Diss. ETH No. 29649

# Dynamics of Adaptive Momentum Optimizers on Challenging Deep Learning Landscapes

A thesis submitted to attain the degree of

Doctor of sciences of ETH Zurich
(Dr. sc. ETH Zurich)

presented by

Antonio Orvieto

MSc ETH in Robotics, Systems and Control
born on 09.10.1993
citizen of Italy

accepted on the recommendation of

| | |
|---|---|
| Prof. Dr. Thomas Hofmann (ETH Zurich), | examiner |
| Prof. Dr. Francis Bach (INRIA, ENS), | co-examiner |
| Prof. Dr. Aurelien Lucchi (University of Basel) | co-examiner |

2023

*To Boris Polyak and Will J. Merry,*
*your legacy will never be forgotten.*

## ABSTRACT

Deep learning technologies are skyrocketing in popularity across a wide range of domains, with groundbreaking accomplishments in fields such as natural language processing, biology, healthcare and computer vision. The remarkable success of neural networks makes machine-learning researchers like myself highly excited by the potential impact of our work on the future of science and technology. As a community, our goal should be to provide scientists and engineers with a solid deep-learning toolbox where convenient architectural choices and best training practices are clearly outlined.

However, today, we are still far from an effective practice-oriented theory of deep learning: state-of-the-art neural networks have complex modular designs, with components whose individual or combined effects on performance are often not well understood. Moreover, when training a network, *best practices are often not well aligned with optimization theory*.

In an attempt to bridge this gap, this thesis delves into the enigmatic dynamics and effectiveness of adaptive momentum methods — state-of-the-art solutions for training complex architectures such as Transformers. Throughout this volume, we unravel their mechanisms through a rigorous analysis of acceleration and adaptive stepsizes on toy problems as well as on modern architectures, including Multi-Layer Perceptrons (MLPs) and Transformers, revealing the profound challenges faced when optimizing these models. Leveraging the lessons learned from our study of Adam, we provide new variants with provable curvature adaptation properties and promising performance. In addition, our deepened understanding of the inner workings of adaptive methods gives us insights for solving an old problem: efficient training of recurrent models, taming the issue of vanishing/exploding gradients and achieving state-of-the-art performance on long-range reasoning tasks with an interpretable architecture. Finally, as we probe the terrain of generalization through better optimization, we propose novel noise injection schemes that can be incorporated into adaptive methods and offer compelling strategies to navigate intricate loss landscapes towards flat minima.

## ESTRATTO

Le tecnologie di apprendimento profondo stanno raggiungendo una popolarità vertiginosa in un'ampia gamma di settori, con risultati rivoluzionari nell'elaborazione del linguaggio naturale, la biologia, l'assistenza sanitaria e la computer vision. Il notevole successo delle reti neurali rende i ricercatori di apprendimento automatico come me molto entusiasti del potenziale impatto del nostro lavoro sul futuro della scienza e della tecnologia. Come comunità, il nostro obiettivo dovrebbe essere quello di fornire a scienziati e ingegneri una solida cassetta degli attrezzi per il deep-learning, in cui le scelte architettoniche più convenienti e le migliori pratiche di addestramento siano chiaramente delineate.

Tuttavia, oggi siamo ancora lontani da un'efficace teoria del *deep learning*: le reti neurali di ultima generazione hanno design modulari complessi, con componenti i cui effetti individuali o combinati sulle prestazioni spesso non sono ben compresi. Inoltre, durante l'addestramento, molte pratiche non si trovano ben allineate con la teoria dell'ottimizzazione.

Nel tentativo di colmare questa lacuna, questa tesi approfondisce le dinamiche enigmatiche e l'efficacia dei metodi di ottimizzazione adattivi —- soluzioni all'avanguardia per l'addestramento di architetture complesse come i trasformatori. Nel corso del volume, sveliamo i loro meccanismi attraverso un'analisi rigorosa dell'accelerazione e dei passi adattive su problemi semplificati e su architetture moderne, tra cui i perceptron multistrato (MLP) e i trasformatori, rivelando le profonde sfide da affrontare nell'ottimizzazione di questi modelli. Sfruttando le lezioni apprese dallo studio di Adam, forniamo nuove varianti con proprietà di adattamento alla curvatura dimostrabili e prestazioni promettenti. Inoltre, la nostra comprensione approfondita del funzionamento interno dei metodi adattativi ci offre spunti per risolvere un vecchio problema: l'addestramento efficiente dei modelli ricorrenti. Infine, mentre sondiamo il terreno della generalizzazione attraverso una migliore ottimizzazione, proponiamo nuovi schemi di iniezione del rumore che possono essere incorporati nei metodi adattivi e che offrono strategie convincenti per navigare in intricati paesaggi verso minimi piatti.

# ACKNOWLEDGEMENTS

A big thank you goes to **Thomas** Hofmann and **Aurelien** Lucchi — spiritual and scientific guides during my PhD at ETH. I learned a lot from you both: from day-to-day research to convincing and successful writing. I hope that with this thesis I can showcase your teachings.

During my PhD I had the privilege of collaborating outside ETH with awesome people, I will name a few: **Francis** Bach, **Hans** Kersting, **Lin** Xiao, **Razvan** Pascanu, **Soham** De, **Sam** Smith, **Frank** Proske, and **Nicolas** Loizou. Each of you has a unique research style, it was incredibly thought-provoking to work with you —— in different parts of this world: *Paris, Seattle, London, Montreal!*

The journey would have never been the same without the support of my beloved lab/Deep Learning friends. Thanks **Paulina M.** for showing me what true organization looks like; thanks **Luca** for showing me the depths of Alpnach and to **Tommaso** for embodying the concept of *fac=turation*; thanks **Enea** for your never-ending excitement; thanks **Dario** for your wisdom; thanks **Anna** for the Covid serenade; thanks **Giambattista** for the lucid dreams; thanks **Jonas** for teaching me how to surf on the complexities of deep networks; thanks **Yannic** for my 5 minutes of glory on Youtube; thanks **Lorenzo** for the international agreements in support of our country; thanks **Sotiris** for you inspiring concentration; thanks **Gregor** for the beverages; thanks **Hadi** for your smiles; thanks **Giulia** for the reading tips; thanks **Leo** and **Janis** for being early in the morning — you really did push me; thanks **Gary** for staying so late at night; thanks to **Octavian**, **Paulina G.** and **Florian** for getting me interested in language.

Outside of my research circle, I wish to first express my deepest gratitude to **Giulia**: ideas that emerged during my journeys on airplanes, eagerly anticipating our reunions, constitute some of the fundamental ideas presented in this work. You also gave me strength in the difficult process of summarizing four years of research into this volume — if it is any good, it's only because of you and your surprise party.

My **family** also played a crucial role, showing me love every time I visited, down in Venice — except for the heat and the mosquitoes —- it is always exciting to come back. Besides, in Venice I never miss a visit to a subset of the **Yabai** (Bet, China, Gian, Daido, Tommy, Sacca, Manu) — you guys never changed, I am so happy we are still close after all these years. I am pretty sure for some cultures the number four is considered perfect: **Gilberto, Ola, Alex** and myself are perfect as a group, especially in combination with horror movies. Please Alex, take the Ph.D. position offer in Texas, would not stand you starting to eat beans for breakfast. Also, **Serena**: thanks for the agenda back in 2020, I must admit that despite my efforts I am still not able to properly organize my day.

Last, I like to thank all the survivors of *pizza brutta* in Zurich: you people are strong, and too many to mention. You know who you are.

# CONTENTS

# 1 PREFACE AND OUTLOOK

*"An ideal thesis is like a circle. You are allowed to choose one point: that is page 1."*

– Will J. Merry

Back in 2018, after writing my Master's thesis, I spent a month at the MPI for Intelligent Systems in Tübingen. In between working on my projects and moving around Airbnbs, I had the opportunity of meeting Prof. Christian Lubich at the University of Tübingen — one of the most prolific and influential researchers in numerical analysis today. While our backgrounds and research interests differed, I was eager to chat with him about the connections between geometric numerical integration and accelerated optimizers for convex problems (the subject of my Master's thesis). Pretty soon in our discussion, our topic shifted to optimization for deep learning, and in particular towards a specific adaptive stepsize scheme for stochastic gradient descent — Adam (Kingma and Ba, 2014), the method of choice today for training large neural networks such as Transformers (Vaswani et al., 2017).

