$\boldsymbol{\theta}((n+1)h) = \boldsymbol{\theta}(nh) - h\left[\boldsymbol{F}^{(n)}(\boldsymbol{\theta}(nh), \ldots, \boldsymbol{\theta}(nh)) + \boldsymbol{M}^{(n)}(\boldsymbol{\theta}(nh))\right] + O(h^3).$$

In fact, we will ensure it is true for nh replaced by any t:

$$\boldsymbol{\theta}(t + h) = \boldsymbol{\theta}(t) - h\left[\boldsymbol{F}^{(n)}(\boldsymbol{\theta}(t), \ldots, \boldsymbol{\theta}(t)) + \boldsymbol{M}^{(n)}(\boldsymbol{\theta}(t))\right] + O(h^3). \tag{13}$$

But, using a Taylor expansion, and recalling that we are finding the trajectory satisfying $\dot{\boldsymbol{\theta}}(t) = \boldsymbol{G}_h(\boldsymbol{\theta}(t))$, hence $\ddot{\boldsymbol{\theta}}(t) = \nabla\boldsymbol{G}_h(\boldsymbol{\theta}(t))\dot{\boldsymbol{\theta}}(t)$, we have

$$\begin{aligned}
\boldsymbol{\theta}(t + h) &= \boldsymbol{\theta}(t) + h\dot{\boldsymbol{\theta}}(t) + \frac{h^2}{2}\ddot{\boldsymbol{\theta}}(t) + O(h^3) \\
&= \boldsymbol{\theta}(t) + h\left\{\boldsymbol{G}_1(\boldsymbol{\theta}(t)) + h\boldsymbol{G}_2(\boldsymbol{\theta}(t)) + O(h^2)\right\} \\
&\quad + \frac{h^2}{2}\left\{\nabla\boldsymbol{G}_1(\boldsymbol{\theta}(t))\boldsymbol{G}_1(\boldsymbol{\theta}(t)) + O(h)\right\} + O(h^3) \\
&= \boldsymbol{\theta}(t) + h\boldsymbol{G}_1(\boldsymbol{\theta}(t)) \\
&\quad + h^2\left\{\boldsymbol{G}_2(\boldsymbol{\theta}(t)) + \frac{\nabla\boldsymbol{G}_1(\boldsymbol{\theta}(t))\boldsymbol{G}_1(\boldsymbol{\theta}(t))}{2}\right\} + O(h^3).
\end{aligned}$$

In order to match (13), we need to have

$$\begin{aligned}
\boldsymbol{G}_1(\boldsymbol{\theta}) &= -\boldsymbol{F}^{(n)}(\boldsymbol{\theta}, \ldots, \boldsymbol{\theta}), \\
\boldsymbol{G}_2(\boldsymbol{\theta}) &= -\left(\boldsymbol{M}^{(n)}(\boldsymbol{\theta})/h + \frac{\nabla\boldsymbol{G}_1(\boldsymbol{\theta})\boldsymbol{G}_1(\boldsymbol{\theta})}{2}\right).
\end{aligned}$$

So, apart from the correction term coming from memory, the ODE $\dot{\boldsymbol{\theta}} = \boldsymbol{G}_1(\boldsymbol{\theta}) + h\boldsymbol{G}_2(h)$ derived has another term

$$h^2\frac{\nabla\boldsymbol{G}_1(\boldsymbol{\theta})\boldsymbol{G}_1(\boldsymbol{\theta})}{2}$$

arising from the fact that the algorithm is discrete.

For the example of full-batch heavy-ball momentum GD as in Section 2, where $\boldsymbol{G}_1(\boldsymbol{\theta}) = -(1 - \beta)^{-1}\nabla\mathcal{L}(\boldsymbol{\theta})$ (ignoring coefficients going to zero exponentially fast in n), this additional term is equal to $h^2(1-\beta)^{-2}\nabla\|\nabla\mathcal{L}(\boldsymbol{\theta})\|^2/4$, providing additional implicit regularization. We recover the ODE derived by Kovachki and Stuart (2021) and Ghosh et al. (2023):

$$\dot{\boldsymbol{\theta}} = -\frac{\nabla\mathcal{L}(\boldsymbol{\theta})}{1-\beta} - h\frac{1+\beta}{(1-\beta)^3}\frac{\nabla\|\nabla\mathcal{L}(\boldsymbol{\theta})\|^2}{4}.$$

5.2 Mini-Batch Training

In specific cases, it is possible to identify the additional implicit regularization that is introduced to the algorithm by noise, if mini-batch training is used as opposed to full-batch. Assume that the form of $\boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)})$ is given by

$$\boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)}) = \sum_{k=0}^{n}\beta^k\boldsymbol{g}^{(n-k)}(\boldsymbol{\theta}^{(n-k)}),$$

where the $\boldsymbol{g}^{(n-k)}(\cdot)$ functions are uniformly bounded along with their derivatives.