Let $f : \mathbb{R}^d \to \mathbb{R}$ be the loss associated with training a neural network (averaged over data points), and consider updating model parameters $x \in \mathbb{R}^d$ iteratively using gradient estimates $g(x)$ computed using a minibatch of samples. The Adam update (without bias correction) is

$$\begin{cases} x_{k+1} &= x_k - \dfrac{\eta}{\sqrt{v_k + \epsilon}} \cdot m_k \\ m_{k+1} &= \beta_1 m_k + (1-\beta_1) g(x_k) \\ v_{k+1} &= \beta_2 v_k + (1-\beta_2) g(x_k)^2 \end{cases}, \tag{Adam}$$

where squares, square roots, multiplications, and divisions by vectors are performed elementwise, and $\eta, \beta_1, \beta_2, \epsilon \in \mathbb{R}_{\geq 0}$ are hyperparameters.

Prof. Lubich did not have prior knowledge of this method — after I wrote it down, *he stared at it in silence for a couple of minutes*, trying to grasp the

essence behind this update rule and potential reasons for its success. His curiosity, precision, and humbleness served as great inspirations for me during my Ph.D. and motivated my investigations on this topic.

At the time of my meeting with Prof. Lubich, Adam had over $14k$ citations; today, it has almost $150k$ citations. While citation counts can be misleading, the impact of Adam is indisputable: it made it possible, for instance, to train large language models (OpenAI, 2023; Touvron et al., 2023; Köpf et al., 2023), and offers practitioners a commendable blend of speed and generalization performance while alleviating the necessity for an extensive hyperparameters search. This in contrast to more theoretically principled approaches like vanilla stochastic gradient descent (SGD) with momentum, which often demands heavy hyperparameter tuning in modern applications. In addition, Adam stands out as one of the few optimization algorithms capable of navigating the intricate landscapes[1] inherent to attention-based Transformers (Vaswani et al., 2017).

While the Adam scheme builds upon a great deal of work on adaptive methods (Ward et al., 2019; Tieleman and Hinton, 2012), its peculiar dynamics and extraordinary effectiveness continue to baffle researchers. Its structure is often connected to approximations of the empirical Fischer information (Kunstner et al., 2019; Martens, 2020), which is related in some settings to the Fisher information preconditioner and therefore to the geometrically principled natural gradient descent update (Amari, 1998) and its practical variant KFAC (Martens and Grosse, 2015). In turn, in some special settings, Fisher information preconditioning coincides with the Generalized Gauss-Newton (GGN) method (Martens, 2020), which uses the neural network model Jacobian to adapt the SGD direction with second-order information. Unfortunately, the approximate connection between Adam and second-order schemes cannot be leveraged quantitatively to derive rates: most results for Adam in the literature (Reddi et al., 2018; Défossez et al., 2022; Shi and Li, 2021) provide guarantees that show no sizable advantage over SGD.

1 Despite rigorous tuning efforts and incorporation of techniques like layer normalization (Ba et al., 2016) and residual connections (He et al., 2016), SGD's performance in training Transformers is notably suboptimal, as indicated by findings in Xiong et al., 2020; Noci et al., 2022. In contrast, well-initialized and normalized convolutional networks demonstrate excellent training results with both SGD and adaptive optimization methods. However, the rise of vision Transformers (Dosovitskiy et al., 2020) in recent years has led to a pronounced increase in the adoption of Adam as the preferred optimization choice also by the vision community.

Motivated by the exceptional impact of Adam in the development of deep learning and by its intriguing yet poorly understood mechanism — which combines an adaptive coordinate-wise learning rate with momentum — we (I will start now using the *pluralis maiestatis*) provide in this thesis a thorough investigation of the dynamics of adaptive momentum methods in the context of deep learning. Our discussion will be mostly linear: starting from recent advancements in the theory of momentum and acceleration — a crucial component of Adam — we will then dive into a study of the dynamics of Adam in deep networks. After a theoretical analysis resulting in new adaptive methods with strong theoretical guarantees, we will discuss settings where training with stochastic gradients and adaptive stepsizes is not enough: in particular, we will open Pandora's box of generalization and discuss how to boost test performance in settings where simply minimizing the training loss does not yield the desired accuracy. This thesis also covers an extremely challenging setting: training deep recurrent models for long sequential data. This scenario is prototypical for an important lesson: on top of adaptive methods, efficient parametrization and normalization of deep networks are crucial for performance.

While the *Leitmotiv* of this thesis is understanding and improving the performance of adaptive optimizers in deep learning, the discussion will often take detours into specific (yet needed) architecture investigations and into detailed discussions of fundamental tools such as noise injection and momentum, which are not specific to adaptive methods *per se*. Due to the large number of papers discussed here, we will only focus on the main ideas and, where possible, illustrate results using toy examples.

A sketch illustrating the structure of this work can be found in Figure 1. What follows is a brief outline which connects together the twelve papers discussed in this volume. For a complete list of papers produced during my PhD, including those we left out of the discussion here, the reader can check the last pages of this chapter.

We want to clarify that the purpose of this thesis is *not to advocate for the widespread adoption of adaptive methods* in the context of deep learning. Instead, our primary objective is to delve into the reasons that contribute to their effectiveness and to bring to light the complexities involved in optimizing modern neural loss landscapes. As such, we would like the reader

to consider adaptive methods as a lens for studying of the structure of deep learning problems to reveal issues, and suggest improvements or theoretically grounded variants.

1. In Chapter 2, we study momentum — a fundamental pillar of modern optimization theory and one of the two main components of Adam (the second is the stepsize adaptation using $\sqrt{v_k}$). Setting $\beta_2 = 0$ and renaming $\beta := \beta_1$ in Adam, we indeed recover the classical Heavy-ball algorithm by Polyak, 1964:

$$\begin{cases} x_{k+1} &= x_k - \eta \cdot m_k \\ m_{k+1} &= \beta m_k + (1-\beta) g(x_k) \end{cases}. \tag{HB}$$

In the convex deterministic setting, HB (more precisely, Nesterov's variant) leads to accelerated convergence compared to gradient descent. While this result enjoys numerous proofs, derived over the years after the celebrated work of Nesterov (Nesterov, 1983), the nature of acceleration — from which we can learn a lot in deep learning — is still subject of debate among researchers. A modern line of research, started with the seminal contribution of Su et al., 2016, proposes to look at HB using ordinary differential equations (ODEs). This perspective is extremely thought-provoking since it allows to simplify and compress the accelerated convergence proof of Nesterov's method to just a few lines. Wibisono et al., 2016 took this approach one step further and showed that Nesterov's ODE solution could be thought of as a stationary point for the action of a specific Lagrangian. As such, Wibisono et al., 2016 claim Nesterov's method is optimal under some metric in the space of curves. This thesis starts with a negative result in Section 2.2 (based on our work in Zhang et al., 2021b, NeurIPS 2021), where we show that, contrary to what is claimed in Wibisono et al., 2016, Nesterov's path is a often saddle point and not a minimizer in the space of curves. With this finding, our aim is not to disincentivize the use of continuous-time models to understand momentum but rather to set the record straight about tools we are/are not allowed to use when studying acceleration. Indeed, Section 2.3 (based on our work in Orvieto et al., 2019, UAI 2019), shows that using stochastic differential equations as models for

momentum reveals intriguing averaging properties and connects acceleration to stable gradient amplification. Motivated by this success, in Section 2.4 (based on our work in Orvieto and Lucchi, 2019b, NeurIPS 2019) we study the foundations of continuous-time approximations of simple optimizers and find conditions under which gradient-based methods are *shadowed* by their continuous-time limit.

2. In Chapter 3, we dive into the second component of Adam: adaptive stepsizes. Setting $\beta_1 = 0$ (no momentum) in Adam one retrieves the RMSprop (Tieleman and Hinton, 2012) update:

$$\begin{cases} x_{k+1} &= x_k - \dfrac{\eta}{\sqrt{v_k + \epsilon}} \cdot g_k \\ v_{k+1} &= \beta_2 v_k + (1-\beta_2) g(x_k)^2 \end{cases} . \qquad \text{(RMSprop)}$$

While momentum helps, this simplified version of Adam is already very effective compared to SGD or HB in the context of deep learning, especially on Transformers (Vaswani et al., 2017).
To study the dynamics of RMSprop and understand its effectiveness in complex landscapes, it is necessary first to analyze the architecture at hand under commonly used initialization schemes (He et al., 2015).
In Section 3.2 (based on our work in Orvieto et al., 2022b, AISTATS 2022), we discuss the distribution of gradients at initialization in multi-layer perceptrons (MLPs) and convolutional networks (CNNs). We provide a detailed description of the landscape for Deep Neural Chains (i.e. deep MLPs with unit width) and show the power of RMSprop in optimizing its weights. In particular, we highlight that, without further tricks such as batch normalization (Ioffe and Szegedy, 2015) or residual connections (He et al., 2016) optimizing MLPs is a difficult task that SGD (with momentum) often cannot solve — but Adam can. Next, in Section 3.3 (based on our work in Noci et al., 2022, NeurIPS 2022), we perform a similar analysis on Transformers (Dong et al., 2021) — state-of-the-art solutions for both text analysis and synthesis (OpenAI, 2023; Touvron et al., 2023) as well as image generation (Dosovitskiy et al., 2020). For the first time in the literature, we precisely characterize signal prop-

agation and representation rank collapse in this architecture. We link the gained insights into the landscape structure of the attention mechanism and to the effectiveness of RMSprop and Adam compared to SGD in this class of models.