The correction term introduced by memory (12) is

$$M_r^{(n)}(\boldsymbol{\theta}) := h\sum_{k=1}^{n}\frac{\partial F_r^{(n)}}{\partial\boldsymbol{\theta}^{(n-k)}}(\boldsymbol{\theta}, \ldots, \boldsymbol{\theta})^{\mathsf{T}}\sum_{s=n-k}^{n-1}\boldsymbol{F}^{(s)}(\boldsymbol{\theta}, \ldots, \boldsymbol{\theta})$$

$$= h\beta\sum_{k=0}^{n-1}\beta^k\nabla g_r^{(n-1-k)}(\boldsymbol{\theta})^{\mathsf{T}}\sum_{l=1}^{k+1}\sum_{b=0}^{n-l}\beta^b\boldsymbol{g}^{(n-l-b)}(\boldsymbol{\theta}).$$

Assume that n is a (large) number of mini-batches in one epoch, and mini-batches are sampled randomly without replacement, with each permutation equally likely. Let us take the expectation of the correction term with respect to this randomness, that is, take the average over all re-orderings π of $(\boldsymbol{g}^{(0)}, \ldots, \boldsymbol{g}^{(n)})$:

$$\mathbb{E}\sum_{k=0}^{n-1}\beta^k\nabla g_r^{(n-1-k)}(\boldsymbol{\theta})^{\mathsf{T}}\sum_{l=1}^{k+1}\sum_{b=0}^{n-l}\beta^b\boldsymbol{g}^{(n-l-b)}(\boldsymbol{\theta})$$

$$:= \frac{1}{(n+1)!}\sum_{\pi}\sum_{k=0}^{n-1}\beta^k\nabla g_r^{(\pi(n-1-k))}(\boldsymbol{\theta})^{\mathsf{T}}$$

$$\times\sum_{l=1}^{k+1}\sum_{b=0}^{n-l}\beta^b\boldsymbol{g}^{(\pi(n-l-b))}(\boldsymbol{\theta}).$$

Note that $\mathbb{E}\nabla g_r^{(i)}(\boldsymbol{\theta})^{\mathsf{T}}\boldsymbol{g}^{(j)}(\boldsymbol{\theta})$ depends only on whether $i = j$ or $i \neq j$. Therefore,

$$\mathbb{E}[M_r^{(n)}(\boldsymbol{\theta})]/h = C_3(\beta)\mathbb{E}[\nabla g_r^{(1)}(\boldsymbol{\theta})^{\mathsf{T}}\boldsymbol{g}^{(1)}(\boldsymbol{\theta})]$$
$$+ C_4(\beta)\mathbb{E}[\nabla g_r^{(1)}(\boldsymbol{\theta})^{\mathsf{T}}\boldsymbol{g}^{(2)}(\boldsymbol{\theta})],$$

where $C_3(\beta)$ and $C_4(\beta)$ can be calculated as

$$C_3(\beta) := \beta\sum_{b=0}^{n-1}\beta^b\sum_{l=1}^{b+1}\beta^{b+1-l} \xrightarrow[n\to\infty]{} \frac{\beta}{(1-\beta)^2(1+\beta)},$$

$$C_4(\beta) := \beta\sum_{k=0}^{n-1}\beta^k\sum_{l=1}^{k+1}\sum_{b=0}^{n-l}\beta^b - C_3(\beta) \xrightarrow[n\to\infty]{} \frac{2\beta^2}{(1-\beta)^3(1+\beta)}.$$

We can simplify

$$\mathbb{E}[\nabla g_r^{(1)}(\boldsymbol{\theta})^{\mathsf{T}}\boldsymbol{g}^{(2)}(\boldsymbol{\theta})] = \frac{1}{(n+1)n}\sum_{i\neq j}\nabla g_r^{(i)}(\boldsymbol{\theta})^{\mathsf{T}}\boldsymbol{g}^{(j)}(\boldsymbol{\theta})$$

11

$$= \nabla g_r(\boldsymbol{\theta})^\mathsf{T} \boldsymbol{g}(\boldsymbol{\theta}) + o_n(1),$$

where $\boldsymbol{g}(\boldsymbol{\theta}) = \mathbb{E}\boldsymbol{g}^{(1)}(\boldsymbol{\theta}) = (n+1)^{-1}\sum_{k=0}^{n+1}\boldsymbol{g}^{(k)}(\boldsymbol{\theta})$ is the average of $\{\boldsymbol{g}^{(k)}(\boldsymbol{\theta})\}$, $o_n(1)$ tends to zero as $n \to \infty$.