3. While Adam is the workhorse of modern training pipelines, its effectiveness is enigmatic from a standard optimization theory viewpoint. Indeed, it is well-known that vanilla Adam and RMSprop do not converge[2] under standard stepsize annealing in convex problems (Reddi et al., 2018). While this issue can be fixed with some algorithmic modifications (increasing $\beta$ parameters over time, modifying the rule for $v_k$, etc.), such adjustments are known to lead to suboptimal performance and to reduce adaptivity. Inspired by this problem and by recent advancements in adapting the first ever introduced adaptive stepsize (Polyak, 1987) to the machine learning setting (Loizou et al., 2021), in Chapter 4 we introduce new optimizers with provable adaptivity features and strong theoretical guarantees. In Section 4.1&4.2 (based on our work in Orvieto et al., 2022c, NeurIPS 2022) we describe the main issues with the stochastic Polyak stepsize and propose a variation which leads to convergence without knowledge of the gradient Lipschitz constant (required by SGD in unbounded domains). In Section 4.3 (based on our work in Orvieto and Xiao, 2023, ICML2023 HiLD Workshop), we instead approach algorithm design from first principles and derive a new method, NGN, which resembles the Polyak stepsize but enjoys stronger convergence guarantees and performance — also on deep learning tasks. This algorithm is currently the subject of our investigation and of a future journal publication.

4. The landscape of some neural networks can be challenging even for state-of-the-art optimizers like Adam. This is the case, for instance, of recurrent models such as RNNs (Hopfield, 1982), LSTMs (Hochreiter and Schmidhuber, 1997b), and GRUs (Cho et al., 2014) that suffer from the vanishing gradient problem due to backpropagation through time (Pascanu et al., 2013). The issue of training RNNs efficiently on long sequential data resulted in an architecture shift:

2 This is arguably the main reason why many optimization researchers chose not to work on Adam, or consider it a "bad" algorithm.

namely, in the rise of attention-based models (Vaswani et al., 2017) for sequence understanding. Unfortunately, while Transformers are scalable in depth and width, the inductive bias of this architecture and its squared complexity in the sequence length makes it ineffective for reasoning tasks comprising sequences of thousands of tokens (Tay et al., 2020). In Section 5.3 (based on our work in Orvieto et al., 2023c, ICML 2023 Oral), inspired by recent advancements in state-space models (Gu et al., 2021), we revisit the RNN architecture for long-range reasoning tasks and show how a smart reparametrization and normalization of the recurrence, designed to work well with Adam, can tame the vanishing gradient issue and unlock fast training, leading to state-of-the-art results. Surprisingly, a fundamental step in gaining scalability of our model was to remove recurrent non-linearities. In Section 5.5 (based on our work in Orvieto et al., 2023a, ICML 2023 HiLD Workshop), we elaborate on this surprising finding and show that the modified linear RNN model introduced in (Orvieto et al., 2023c) — the LRU — when paired with token-wise MLPs can approximate any nonlinear dynamical system.

5. While in the last four chapters we focused on algorithmic and architectural insights regarding optimization of complex *training loss* landscapes with adaptive momentum methods, in Chapter 6, we conclude the thesis by discussing the critical issue of generalization. Indeed, in machine learning tasks, while finding local minimizes often leads to vanishing training loss (Kawaguchi, 2016), one is mainly interested in regions of the loss landscape that also yield good generalization performance. While characterizing mathematically such regions is not easy; evidence shows that *flat* minimizes often yield improved test performance (Jiang et al., 2019; Keskar et al., 2016). Standard approaches known to boost generalization through inducing flatness are, for instance, sharpness-aware minimization (Foret et al., 2020) (requires two gradient evaluations), adversarial perturbations (Wu et al., 2020) (requires data Jacobian) and label noise injection (Song et al., 2022). A general belief overarching the literature on the topic of flat minimizes is that the (data dependent) noise in SGD can shape its trajectory towards flat minima (Simsekli et al., 2019) — a mechanism that can be amplified by

inserting additional noise in the labels (HaoChen et al., 2021). The question we ask in this chapter is "*Can we find a data-independent noise injection scheme which drives the dynamics to flat minima?*". In Section 6.2 (based on our work in Lucchi et al., 2022, NeurIPS 2022), we show that, compared to vanilla Gaussian noise injection on the gradients, fractional noise injection (Mandelbrot and Van Ness, 1968) leads to improved landscape exploration features. In Section 6.3 (based on our work in Orvieto et al., 2022a, ICML 2022), we show how to leverage insights from fractional noise to design new noise injection schemes capable of provably driving convergence to flat minima. Last, in Section 6.4 (based on our work in Orvieto et al., 2023b, AISTATS 2023), we discuss and solve the challenges for successful noise injection in the context of deep learning.

To ease the reading process, at the beginning of each section we summarize its content and main findings in a question/answer format:

**Question :** Which optimization tricks make RNNs trainable?

**Answer (Orvieto et al., 2023c)**: Check Section 5.3!

We will use the **bold font** to highlight important keywords, and *italic* for questions, new terminology, and remarks.

LIST OF PUBLICATIONS. In this paragraph, we list the 12 publications (10 conference + 2 workshops) discussed in this thesis (in the order of discussion, with alphabetic list matching Figure 1). We then list 19 additional works (some of which are first-author publications at top ML venues) that are not discussed in this volume in the interest of space:

A. *Rethinking the Variational Interpretation of Nesterov's Method*
NeurIPS 2021 (Zhang et al., 2021b)
P. Zhang*, **A. Orvieto***, Hadi Daneshmand

B. *The Role of Memory in Stochastic Optimization*
UAI 2019 (Orvieto et al., 2019)
**A. Orvieto**, J. Kohler, A. Lucchi

C. *Shadowing Properties of Optimization Algorithms*
NeurIPS 2019 (Orvieto and Lucchi, 2019b)
**A. Orvieto**, A. Lucchi

D. *Vanishing Curvature in Randomly Initialized Deep ReLU Networks*
AISTATS 2022 (Orvieto et al., 2022b)
**A. Orvieto***, J. Kohler*, D. Pavllo, T. Hofmann, A. Lucchi

E. *Signal Propagation in Transformers: Theoretical Perspectives and the Role of Rank Collapse*
NeurIPS 2022 (Noci et al., 2022)
L. Noci*, S. Anagnostidis*, L. Biggio*, **A. Orvieto***, S. Pal Singh*, A. Lucchi

F. *Dynamics of SGD with Stochastic Polyak Stepsizes: Truly Adaptive Variants and Convergence to Exact Solution*
NeurIPS 2022 (Orvieto et al., 2022c)
**A. Orvieto**, S. Lacoste-Julien, N. Loizou

G. *A New Adaptive Method for Minimizing Non-negative Losses*
ICML 2023 HiLD Workshop (full paper in preparation)
(Orvieto and Xiao, 2023)
**A. Orvieto**, L.Xiao

H. *Resurrecting Recurrent Neural Networks for Long Sequences*
ICML 2023 Oral, (Orvieto et al., 2023c)
**A. Orvieto**, S. L Smith, A. Gu, A. Fernando, C. Gulcehre, R. Pascanu, S. De

I. *Universality of Linear RNNs Followed by Nonlinear Projections*
ICML 2023 HiLD Workshop (full paper in preparation)
(Orvieto et al., 2023a)
**A. Orvieto**, S. De, C. Gulcehre, R. Pascanu, S. L Smith

J. *On the Theoretical Properties of Noise Correlation in SGD*
NeurIPS 2022 (Lucchi et al., 2022)
H. Kersting, **A. Orvieto**, F. Bach, F. Proske, A. Lucchi

K. *Anticorrelated Noise Injection for Improved Generalization*
ICML 2022 (Orvieto et al., 2022a)
**A. Orvieto***, H. Kersting*, F. Proske, F. Bach, A. Lucchi

L. *Explicit Regularization in Overparametrized Models via Noise Injection*
AISTATS 2023, (Orvieto et al., 2023b)
**A. Orvieto***, A. Raj*, H. Kersting*, F. Bach

The 19 publications not discussed in this volume are listed next, in order of appearence.

1. *Continuous-time Models for Stochastic Optimization Algorithms*
NeurIPS 2019 (Orvieto and Lucchi, 2019a)
**A. Orvieto**, A. Lucchi

2. *Continuous-time Acceleration in Riemannian Optimization*
AISTATS 2020 (Alimisis et al., 2020)
F. Alimisis, **A. Orvieto**, G. Becigneul, A. Lucchi

3. *An Accelerated DFO Algorithm for Finite-sum Convex Functions*
ICML 2020 (Chen et al., 2020b)
C. Yuwen, **A. Orvieto**, A. Lucchi

4. *Two-Level K-FAC Preconditioning for deep learning*
NeurIPS OPT Workshop 2020 (Tselepidis et al., 2020)
N. Tselepidis, J. Kohler, **A. Orvieto**

5. *Learning explanations that are hard-to-vary*
ICLR 2021 (Parascandolo et al., 2021)
G. Parascandolo*, A. Neitz*, **A. Orvieto**, L. Gresele, B. Schölkopf

6. *Revisiting the Role of Symplectic Numerical Integration on Acceleration and Stability in Convex Optimization*
AISTATS 2021 (Zhang et al., 2021c)
P. Zhang, **A. Orvieto**, H. Daneshmand, R. Smith, T. Hofmann

7. *Momentum Improves Optimization on Riemannian Manifolds*
AISTATS 2021 (Alimisis et al., 2021)
F. Alimisis, **A. Orvieto**, G. Becigneul, A. Lucchi

8. *On the Second-order Convergence of Random Search Methods*
NeurIPS 2021 (Lucchi et al., 2021)
A. Lucchi*, **A. Orvieto***, Adamos Solomou*

9. *Analysis of Pharmacological Modulation of Senescence in Human Epithelial Stem Cells*
   Cellular and Molecular Medicine 2022 (Barbaro et al., 2022)
   V. Barbaro, **A. Orvieto**, and many others.

10. *Faster Single-loop Algorithms for Minimax without Strong Concavity*
    AISTATS 2022 (Yang et al., 2022)
    J. Yang, **A. Orvieto**, A. Lucchi, N. He

11. *Enhancing Unit-Tests for Invariance Discovery*
    ICML SCIS Workshop 2022 (De Bartolomeis et al., 2022)
    P. De Bartolomeis, **A. Orvieto**, G. Parascandolo

12. *Should you follow the gradient flow? Insights from Runge-Kutta Descent*
    ICML CTPML Workshop 2022 (Li and Orvieto, 2022)
    X. Li, **A. Orvieto**

13. *Batch-size Selection by Stochastic Optimal Control*
    NeurIPS HITY Workshop 2022 (Zhao et al., 2022)
    J. Zhao, A. Lucchi, F. N. Proske, **A. Orvieto**, H. Kersting

14. *First Exit Times of Ornstein-Uhlenbeck Processes in High Dimensions*
    Journal of Physics A 2023 (Kersting et al., 2023)
    H. Kersting, **A. Orvieto**, F. Proske, A. Lucchi

15. *Achieving a Better Stability-Plasticity Trade-off via Auxiliary Networks in Continual Learning*
    CVPR 2023 (Kim et al., 2023)
    S. Kim, L. Noci, **A. Orvieto**, T. Hofmann

16. *On the Effectiveness of Randomized Signatures as Reservoir for Learning Rough Dynamics*
    IJCNN 2023 (Compagnoni et al., 2023b)
    E. Monzio Compagnoni, A. Scampicchio, L. Biggio, **A. Orvieto**, T. Hofmann, J. Teichmann

17. *An SDE for Modeling SAM: Theory and Insights*
    ICML 2023 (Compagnoni et al., 2023a)
    E. Monzio Compagnoni, L. Biggio, **A. Orvieto**, H. Kersting, F. N. Proske, A. Lucchi

18. *On the Advantage of Lion Compared to signSGD with Momentum*
ICML HiLD Workshop 2023 (Noiato et al., 2023)
A. Noiato, L. Biggio, **A. Orvieto**

19. *An Accelerated Lyapunov Function for Heavy-Ball on Convex Quadratics*
Under Submission to Optimization Letters 2023 (Orvieto, 2023)
**A. Orvieto**

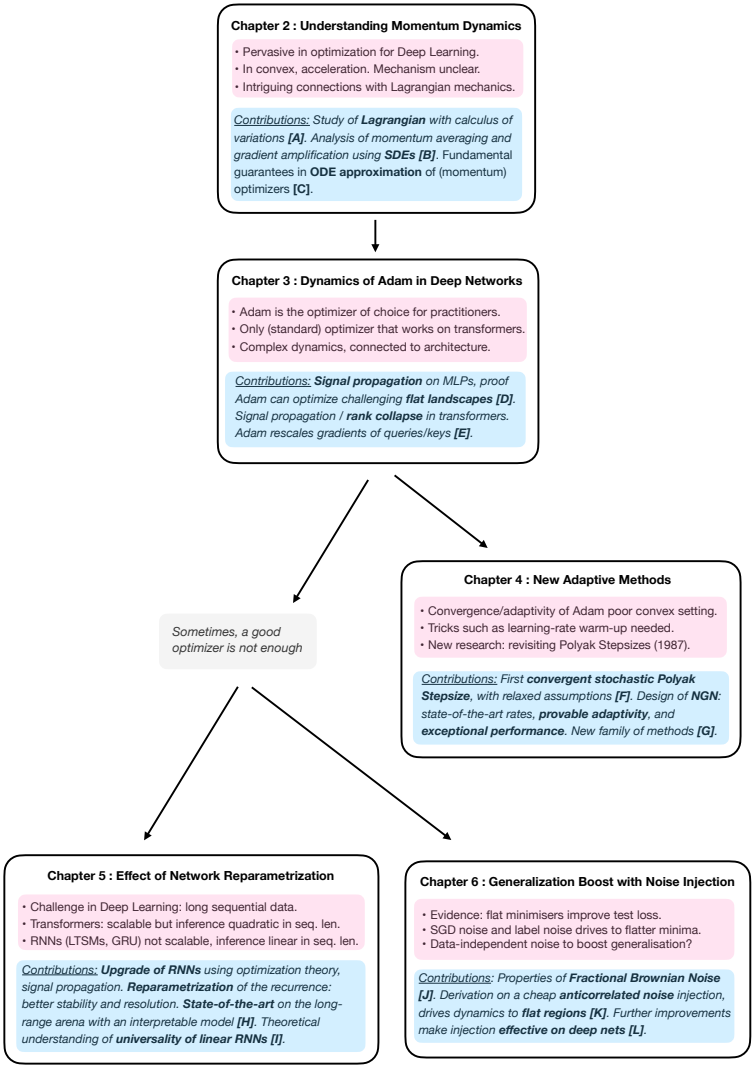


**Figure 1:** Thesis map. Citations follow the list provided in the introduction.

# 2 UNDERSTANDING MOMENTUM WITH SDES

*The simplicities of natural laws arise through the complexities of the language we use for their expression.*

– Eugene Wigner.

As outlined in the introduction, in this chapter we study the **first fundamental component of adaptive momentum methods** — momentum. We provide a discussion of this mechanism in simple settings and bring to light interesting properties that can be used for the design and analysis of (deep learning) optimizers. After our preliminaries in Section 2.1, in Section 2.2, we discuss the variational interpretation of momentum — i.e. we revisit its **optimality on the space of curves**. Next, in Section 2.3 we show how the behavior of accelerated momentum methods can be linked to **gradient amplification and polynomial averaging**. Finally, in Section 2.4, we discuss the fundamental properties of **continuous-time models** of optimizers.

For a summary of the mathematical tools required in this chapter — i.e. **ordinary and stochastic differential equations**, the reader can check Appendix A. Basic definitions and inequalities used in smooth convex optimization can be also found in Appendix A.

## 2.1 CONTINUOUS-TIME MODELS OF ACCELERATION

Consider the problem of unconstrained convex optimization, i.e. to find

$$x^* \in \arg\min_{x \in \mathbb{R}^d} f(x), \tag{P}$$

for some lower bounded convex $L$-smooth[1] loss $f \in \mathcal{C}^1(\mathbb{R}^d, \mathbb{R})$[2].

1 A differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ is said to be $L$-smooth if it has $L$-Lipschitz gradients.
2 The family of $\mathcal{C}^1(\mathbb{R}^d, \mathbb{R})$ contains all continuously differentiable functions from $\mathbb{R}^d$ to $\mathbb{R}$.