So, for large n we can write

$$
\begin{aligned}
\mathbb{E}&[M_r^{(n)}(\boldsymbol{\theta})]/h \\
&\approx C_3(\beta)\mathbb{E}[\nabla g_r^{(1)}(\boldsymbol{\theta})^\mathsf{T}\boldsymbol{g}^{(1)}(\boldsymbol{\theta})] + C_4(\beta)\nabla g_r(\boldsymbol{\theta})^\mathsf{T}\boldsymbol{g}(\boldsymbol{\theta}) \\
&= \big(C_3(\beta) + C_4(\beta)\big)\nabla g_r(\boldsymbol{\theta})^\mathsf{T}\boldsymbol{g}(\boldsymbol{\theta}) \\
&\quad + C_3(\beta)\mathbb{E}[(\nabla g_r^{(1)}(\boldsymbol{\theta}) - \nabla g_r(\boldsymbol{\theta}))^\mathsf{T}(\boldsymbol{g}^{(1)}(\boldsymbol{\theta}) - \boldsymbol{g}(\boldsymbol{\theta}))] \\
&\approx \frac{\beta}{(1-\beta)^3}\nabla g_r(\boldsymbol{\theta})^\mathsf{T}\boldsymbol{g}(\boldsymbol{\theta}) \\
&\quad + C_3(\beta)\mathbb{E}[(\nabla g_r^{(1)}(\boldsymbol{\theta}) - \nabla g_r(\boldsymbol{\theta}))^\mathsf{T}(\boldsymbol{g}^{(1)}(\boldsymbol{\theta}) - \boldsymbol{g}(\boldsymbol{\theta}))].
\end{aligned}
$$

The second term can be interpreted as additional implicit regularization by noise.

For example, take $\boldsymbol{g}^{(k)}(\boldsymbol{\theta}) = \nabla\mathcal{L}^{(k)}(\boldsymbol{\theta})$ the kth minibatch loss. Then we obtained that "on average" mini-batch GD with momentum is given by the iteration like (6), except the modified loss has an additional term:

$$\tilde{\mathcal{L}}(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) + \frac{h\beta}{2(1-\beta)^2}\|\nabla\mathcal{L}(\boldsymbol{\theta})\|^2 + \frac{h\beta}{2(1-\beta)(1+\beta)}\mathbb{E}\|\nabla\mathcal{L}^{(1)}(\boldsymbol{\theta}) - \nabla\mathcal{L}(\boldsymbol{\theta})\|^2.$$

Acknowledgments

References

Andriushchenko, Maksym et al. (2023). "A Modern Look at the Relationship between Sharpness and Generalization". *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. Proceedings of Machine Learning Research. PMLR, pp. 840–902 (cit. on p. 6).

Anil, Rohan et al. (2019). "Memory Efficient Adaptive Optimization". *Advances in Neural Information Processing Systems*. Vol. 32 (cit. on p. 2).

Arora, Sanjeev, Nadav Cohen, et al. (2019). "Implicit Regularization in Deep Matrix Factorization". *Advances in Neural Information Processing Systems*. Vol. 32 (cit. on p. 4).

Arora, Sanjeev, Zhiyuan Li, and Abhishek Panigrahi (2022). "Understanding Gradient Descent on the Edge of Stability in Deep Learning". *Proceedings of the 39th International Conference on Machine Learning*. Vol. 162. Proceedings of Machine Learning Research. PMLR, pp. 948–1024 (cit. on p. 4).

Barakat, Anas and Pascal Bianchi (2021). "Convergence and Dynamical Behavior of the ADAM Algorithm for Nonconvex Stochastic Optimization". *SIAM Journal on Optimization* 31.1, pp. 244–274 (cit. on p. 4).

Barrett, David and Benoit Dherin (2021). "Implicit Gradient Regularization". *International Conference on Learning Representations* (cit. on pp. 4, 6).

Cattaneo, Matias D., Jason Matthew Klusowski, and Boris Shigida (2024). "On the Implicit Bias of Adam". *Proceedings of the 41st International Conference on Machine Learning*. Vol. 235. Proceedings of Machine Learning Research. PMLR, pp. 5862–5906 (cit. on pp. 4, 16).

Chen, Lizhang et al. (2024). "Lion Secretly Solves a Constrained Optimization: As Lyapunov Predicts". *The Twelfth International Conference on Learning Representations* (cit. on pp. 3, 4).

Chen, Xiangning et al. (2023). "Symbolic Discovery of Optimization Algorithms". *Advances in Neural Information Processing Systems*. Vol. 36, pp. 49205–49233 (cit. on pp. 1, 3, 10).

Choi, Dami et al. (2020). "On Empirical Comparisons of Optimizers for Deep Learning". *arXiv preprint arXiv:1910.05446* (cit. on p. 2).

Chowdhery, Aakanksha et al. (2023). "PaLM: Scaling Language Modeling with Pathways". *Journal of Machine Learning Research* 24.240, pp. 1–113 (cit. on p. 1).

Damian, Alex, Tengyu Ma, and Jason D Lee (2021). "Label Noise SGD Provably Prefers Flat Global Minimizers". *Advances in Neural Information Processing Systems*. Vol. 34, pp. 27449–27461 (cit. on p. 4).

Damian, Alex, Eshaan Nichani, and Jason D. Lee (2023). "Self-Stabilization: The Implicit Bias of Gradient Descent at the Edge of Stability". *The Eleventh International Conference on Learning Representations* (cit. on p. 4).

DeepSeek-AI et al. (2024). "DeepSeek-V3 Technical Report". *arXiv preprint arXiv:2412.19437* (cit. on pp. 1, 2).

Dozat, Timothy (2016). "Incorporating Nesterov Momentum into Adam". *ICLR Workshops* (cit. on p. 3).