NESTEROV'S ACCELERATION. Nemirovskii and Yudin, 1983 showed that no gradient-based optimizer can converge to a solution of (P) faster than $\mathcal{O}(k^{-2})$, where $k$ is the number of gradient evaluations[3]. While Gradient Descent (GD) converges like $\mathcal{O}(k^{-1})$, the optimal rate $\mathcal{O}(k^{-2})$ is achieved by the celebrated Nesterov's Accelerated Gradient Descent (NAG) method, proposed by Nesterov, 1983:

$$x_{k+1} = y_k - \eta \nabla f(y_k), \quad \text{with} \quad y_k = x_k + \frac{k-1}{k+2}(x_k - x_{k-1}). \tag{NAG}$$

Despite its proven effectiveness and optimal convergence rate, the precise **intuition behind Nesterov's method**, and the core mechanism triggering its acceleration, continue to be **enigmatic and actively researched**, as highlighted by studies from Allen-Zhu and Orecchia, 2017; Defazio, 2019; Ahn and Sra, 2022. The puzzling aspects include its counterintuitive "leap-ahead" **extrapolation step** and the utilization of past iteration data, both of which remarkably enhance performance without a clear, widely accepted theoretical explanation.

DIFFERENTIAL EQUATIONS MODELS. Towards understanding the acceleration mechanism, Su et al., 2016 made an interesting observation: the convergence rate gap between GD and NAG is **retained in the continuous-time limits** (as the step-size $\eta$ vanishes):

$$\dot{X} + \nabla f(X) = 0 \quad \text{(GD-ODE)}, \qquad \ddot{X} + \frac{3}{t}\dot{X} + \nabla f(X) = 0 \quad \text{(NAG-ODE)},$$

where $\dot{X} := dX/dt$ denotes the time derivative (velocity) and $\ddot{X} := d^2X/dt^2$ the acceleration. Namely, we have that GD-ODE converges like $\mathcal{O}(t^{-1})$ and NAG-ODE like $\mathcal{O}(t^{-2})$, where $t > 0$ is the time variable. This seminal paper gave researchers a new tool to understand the nature of accelerated optimizers through Bessel Functions (Su et al., 2016), and led to the design of many novel fast and interpretable algorithms outside the Euclidean setting (Wibisono et al., 2016; Wilson et al., 2019), in the stochastic setting (Krichene et al., 2015; Xu et al., 2018) and also in the manifold setting (Alimisis et al., 2020; Duruisseaux and Leok, 2021).

3 This lower bound holds for $k < d$ hence it is only interesting in the high-dimensional setting.

NESTEROV AS SOLUTION TO EULER-LAGRANGE EQUATIONS. It is easy to see that NAG-ODE can be recovered from **Euler-Lagrange equations**, starting from the time-dependent *Lagrangian*

$$L(X, \dot{X}, t) = t^3 \left( \frac{1}{2} \|\dot{X}\|^2 - f(X) \right). \tag{1}$$

Indeed, the Euler-Lagrange equation

$$\frac{d}{dt} \left( \frac{\partial}{\partial \dot{X}} L(X, \dot{X}, t) \right) = \frac{\partial}{\partial X} L(X, \dot{X}, t) \tag{2}$$

reduces in this case to $t^3 \ddot{X} + 3t^2 \dot{X} + t^3 \nabla f(X) = 0$, which is equivalent to NAG-ODE (assuming $t > 0$). In an influential paper, Wibisono et al., 2016 generalized the derivation above to non-Euclidean spaces, where the degree of separation between points $x$ and $y$ is measured by means of the *Bregman Divergence* (Bregman, 1967) $D_\psi(x, y) = \psi(y) - \psi(x) - \langle \nabla \psi(x), y - x \rangle$, where $\psi : \mathbb{R}^d \to \mathbb{R}$ is a strictly convex and continuously differentiable function (see e.g. Chapter 1.3.2 in Amari, 2016). Namely, they introduced the so-called *Bregman Lagrangian*:

$$L_{\alpha,\beta,\gamma}(X, \dot{X}, t) = e^{\alpha(t)+\gamma(t)} \left( D_\psi(X + e^{-\alpha(t)} V, X) - e^{\beta(t)} f(X) \right), \tag{3}$$

where $\alpha, \beta, \gamma$ are continuously differentiable functions of time. The Euler-Lagrange equations imply

$$\ddot{X} + (e^{\alpha(t)} - \dot{\alpha}(t))\dot{X} + e^{2\alpha(t)+\beta(t)} \left[ \nabla^2 \psi(X + e^{-\alpha(t)} \dot{X}) \right]^{-1} \nabla f(X) = 0. \tag{4}$$

The main result of Wibisono et al., 2016 is that, under the *ideal-scaling conditions* $\dot{\beta}(t) \leq e^{\alpha(t)}$ and $\dot{\gamma}(t) = e^{\alpha(t)}$, any solution to Eq. (4) converges to a solution of (P) at the rate $\mathcal{O}(e^{-\beta(t)})$. Under the choice $\psi(x) = \frac{1}{2}\|x\|_2^2$, we get back to the Euclidean metric $D_\psi(x, y) = \frac{1}{2}\|x - y\|_2^2$. Choosing $\alpha(t) = \log(2/t)$, $\beta(t) = \gamma(t) = 2\log(t)$, we recover the original Lagrangian in Eq. (1) and $\mathcal{O}(e^{-\beta(t)}) = \mathcal{O}(t^{-2})$, as derived in Su et al., 2016.

The formulation in Wibisono et al., 2016 has had a considerable impact on the recent developments in the theory of accelerated methods. Indeed, this approach can be used to **design and analyze new accelerated algorithms**. For instance, Xu et al., 2018 used the Lagrangian mechanics formalism to derive a novel simplified variant of accelerated stochastic mirror descent. Similarly, França et al., 2021, Muehlebach and Jordan, 2021

used the dual Hamiltonian formalism to study the link between **symplectic integration** of dissipative ODEs and acceleration. Due to its rising importance in the field of optimization, the topic was also presented by Prof. M. I. Jordan as a plenary lecture at the *International Congress of Mathematicians* in 2018 (Jordan, 2018), centered around the question "*what is the optimal way to optimize?*".

## 2.2 FAILURES OF THE VARIATIONAL PERSPECTIVE

*The object of mathematics is the honor of the human spirit.*
– Carl Gustav Jacob Jacobi

While the Lagrangian formalism has been inspiring for algorithm design and analysis, **its precise implications for the geometry and the path of accelerated solutions have not been examined in a mathematically rigorous way**. In Jordan, 2018 it is hinted that, since Nesterov's method solves the Euler-Lagrange equations, it **minimizes the action** functional $\int_{t_1}^{t_2} L_{\alpha,\beta,\gamma}(Y, \dot{Y}, t)dt$ over the space of curves by the *minimum action principle* of classical mechanics (Arnol'd, 2013). This claim[4] is inaccurate. Indeed, the term *minimum action principle* is misleading[5]: solving Euler-Lagrange only makes the action **stationary** (necessary condition: vanishing first-order derivative), but does not guarantee minimality — this only holds in physics for very special cases[6], which do not include even simple mechanical systems like the pendulum (proof in Section 36.2 of Gelfand and Fomin, 2000). Indeed, from a theoretical perspective, the claim of (Wibisono et al., 2016) requires computing the **second variation along Nesterov's path**. Quite surprisingly, even though many papers are dedicated to the variational formulation (Wibisono et al., 2016; Jordan, 2018; Casgrain, 2019; Duruisseaux and Leok, 2021), to the best of our knowledge there is no work which provides an in-depth rigorous study of the action relative to Bregman Lagrangian and that characterizes minimality of Nesterov in the space of curves.

**Question:** The variational perspective on Nesterov's method by Wibisono et al., 2016 suggests Nesterov's path is optimal under some metric on the space of trajectories. What are the implications of this? How can we use this finding to learn how to design optimal methods?

4 From Jordan, 2018: "*[...] we use standard calculus of variations to obtain a differential equation whose solution* ***is the path that optimizes*** *the time-integrated Bregman Lagrangian*".

5 From Gelfand and Fomin, 2000: "*The principle of least action is widely used [...]. However, in a certain sense the principle is not quite true [...]. We shall henceforth replace the principle of least action by the principle of stationary action. In other words, the actual trajectory of a given mechanical system* ***will not be required to minimize the action*** *but only to cause its first variation to vanish.*"

6 e.g. free particle in vanishing potentials, or $t_1 \approx t_2$.

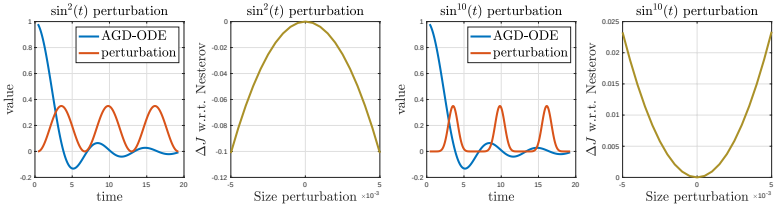


**Figure 2:** *Optimization of $f(x) = x^2/2$ using NAG-ODE. Peturbations (vanishing at extrema) are added to the NAG-ODE solution: depending on the perturbation kind (i.e. direction in space of curves), the local behavior is either a max or a min. Hence, Nesterov's path can be a saddle point for the action (formally shown in Section 2.2.2.1).*

**Answer (Zhang et al., 2021b):** The result of Wibisono et al., 2016 is unfortunately misleading. An in-depth analysis using tools from calculus of variations highlights that Nesterov's path is a saddle point for the action of its Lagrangian. We therefore suggest to take the narrative of Wibisono et al., 2016 very carefully.

IMPRECISE IMPLICATIONS OF THE VARIATIONAL FORMULATION.
Intrigued by the non-trivial open question of minimality of Nesterov's path and by the **enigmatic geometry of accelerated flows**, in this section, we examine the properties of accelerated gradient methods from the perspective of calculus of variations.

1. In Section 2.2.2 we study the minimality of classical Nesterov's ODE (damping $3/t$) proposed by Su et al., 2016 on multidimensional quadratic losses. By using **Jacobi's theory for the second variation** (summarized in Section 2.2.1), we find that Nesterov's path is optimal only if the integration interval $[t_1, t_2]$ is small enough. In contrast, if $t_2 - t_1 > \sqrt{40/\beta}$ ($\beta$ is Lipschitz constant for the gradient), Nesterov's path is actually a **saddle point** for the action (see Fig. 2).

2. In Section 2.2.2.5 we extend the analysis to the $\mu$-strongly convex setting and thus consider a constant damping $\alpha$. We show that, for extremely overdamped Nesterov flows ($\alpha \geq 2\sqrt{\beta}$), i.e for highly suboptimal parameter tuning (acceleration holds only for $\alpha \approx 2\sqrt{\mu}$),

Nesterov's path is always a minimizer for the action.
In contrast, we show that for $\alpha < 2\sqrt{\beta}$ (acceleration setting), if $t_2 - t_1 > 2\pi/\sqrt{4\beta - \alpha}$, Nesterov's path is again a saddle point.

3. In Section 2.2.3 we discuss the implications of our results for the theory of accelerated methods and propose a few interesting directions for future research.

We start by recalling some definitions and results from calculus of variations, which we adapt from classical textbooks (Landau and Lifshitz, 1976; Arnol'd, 2013; Gelfand and Fomin, 2000).

### 2.2.1 *Primer on Calculus of Variations*

We work on the vector space of curves $\mathcal{C}^1([t_1, t_2], \mathbb{R}^d)$ with $t_1, t_2 \in [0, \infty)$. We equip this space with the standard norm $\|Y\| = \max_{t_1 \le t \le t_2} \|Y(t)\|_2 + \max_{t_1 \le t \le t_2} \|\dot{Y}(t)\|_2$. Under this choice, for any regular Lagrangian $L$, the action functional below is continuous:

$$J[Y] := \int_{t_1}^{t_2} L(Y, \dot{Y}, t)dt. \tag{5}$$

FIRST VARIATION. Let $D$ be the linear subspace of continuously differentiable displacements curves $h$ such that $h(t_1) = h(t_2) = 0$. The corresponding *increment* of $J$ at $Y$ along $h$ is defined as $\Delta J[Y; h] := J[Y + h] - J[Y]$. Suppose that we can write $\Delta J[Y; h] = \varphi[Y; h] + \epsilon\|h\|$, where $\varphi$ is linear in $h$ and $\epsilon \to 0$ as $\|h\| \to 0$. Then, $J$ is said to be differentiable at $Y$ and the linear functional $\delta J[Y; \cdot] : D \to \mathbb{R}$ such that $\delta J[Y; h] := \varphi[Y; h]$ is called *first variation* of $J$ at $Y$. It can be shown that, if $J$ is differentiable at $Y$, then its first variation at $Y$ is unique.

EXTREMA. $J$ is said to have an *extremum* at $Y$ if $\exists \delta > 0$ such that, $\forall h \in D$ with $\|h\| \le \delta$, the sign of $J[Y + h] - J[Y]$ is constant. A *necessary* condition for $J$ to have an extremum at $Y$ is that

$$\delta J[Y; h] = 0, \quad \text{for all } h \in D. \tag{6}$$

The most well-known results in calculus of variations follows by using Taylor's theorem on $J[Y] = \int_{t_1}^{t_2} L(Y, \dot{Y}, t)dt$:

**Theorem 2.2.1** (Euler-Lagrange equation)**.** *A necessary condition for the curve $Y \in \mathcal{C}^1([t_1, t_2], \mathbb{R}^d)$ to be an extremum for $J$ (w.r.t. $D$) is that it satisfies the Euler-Lagrange equations* (2).

It is crucial to note that Theorem 2.2.1 provides a *necessary, but not sufficient* condition for an extremum — indeed, the next paragraph is completely dedicated to this.

SECOND VARIATION. Theorem 2.2.1 does not distinguish between extrema (maxima or minima) and saddles. For this purpose, we need to look at the *second variation*.
Suppose that the increment of $J$ at $Y$ can be written as

$$\Delta J[Y; h] = \varphi_1[Y; h] + \varphi_2[Y; h] + \epsilon \|h\|^2, \tag{7}$$

where $\varphi_1$ is linear in $h$, $\varphi_2$ is quadratic in $h$ and $\epsilon \to 0$ as $\|h\| \to 0$. Then $J$ is said to be *twice differentiable* and the functional $\delta^2 J[Y :, \cdot] : D \to \mathbb{R}$ s.t. $\delta^2 J[Y, h] := \varphi_2[Y; h]$ is called the *second variation* of $J$ at $Y$. Uniqueness of second variation is proved in the same way as the first variation.

**Theorem 2.2.2.** *A necessary condition for the curve $Y \in \mathcal{C}^1([t_1, t_2], \mathbb{R}^d)$ to be a local minimum for $J$ (w.r.t $D$) is that it satisfies $\delta^2 J[Y; h] \geq 0$. For local maxima, the sign is flipped.*

JACOBI EQUATIONS. Recall that $J[Y] = \int_{t_1}^{t_2} L(Y, \dot{Y}, t) dt$. Using the notation $L_{YZ} = \partial^2 L / (\partial Y \partial Z)$, the Taylor expansion for $\Delta J[Y; h] = J[Y + h] - J[Y]$ if $\|h\| \to 0$ converges to

$$\Delta J[Y; h] = \int_{t_1}^{t_2} \left( L_Y h + L_{\dot{Y}} \dot{h} \right) dt + \frac{1}{2} \int_{t_1}^{t_2} \left( L_{YY} h^2 + L_{\dot{Y}\dot{Y}} \dot{h}^2 + 2 L_{Y\dot{Y}} h \dot{h} \right) dt, \tag{8}$$

where the equality holds coordinate-wise.
Therefore, $\delta J[Y; h] = \int_{t_1}^{t_2} \left( L_Y h + L_{\dot{Y}} \dot{h} \right) dt$ and

$$\begin{aligned} \delta^2 J[Y; h] &= \frac{1}{2} \int_{t_1}^{t_2} \left( L_{YY} h^2 + 2 L_{Y\dot{Y}} h \dot{h} + L_{\dot{Y}\dot{Y}} \dot{h}^2 \right) dt \\ &= \frac{1}{2} \int_{t_1}^{t_2} \left( L_{YY} - \frac{d}{dt} L_{Y\dot{Y}} \right) h^2 dt + \frac{1}{2} \int_{t_1}^{t_2} L_{\dot{Y}\dot{Y}} \dot{h}^2 dt \\ &= \frac{1}{2} \int_{t_1}^{t_2} \left( P \dot{h}^2 + Q h^2 \right) dt, \end{aligned} \tag{9}$$

where

$$P = L_{\dot{Y}\dot{Y}}, \qquad Q = L_{YY} - \frac{d}{dt} L_{Y\dot{Y}}, \tag{10}$$

and the second equality follows from integration by parts since $h$ vanishes at $t_1$ and $t_2$. Using this expression, it is possible to derive an easy necessary (but not sufficient) condition for minimality.