Ernst Hairer, Christian Lubich and Gerhard Wanner (2006). *Geometric numerical integration*. 2nd ed. Springer-Verlag, Berlin. ISBN: 3-540-30663-3 (cit. on p. 4).

Farazmand, Mohammad (2020). "Multiscale Analysis of Accelerated Gradient Methods". *SIAM Journal on Optimization* 30.3, pp. 2337–2354 (cit. on p. 4).

Foret, Pierre et al. (2021). "Sharpness-aware Minimization for Efficiently Improving Generalization". *International Conference on Learning Representations* (cit. on p. 6).

Ghosh, Avrajit et al. (2023). "Implicit regularization in Heavy-ball momentum accelerated stochastic gradient descent". *The Eleventh International Conference on Learning Representations* (cit. on pp. 4, 6, 11, 16).

Grattafiori, Aaron et al. (2024). "The Llama 3 Herd of Models". *arXiv preprint arXiv:2407.21783* (cit. on pp. 1, 2).

Gunasekar, Suriya, Jason Lee, et al. (2018). "Characterizing Implicit Bias in Terms of Optimization Geometry". *Proceedings of the 35th International Conference on Machine Learning*. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 1832–1841 (cit. on p. 4).

Gunasekar, Suriya, Blake E Woodworth, et al. (2017). "Implicit Regularization in Matrix Factorization". *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc. (cit. on p. 4).

Ji, Ziwei and Matus Telgarsky (2018). "Risk and parameter convergence of logistic regression". *arXiv preprint arXiv:1803.07300* (cit. on p. 4).

— (2019). "The implicit bias of gradient descent on nonseparable data". *Proceedings of the Thirty-Second Conference on Learning Theory*. Vol. 99. Proceedings of Machine Learning Research. PMLR, pp. 1772–1798 (cit. on p. 4).

Kingma, Diederik P. and Jimmy Ba (2017). "Adam: A Method for Stochastic Optimization". *arXiv preprint arXiv:1412.6980* (cit. on pp. 1, 2).

Kovachki, Nikola B. and Andrew M. Stuart (2021). "Continuous Time Analysis of Momentum Methods". *Journal of Machine Learning Research* 22.17, pp. 1–40 (cit. on pp. 4, 11).

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". *Advances in Neural Information Processing Systems*. Vol. 25 (cit. on p. 1).

Kunstner, Frederik et al. (2023). "Noise Is Not the Main Factor Behind the Gap Between SGD and Adam on Transformers, but Sign Descent Might Be". *arXiv preprint arXiv:2304.13960* (cit. on p. 2).

Li, Qianxiao, Cheng Tai, and Weinan E (2017). "Stochastic Modified Equations and Adaptive Stochastic Gradient Algorithms". *Proceedings of the 34th International Conference on Machine Learning*. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 2101–2110 (cit. on p. 4).

Li, Yuanzhi, Colin Wei, and Tengyu Ma (2019). "Towards Explaining the Regularization Effect of Initial Large Learning Rate in Training Neural Networks". *Advances in Neural Information Processing Systems*. Vol. 32 (cit. on p. 4).

Li, Zhiyuan, Tianhao Wang, and Sanjeev Arora (2022). "What Happens after SGD Reaches Zero Loss? –A Mathematical Framework". *International Conference on Learning Representations* (cit. on p. 4).

Liu, Liyuan et al. (2020). "Understanding the Difficulty of Training Transformers". *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5747–5763 (cit. on p. 2).

Loshchilov, Ilya and Frank Hutter (2019). "Decoupled Weight Decay Regularization". *International Conference on Learning Representations* (cit. on pp. 1, 2).

Ma, Chao, Lei Wu, and Weinan E (2022). "A Qualitative Study of the Dynamic Behavior for Adaptive Gradient Algorithms". *Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference.* Vol. 145. Proceedings of Machine Learning Research. PMLR, pp. 671–692 (cit. on p. 4).

Malladi, Sadhika et al. (2022). "On the SDEs and Scaling Rules for Adaptive Gradient Algorithms". *Advances in Neural Information Processing Systems.* Vol. 35, pp. 7697–7711 (cit. on p. 4).

Miyagawa, Taiki (2022). "Toward Equation of Motion for Deep Neural Networks: Continuous-time Gradient Descent and Discretization Error Analysis". *Advances in Neural Information Processing Systems.* Vol. 35, pp. 37778–37791 (cit. on p. 4).

Nacson, Mor Shpigel, Jason Lee, et al. (2019). "Convergence of Gradient Descent on Separable Data". *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics.* Vol. 89. Proceedings of Machine Learning Research. PMLR, pp. 3420–3428 (cit. on p. 4).

Nacson, Mor Shpigel, Nathan Srebro, and Daniel Soudry (2019). "Stochastic Gradient Descent on Separable Data: Exact Convergence with a Fixed Learning Rate". *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics.* Vol. 89. Proceedings of Machine Learning Research. PMLR, pp. 3051–3059 (cit. on p. 4).

Nesterov, Yurii (1983). "A method for solving the convex programming problem with convergence rate $O\left(\frac{1}{k^2}\right)$". *Dokl. Akad. Nauk SSSR.* Vol. 269, pp. 543–547 (cit. on p. 2).

Polyak, B.T. (1964). "Some methods of speeding up the convergence of iteration methods". *USSR Computational Mathematics and Mathematical Physics* 4.5, pp. 1–17. ISSN: 0041-5553 (cit. on p. 2).

Qian, Qian and Xiaoyuan Qian (2019). "The Implicit Bias of AdaGrad on Separable Data". *Advances in Neural Information Processing Systems.* Vol. 32 (cit. on p. 4).

Rosca, Mihaela et al. (2023). "On a continuous time model of gradient descent dynamics and instability in deep learning". *Transactions on Machine Learning Research.* ISSN: 2835-8856 (cit. on p. 4).

Shazeer, Noam and Mitchell Stern (2018). "Adafactor: Adaptive Learning Rates with Sublinear Memory Cost". *Proceedings of the 35th International Conference on Machine Learning.* Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 4596–4604 (cit. on p. 1).

Smith, Samuel L et al. (2021). "On the Origin of Implicit Regularization in Stochastic Gradient Descent". *International Conference on Learning Representations* (cit. on pp. 4, 6).

Soudry, Daniel, Elad Hoffer, and Nathan Srebro (2018). "The Implicit Bias of Gradient Descent on Separable Data". *International Conference on Learning Representations* (cit. on p. 4).

Tieleman, Tijmen, Geoffrey Hinton, et al. (2012). "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude". *COURSERA: Neural networks for machine learning* 4.2, pp. 26–31 (cit. on p. 1).

Wang, Bohan et al. (2022). "Does Momentum Change the Implicit Regularization on Separable Data?" *Advances in Neural Information Processing Systems.* Vol. 35, pp. 26764–26776 (cit. on p. 4).

Wen, Kaiyue, Tengyu Ma, and Zhiyuan Li (2022). "How Does Sharpness-Aware Minimization Minimizes Sharpness?" *OPT 2022: Optimization for Machine Learning (NeurIPS 2022 Workshop)* (cit. on p. 4).

Xie, Shuo and Zhiyuan Li (2024). "Implicit Bias of AdamW: ℓ_∞-Norm Constrained Optimization". *Proceedings of the 41st International Conference on Machine Learning.* Vol. 235. Proceedings of Machine Learning Research. PMLR, pp. 54488–54510 (cit. on p. 4).

Zhang, Jingzhao et al. (2020). "Why are Adaptive Methods Good for Attention Models?" *Advances in Neural Information Processing Systems.* Vol. 33, pp. 15383–15393 (cit. on p. 2).

A   Proof of Theorem 3.3

Since, by the assumptions of the theorem,

$$
\begin{aligned}
\tilde{\theta}_r^{(n+1)} - \tilde{\theta}_r^{(n)} &= -hF_r^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)}) \\
&\quad - h^2 \sum_{k=1}^{n} \frac{\partial F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k)}}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)})^{\mathsf{T}} \underline{\sum_{s=n-k}^{n-1} \boldsymbol{F}^{(s)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)})},
\end{aligned}
\tag{14}
$$

we need to show that

$$
\begin{aligned}
&F_r^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(0)}) - F_r^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)}) \\
&= h \sum_{k=1}^{n} \frac{\partial F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k)}}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)})^{\mathsf{T}} \sum_{s=n-k}^{n-1} \boldsymbol{F}^{(s)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)}) + O(h^2).
\end{aligned}
\tag{15}
$$

By Taylor expansion with the Lagrange remainder,

$$
\begin{aligned}
&F_r^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(0)}) - F_r^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)}) \\
&= \sum_{k=1}^{n} \frac{\partial F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k)}}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)})^{\mathsf{T}} \big(\tilde{\boldsymbol{\theta}}^{(n-k)} - \tilde{\boldsymbol{\theta}}^{(n)}\big) \\
&\quad + \frac{1}{2} \sum_{k_1, k_2 = 1}^{n} \big(\tilde{\boldsymbol{\theta}}^{(n-k_1)} - \tilde{\boldsymbol{\theta}}^{(n)}\big)^{\mathsf{T}} \frac{\partial^2 F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k_1)} \partial \boldsymbol{\theta}^{(n-k_2)}}(\boldsymbol{\zeta}) \big(\tilde{\boldsymbol{\theta}}^{(n-k_2)} - \tilde{\boldsymbol{\theta}}^{(n)}\big) \\
&\overset{(a)}{=} \sum_{k=1}^{n} \frac{\partial F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k)}}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)})^{\mathsf{T}} \big(\tilde{\boldsymbol{\theta}}^{(n-k)} - \tilde{\boldsymbol{\theta}}^{(n)}\big) + O(h^2),
\end{aligned}
\tag{16}
$$

where $\boldsymbol{\zeta}$ is some point on the segment between $(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(0)})$ and $(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)})$; in (a) we used Eq. (9) and $\tilde{\boldsymbol{\theta}}^{(n-k_1)} - \tilde{\boldsymbol{\theta}}^{(n)} = O(k_1 h)$, $\tilde{\boldsymbol{\theta}}^{(n-k_2)} - \tilde{\boldsymbol{\theta}}^{(n)} = O(k_2 h)$.