**Theorem 2.2.3** (Legendre's necessary condition)**.** *A necessary condition for the curve $Y$ to be a minimum of $J$ is that $L_{\dot{Y}\dot{Y}}$ is positive semidefinite.*

CONJUGATE POINTS. A crucial role in the behavior of $\delta^2 J[Y;h]$ is played by the shape of the solutions to Jacobi's differential equation $\frac{d}{dt}(P\dot{h}) - Qh = 0$. A point $t \in (t_1, t_2)$ is said to be **conjugate** to point $t_1$ (w.r.t. $J$) if Jacobi's equation admits a solution which vanishes at both $t_1$ and $t$ but is not identically zero. We have the following crucial result.

**Theorem 2.2.4** (Jacobi's condition)**.** *Necessary and sufficient conditions for $Y$ to be a local minimum for $J$ are: (1) $Y$ satiesfies the Euler-Lagrange Equation; (2) $P$ positive definite; (3) $(t_1, t_2)$ contains no points conjugate to $t_1$.*

### 2.2.2 *Nesterov's Path is a Saddle Point for the Action*

This section is dedicated to the analysis of the action functional relative to Eq. (3). We start by a general abstract analysis in the convex quadratic case in Section 2.2.2.1, and then present an intuitive analytical computation in Section 2.2.2.2. The non-quadratic case is discussed in Section 2.2.2.4.

#### 2.2.2.1 *Solutions to Jacobi's Equation*

For the sake of clarity, we start by considering the Lagrangian in Eq. (1) for the simple one-dimensional case $f(x) = \beta x^2/2$. We have

$$Q = L_{YY} - \frac{d}{dt} L_{Y\dot{Y}} = -\beta t^3, \quad P = L_{\dot{Y}\dot{Y}} = t^3. \tag{11}$$

Therefore, Jacobi's equation for the action $\int_{t_1}^{t_2} L(Y, \dot{Y}, t)dt$ with $t_1 > 0$ is

$$\frac{d}{dt}(t^3\dot{h}) - \beta t^3 h = 0 \implies t^3\ddot{h} + 3t^2\dot{h} + \beta t^3 h = 0 \implies \ddot{h} + \frac{3}{t}\dot{h} + \beta h = 0, \tag{12}$$

which is itself Nesterov's ODE. Following the procedure outlined in Theorem 2.2.2, we now study the solutions $h$ such that $h(t_1) = 0$. Any solution to Eq. (12) can be written as[7]

$$h(t) = C\,\frac{\mathcal{Y}_1(\sqrt{\beta}\,t)}{t} - C\,\frac{\mathcal{Y}_1(\sqrt{\beta}\,t_1)\,\mathcal{J}_1(\sqrt{\beta}\,t)}{\mathcal{J}_1(\sqrt{\beta}\,t_1)\,t}, \tag{13}$$

where $C > 0$ specifies the initial velocity (see Fig. 3), $\mathcal{J}_\alpha$ is the Bessel function of the first kind and $\mathcal{Y}_\alpha$ is the Bessel function of the second kind.

$$\mathcal{J}_\alpha(x) = \sum_{m=0}^{\infty} \frac{(-1)^m}{m!\,\Gamma(m+\alpha+1)} \left(\frac{x}{2}\right)^{2m+\alpha}, \tag{14}$$

$$\mathcal{Y}_\alpha(x) = \frac{\mathcal{J}_\alpha(x)\cos(\alpha\pi) - \mathcal{J}_{-\alpha}(x)}{\sin(\alpha\pi)}. \tag{15}$$

Points $t > t_1$ conjugate to $t_1$ satisfy $h(t) = 0$, which results in the identity $\mathcal{Y}_1(\sqrt{\beta}\,t)/\mathcal{Y}_1(\sqrt{\beta}\,t_1) = \mathcal{J}_1(\sqrt{\beta}\,t)/\mathcal{J}_1(\sqrt{\beta}\,t_1)$. Remarkably, this condition does not depend on $C$, but only on $t_1$ and on the sharpness $\beta$. Let us now fix these parameters and name $K_{\beta,t_1} = \mathcal{Y}_1(\sqrt{\beta}\,t_1)/\mathcal{J}_1(\sqrt{\beta}\,t_1)$. Points conjugate to $t_1$ then satisfy $\mathcal{Y}_1(\sqrt{\beta}\,t) = K_{\beta,t_1}\mathcal{J}_1(\sqrt{\beta}\,t)$. Recall the following expansions (Watson, 1995), also used by Su et al., 2016:

$$\mathcal{J}_1(x) = \sqrt{\frac{2}{\pi x}}\left(\cos\left(x - \frac{3\pi}{4}\right) + \mathcal{O}\left(\frac{1}{x}\right)\right), \tag{16}$$

$$\mathcal{Y}_1(x) = \sqrt{\frac{2}{\pi x}}\left(\sin\left(x - \frac{3\pi}{4}\right) + \mathcal{O}\left(\frac{1}{x}\right)\right). \tag{17}$$

Since $\mathcal{J}_1$ and $\mathcal{Y}_1$ asymptotically **oscillate around zero** and are out of sync ($\pi/2$ difference in phase), for $t$ big enough the condition $\mathcal{Y}_1(\sqrt{\beta}\,t) = K_{\beta,t_1}\mathcal{J}_1(\sqrt{\beta}\,t)$ is fulfilled. This condition is going to be satisfied for a smaller value for $t$ if $\beta$ is increased, as confirmed by Figure 3.

**Theorem 2.2.5** (Local optimality of Nesterov with vanishing damping)**.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex quadratic, and let $X : \mathbb{R} \to \mathbb{R}^d$ be a solution to the ODE $\ddot{X} + \frac{3}{t}\dot{X} + \nabla f(X) = 0$. For $0 < t_1 < t_2$, consider the action functional $J[Y] = \int_{t_1}^{t_2} L(Y, \dot{Y}, t)dt$, mapping $Y \in \mathcal{C}^1([t_1, t_2], \mathbb{R}^d)$, to a real number. Then, if $|t_2 - t_1|$ is small enough, there are no points conjugate to $t_1$*

7 All symbolic computations are checked in Maple/Mathematica.

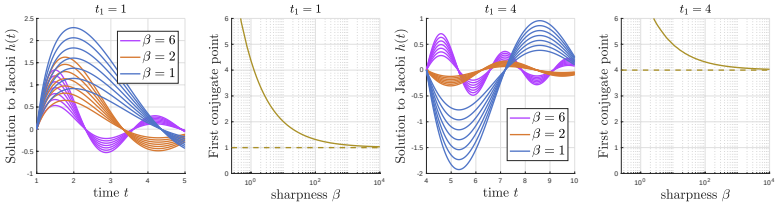


**Figure 3:** *First conjugate point to $t_1 = 1, 4$ for quadratics $\beta x^2/2$, under the settings of Section 2.2.2.1. For each value of $\beta$, six solutions $h(t)$ to the Jacobi equation (each one has different velocity) are shown.*

*and Nesterov's path minimizes $J$ over all curves such that $Y(t_1) = X(t_1)$ and $Y(t_2) = X(t_2)$. The length of the optimality interval $|t_2 - t_1|$ shrinks as $\beta$, the maximum eigenvalue of the Hessian of $f$, increases.*

*Proof.* The argument presented in this section can be lifted to the multidimensional case. Indeed, since the dynamics in phase space is linear, it's geometry is invariant to rotations and we can therefore assume the Hessian is diagonal. Next, Jacobi's equation has to be solved coordinate-wise, which leads to a logical AND between conjugacy conditions. By the arguments above, the dominating condition is the one relative to the maximum eigenvalue $\beta$. □

The following corollary shows that **Nesterov's path actually becomes suboptimal if the considered time interval is big enough**. This is also verified numerically in Figure 2.

**Corollary 2.2.1** (Nesterov with vanishing damping is not globally optimal)**.** *In the settings of Theorem 2.2.5, for $|t_2 - t_1|$ big enough, Nesterov's path becomes a saddle point for $J$.*

*Proof.* Non-existence of conjugate points is necessary and sufficient for minimality/maximality. □

In the next subsection, we provide a constructive proof for Cor. 2.2.1, which allows us to derive a concrete bound for $|t_2 - t_1|$. In particular, we show in Prop. 2.2.1 that, for the case of vanishing damping $3/t$, Nesterov's path is always a saddle point for the action if $|t_2 - t_1| > \sqrt{40/\beta}$. But first, we present two remarks.