Since the underlined term in Eq. (14) is $O(1)$, we have

$$
\begin{aligned}
\tilde{\boldsymbol{\theta}}^{(n-k)} - \tilde{\boldsymbol{\theta}}^{(n)} &= \sum_{s=n-k}^{n-1} \big(\tilde{\boldsymbol{\theta}}^{(s)} - \tilde{\boldsymbol{\theta}}^{(s+1)}\big) \\
&= h \sum_{s=n-k}^{n-1} \big\{ \boldsymbol{F}^{(s)}(\tilde{\boldsymbol{\theta}}^{(s)}, \ldots, \tilde{\boldsymbol{\theta}}^{(s)}) + O(h) \big\} \\
&= h \sum_{s=n-k}^{n-1} \boldsymbol{F}^{(s)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)}) + O(k^2 h^2),
\end{aligned}
$$

where in the last step we used $\boldsymbol{F}^{(s)}(\tilde{\boldsymbol{\theta}}^{(s)}, \ldots, \tilde{\boldsymbol{\theta}}^{(s)}) - \boldsymbol{F}^{(s)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)}) = O((n-s)h)$, which follows from Taylor expansion, Eq. (8) and $\tilde{\boldsymbol{\theta}}^{(n+1)} - \tilde{\boldsymbol{\theta}}^{(n)} = O(h)$. Combine this with Eq. (16) to get

$$
\begin{aligned}
&F_r^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(0)}) - F_r^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)}) \\
&= \sum_{k=1}^{n} \frac{\partial F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k)}}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)})^{\mathsf{T}} \bigg\{ h \sum_{s=n-k}^{n-1} \boldsymbol{F}^{(s)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)}) + O(k^2 h^2) \bigg\} + O(h^2) \\
&= h \sum_{k=1}^{n} \frac{\partial F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k)}}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)})^{\mathsf{T}} \sum_{s=n-k}^{n-1} \boldsymbol{F}^{(s)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(n)}) + O(h^2),
\end{aligned}
$$

which is (15), and the proof is complete.

B   Proof of Corollary 3.4

We follow a standard argument, e.g. Ghosh et al. (2023) and Cattaneo, Klusowski, and Shigida (2024). We prove the following claim by induction over $n \in \mathbb{Z}_{\geq 0}$:

$$\|\boldsymbol{\theta}^{(n)} - \tilde{\boldsymbol{\theta}}^{(n)}\|_\infty \leqslant d_1 e^{d_2 nh} h^2, \quad \|\boldsymbol{\theta}^{(n+1)} - \tilde{\boldsymbol{\theta}}^{(n+1)} - \boldsymbol{\theta}^{(n)} + \tilde{\boldsymbol{\theta}}^{(n)}\|_\infty \leqslant d_3 e^{d_2 nh} h^3,$$

where

$$d_1 = C_1, \quad d_2 = 1 + d \sum_{k=0}^\infty \gamma_k, \quad d_3 = C_1 d_2.$$

Because $nh \leqslant T$, Corollary 3.4 will follow.

Base: $n = 0$. It is indeed true that $\|\boldsymbol{\theta}^{(0)} - \tilde{\boldsymbol{\theta}}^{(0)}\|_\infty \leqslant d_1 h^2$ because the left-hand side is zero. It is indeed true that $\|\boldsymbol{\theta}^{(1)} - \tilde{\boldsymbol{\theta}}^{(1)} - \boldsymbol{\theta}^{(0)} + \tilde{\boldsymbol{\theta}}^{(0)}\|_\infty \leqslant d_3 h^3$ for the same reason.

Assume $n \in \mathbb{Z}_{\geq 1}$ and it is true that

$$\|\boldsymbol{\theta}^{(n')} - \tilde{\boldsymbol{\theta}}^{(n')}\|_\infty \leqslant d_1 e^{d_2 n'h} h^2, \quad \|\boldsymbol{\theta}^{(n'+1)} - \tilde{\boldsymbol{\theta}}^{(n'+1)} - \boldsymbol{\theta}^{(n')} + \tilde{\boldsymbol{\theta}}^{(n')}\|_\infty \leqslant d_3 e^{d_2 n'h} h^3.$$

for all $0 \leqslant n' \leqslant n - 1$. Then

$$\|\boldsymbol{\theta}^{(n)} - \tilde{\boldsymbol{\theta}}^{(n)}\|_\infty \leqslant \|\boldsymbol{\theta}^{(n-1)} - \tilde{\boldsymbol{\theta}}^{(n-1)}\|_\infty + \|\boldsymbol{\theta}^{(n)} - \tilde{\boldsymbol{\theta}}^{(n)} - \boldsymbol{\theta}^{(n-1)} + \tilde{\boldsymbol{\theta}}^{(n-1)}\|_\infty$$

by the triangle inequality,

$$\leqslant d_1 e^{d_2 (n-1)h} h^2 + d_3 e^{d_2 (n-1)h} h^3$$

by the induction hypothesis,

$$= d_1 \left(1 + \frac{d_3}{d_1} h \right) e^{d_2 (n-1)h} h^2 \leqslant d_1 (1 + d_2 h) e^{d_2 (n-1)h} h^2$$

by $d_3 \leqslant d_1 d_2$,

$$\leqslant d_1 e^{d_2 nh} h^2$$

by the inequality $1 + x \leqslant e^x$ for all $x \geq 0$.

Next, write

$$\boldsymbol{\theta}^{(n+1)} - \boldsymbol{\theta}^{(n)} - \tilde{\boldsymbol{\theta}}^{(n+1)} + \tilde{\boldsymbol{\theta}}^{(n)}$$
$$= -h \boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)}) - \{\tilde{\boldsymbol{\theta}}^{(n+1)} - \tilde{\boldsymbol{\theta}}^{(n)}\}$$
$$= h\big[\boldsymbol{F}^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(0)}) - \boldsymbol{F}^{(n)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)})\big] - \{\tilde{\boldsymbol{\theta}}^{(n+1)} - \tilde{\boldsymbol{\theta}}^{(n)} + h \boldsymbol{F}^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(0)})\}$$

Then

$$|\theta_r^{(n+1)} - \theta_r^{(n)} - \tilde{\theta}_r^{(n+1)} + \tilde{\theta}_r^{(n)}|$$
$$\leqslant h\big|F_r^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(0)}) - F_r^{(n)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)})\big| + \big|\tilde{\theta}_r^{(n+1)} - \tilde{\theta}_r^{(n)} + h F_r^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(0)})\big|$$
$$\leqslant h\big|F_r^{(n)}(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(0)}) - F_r^{(n)}(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)})\big| + C_1 h^3$$

by Theorem 3.3,

$$= h\left|\sum_{k=0}^n \frac{\partial F_r^{(n)}}{\partial \boldsymbol{\theta}^{(n-k)}}(\boldsymbol{\zeta})^\mathsf{T} \big(\tilde{\boldsymbol{\theta}}^{(n-k)} - \boldsymbol{\theta}^{(n-k)}\big)\right| + C_1 h^3,$$

where $\boldsymbol{\zeta}$ is a point on the segment between $(\tilde{\boldsymbol{\theta}}^{(n)}, \ldots, \tilde{\boldsymbol{\theta}}^{(0)})$ and $(\boldsymbol{\theta}^{(n)}, \ldots, \boldsymbol{\theta}^{(0)})$,

$$\leqslant hd \sum_{k=0}^{n} \gamma_k \|\tilde{\boldsymbol{\theta}}^{(n-k)} - \boldsymbol{\theta}^{(n-k)}\|_{\infty} + C_1 h^3$$

by (8) (recall that d is the dimension of $\boldsymbol{\theta}$),

$$\leqslant d_1 h^3 d \sum_{k=0}^{\infty} \gamma_k e^{d_2(n-k)h} + C_1 h^3$$

by the induction hypothesis and the bound on $\|\tilde{\boldsymbol{\theta}}^{(n)} - \boldsymbol{\theta}^{(n)}\|_{\infty}$ already proven

$$\leqslant \underbrace{\left(d_1 d \sum_{k=0}^{\infty} \gamma_k + C_1 \right)}_{\leqslant d_3} e^{d_2 nh} h^3$$

$$\leqslant d_3 e^{d_2 nh} h^3.$$

C   Correction Terms for All Examples

For GD with momentum (Example 1.1):

$$\boldsymbol{M}^{(n)}(\boldsymbol{\theta}) = \frac{h\beta}{2(1-\beta)^3} \nabla \|\nabla \mathcal{L}(\boldsymbol{\theta})\|^2.$$

For Nesterov's accelerated GD (Example 1.2):

$$\boldsymbol{M}^{(n)}(\boldsymbol{\theta}) = \frac{h\beta^2}{2(1-\beta)^3} \nabla \|\nabla \mathcal{L}(\boldsymbol{\theta})\|^2.$$

For AdamW (Example 1.3, also discussed in Section 4):

$$\boldsymbol{M}^{(n)}(\boldsymbol{\theta}) = h \left(\frac{\beta_1(1-\beta_1)^{-1} - \beta_2(1-\beta_2)^{-1}}{\left(|\nabla \mathcal{L}(\boldsymbol{\theta})|^2 + \varepsilon\right)^{1/2}} + \varepsilon \frac{\beta_2(1-\beta_2)^{-1}}{\left(|\nabla \mathcal{L}(\boldsymbol{\theta})|^2 + \varepsilon\right)^{3/2}} \right) \left(\nabla \|\nabla \mathcal{L}(\boldsymbol{\theta})\|_{1,\varepsilon} + \lambda \nabla^2 \mathcal{L}(\boldsymbol{\theta})\boldsymbol{\theta} \right).$$

For Nadam (Example 1.4):

$$\boldsymbol{M}^{(n)}(\boldsymbol{\theta}) = h \left(\frac{\beta_1^2(1-\beta_1)^{-1} - \beta_2(1-\beta_2)^{-1}}{\left(|\nabla \mathcal{L}(\boldsymbol{\theta})|^2 + \varepsilon\right)^{1/2}} + \varepsilon \frac{\beta_2(1-\beta_2)^{-1}}{\left(|\nabla \mathcal{L}(\boldsymbol{\theta})|^2 + \varepsilon\right)^{3/2}} \right) \left(\nabla \|\nabla \mathcal{L}(\boldsymbol{\theta})\|_{1,\varepsilon} + \lambda \nabla^2 \mathcal{L}(\boldsymbol{\theta})\boldsymbol{\theta} \right).$$

For Lion-$\mathcal{K}$ (Example 1.5, also discussed in Section 4):

$$\boldsymbol{M}^{(n)}(\boldsymbol{\theta}) = -h \frac{\rho_1}{1-\rho_2} \nabla^2 \mathcal{K}(-\nabla \mathcal{L}(\boldsymbol{\theta})) \nabla^2 \mathcal{L}(\boldsymbol{\theta}) \left[\nabla \mathcal{K}(-\nabla \mathcal{L}(\boldsymbol{\theta})) - \lambda \boldsymbol{\theta} \right].$$

D   Auxiliary Results

Lemma D.1 (Memory decays exponentially fast). *If $\boldsymbol{F}^{(n)}$ is a function of momentum variables as described in Remark 3.2, then for any n and $k \leqslant n$*

$$\max_{r,i} \left| \frac{\partial F_r^{(n)}}{\partial \theta_i^{(n-k)}} \right| \leqslant \gamma_k, \tag{17}$$

and similarly for any n and $k_1, k_2 \leqslant n$

$$\max_{r,i,j} \left| \frac{\partial^2 F_r^{(n)}}{\partial \theta_i^{(n-k_1)} \partial \theta_j^{(n-k_2)}} \right| \leqslant \gamma_{k_1,k_2}, \tag{18}$$

17

where $\{\gamma_k\}$ and $\{\gamma_{k_1,k_2}\}$ are sequences decaying exponentially fast: specifically,

$$\gamma_k := C_\gamma \{\max_\ell \beta_\ell\}^k, \quad \gamma_{k_1,k_2} := C_\gamma \{\max_\ell \beta_\ell\}^{k_1+k_2}$$

for some constant $C_\gamma > 0$.

Proof. It is easy to see (17) by taking the derivative:

$$\frac{\partial}{\partial \boldsymbol{\theta}^{(n-k)}} F_r^{(n)}(\boldsymbol{\theta}^{(n)}, \dots, \boldsymbol{\theta}^{(0)})$$

$$= \sum_{\ell=1}^{L} \sum_i \frac{\partial \Phi_r}{\partial m_{\ell;i}}(\boldsymbol{m}_1^{(n+1)}, \dots, \boldsymbol{m}_L^{(n+1)}) \frac{\partial m_{\ell;i}^{(n+1)}}{\boldsymbol{\theta}^{(n-k)}}$$

$$= \sum_{\ell=1}^{L} \sum_i b_\ell^{(n+1)} \beta_\ell^k \frac{\partial \Phi_r}{\partial m_{\ell;i}}(\boldsymbol{m}_1^{(n+1)}, \dots, \boldsymbol{m}_L^{(n+1)})$$

$$\times \nabla g_{\ell;i}^{(n-k)}(\boldsymbol{\theta}^{(n-k)}),$$

and using the uniform boundedness of derivatives of $g_{\ell;i}^{(n-k)}$ and Φ_r. Eq. (18) is proven similarly. $\square$

Lemma D.2. *Let* $\{a_k\}_{k=1}^\infty$ *and* $\{b_k\}_{k=1}^\infty$ *be sequences of reals such that* $\sum_{k=1}^\infty (|a_k| + |b_k|) < \infty$. *Then*

$$\sum_{k=1}^n a_k \sum_{s=n-k}^{n-1} b_s \xrightarrow[n\to\infty]{} 0.$$

Proof. Fix $\varepsilon > 0$. Take such positive integer n_0 that for any $n_0 \leqslant n_1 \leqslant n_2$ we have $\sum_{s=n_1}^{n_2} (|a_k| + |b_k|) < \varepsilon$. Then for any $n \geq 2n_0 - 1$ the following holds:

$$\sum_{k=1}^n |a_k| \sum_{s=n-k}^{n-1} |b_s| = \sum_{k=1}^{n-n_0} |a_k| \underbrace{\sum_{s=n-k}^{n-1} |b_s|}_{<\varepsilon} + \underbrace{\sum_{k=n-n_0+1}^{n} |a_k|}_{<\varepsilon} \sum_{s=n-k}^{n-1} |b_s| < \varepsilon \sum_{k=1}^\infty (|a_k| + |b_k|).$$

Since ε is arbitrary and $\sum_{k=1}^\infty (|a_k| + |b_k|)$ is a finite constant, the statement follows. $\square$