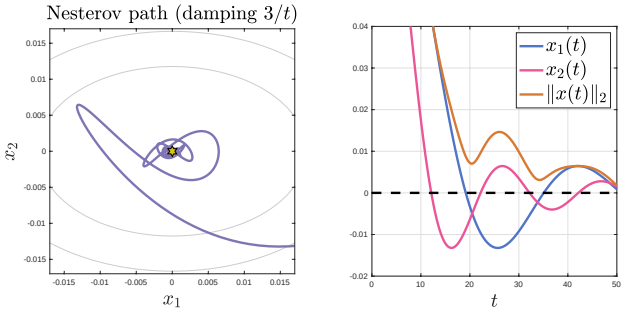


**Figure 4:** *Dynamics of Nesterov's method on two-dimensional quadratics.*

OPTIMALITY FOR THE SPECIAL CASE $X(t_2) = x^*$. Some readers might have already realized that Jacobi's equation Eq. (12) in the quadratic potential case is itself the solution of Nesterov's ODE. This means that, if $t_1 \approx 0$, **the first time conjugate to $t_1$ is exactly when Nesterov's path reaches the minimizer**. Hence — only in the one-dimensional case — it is actually true that, if we do not consider arbitrary time intervals but only the first interval before the solution first touches the optimizer, Nesterov's path always minimizes the action. Sadly, this strong and interesting result is not valid in higher dimensions, since Nesterov's path in general never actually crosses the minimizer in finite time (see Figure 4): the first crossing time in each direction depends on the sharpness in each direction, hence **by the time each coordinate reaches zero, we already have a conjugate point**.

REMARK ON DROPPING BOUNDARY CONDITIONS. The results in this section are formulated for the fixed boundaries case $Y(t_1) = X(t_1)$ and $Y(t_2) = X(t_2)$, where $X$ is the solution to Nesterov's ODE. From an optimization viewpoint, this requirement seems strong. Ideally, we would want $X(t_2)$ to be *any point close to the minimizer* (say inside an $\epsilon$-ball). A simple reasoning proves that Nesterov's path can be a saddle point for the action also in this case. By contradiction, assume Nesterov's trajectory minimizes the action among all curves that reach any point inside a small $\epsilon$-ball at time $t_2$. Then, Nesterov's path also minimizes the action in the (smaller) space of curves that reach exactly $X(t_2)$. By Cor. 2.2.1, this leads to a contradiction if $t_2$ is big enough.

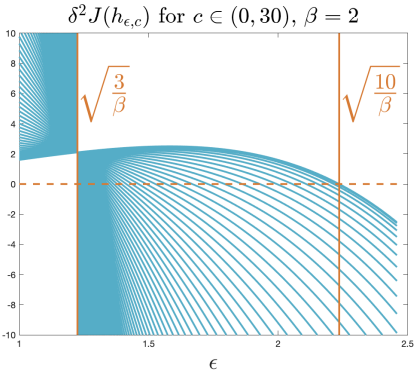


**Figure 5:** *Plot of Equation (19).*

#### 2.2.2.2 *A constructive proof of Nesterov's suboptimality*

We now present a direct computation, to shed some light on the suboptimality of Nesterov's path in the context of Theorem 2.2.5. In the setting of Section 2.2.2.1, the second variation of $J$ along $\gamma$ is $\frac{1}{2}\int_{t_1}^{t_2} t^3[\dot{h}(t)^2 - \beta h(t)^2]dt$, independent of $\gamma$. Consider now the finite-norm perturbation (vanishing at boundary):

$$\tilde{h}_{\epsilon,c}(t) = \begin{cases} 0 & t \leq c-\epsilon \text{ or } t \geq c+\epsilon \\ \frac{t-c+\epsilon}{\epsilon} & t \in (c-\epsilon, c) \\ \frac{c+\epsilon-t}{\epsilon} & t \in (c, c+\epsilon) \end{cases}, \tag{18}$$

for $c \in (t_1, t_2)$ and $\epsilon < \min(c - t_1, t_2 - c)$. This is a triangular function with support $(c-\epsilon, c+\epsilon)$ and height one. Let $h_{\epsilon,c}$ be a $\mathcal{C}^1$ modification of $\tilde{h}_{\epsilon,c}$ such that $\|\tilde{h}_{\epsilon,c} - h_{\epsilon,c}\|$ is negligible[8]. For any scaling factor $\sigma > 0$,

$$\delta^2 J(\sigma \cdot h_{\epsilon,c}) = -\sigma^2 \frac{\left(\frac{3\beta\epsilon^4}{10} + (\beta c^2 - 3)\epsilon^2 - 3c^2\right) c}{3\epsilon}. \tag{19}$$

The denominator is always positive. Hence, we just need to study the sign of the numerator, with respect to changes in $\epsilon > 0$ and $c > 0$. Con-

8 Standard technique in calculus of variation, see e.g. proof of Legendre's Thm (Gelfand and Fomin, 2000).

sider for now $c$ fixed, then the zeros of the numerator are at $(15 - 5u \pm \sqrt{25u^2 - 60u + 225})/(3\beta)$, with $u := \beta c^2$. Since $25u^2 - 60u + 225 > 0$ for all $u \geq 0$, the solution has two real roots. However, only one root $\epsilon_*^2(u, \beta)$ is admissible, since the smallest one is always negative[9]. As a result, for fixed $c > 0$, $\delta^2 J(\sigma h_{\epsilon,c})$ changes sign only at $\epsilon_*^2(u, \beta)$. Note that $\epsilon_*^2(u, \beta)$ is decreasing as a function of $u$ and $\epsilon_*^2(0, \beta) = 10/\beta$ as well as $\lim_{u\to\infty} \epsilon_*^2(u, \beta) = 3/\beta$. Therefore, for any $c, \beta > 0$, we showed that $\delta^2 J(\sigma h_{\epsilon,c})$ changes sign when $\epsilon_* \in [\sqrt{3/\beta}, \sqrt{10/\beta}]$. If choosing $\epsilon$ big is allowed by the considered interval ($h_{\epsilon,c}$ has to vanish at $t_1, t_2$), then the second variation is indefinite. This happens if $|t_2 - t_1| > 2\epsilon_*$. By taking $\sigma \to 0$, we get the following result.

**Proposition 2.2.1** (Sufficient condition for saddle)**.** *The second variation of the action of Nesterov's Lagrangian (damping $3/t$) on $f(x) = \beta x^2/2$ is an indefinite quadratic form for $|t_2 - t_1| > \sqrt{40/\beta}$. This result generalizes to $\beta$-smooth multidimensional convex quadratics.*

We remark that the inverse dependency on the square root of $\beta$ is also predicted by the general proof in Section 2.2.2.1, where the argument of the Bessel functions is always $t\sqrt{\beta}$.

#### 2.2.2.3 *Unboundedness of the action for large integration intervals*

In Prop. 2.2.1, we showed that for big enough integration intervals, Nesterov's method with damping $3/t$ on $f(x) = \beta x^2/2$ is saddle point for the action. This suggests that **the action is itself unbounded** — both from above and below. It is easy to show this formally.

**Proposition 2.2.2** (Unboundedness of the action)**.** *Let $L$ be Lagrangian of Nesterov's method with damping $3/t$ on a $\beta$-smooth convex quadratic and $J[Y] = \int_{t_1}^{t_2} L(Y, \dot{Y}, t)dt$. Let $a, b$ be two arbitrary vectors in $\mathbb{R}^d$. There exists a sequence of curves $(Y_k)_{k\in\mathbb{N}}$, with $Y_k \in \mathcal{C}^1([t_1, t_2], \mathbb{R}^d)$ and $Y_k(t_1) = a, Y_k(t_2) = b$ for all $k \in \mathbb{N}$, such that $J[Y_k] \overset{k}{\to} \infty$. In addition, if $|t_2 - t_1| > \sqrt{40/\beta}$ there exists another sequence with the same properties diverging to $-\infty$.*

*Proof.* The proof is based on the computation performed for Prop. 2.2.1. Crucially, note that for the quadratic loss function case we have $\delta^2 J = J$. For the case $a = b = 0$, we showed that for any interval $[t_1, t_2]$, by

9 If $u > 0$, then $15 - 5u - \sqrt{25u^2 - 60u + 225} < 0$.

picking $\epsilon$ small enough, we have $J(h_{\epsilon,c}) = \delta^2 J(h_{\epsilon,c}) > 0$ (also illustrated in the figure supporting the proof). Hence, $J(\sigma \cdot h_{\epsilon,c}) \to +\infty$ as $\sigma \to \infty$. Same argument holds for $-\infty$ in the large interval case. This proves the assertion for $a = b = 0$. Note that the curves corresponding to the diverging sequences can be modified to start/end at any $a, b \in \mathbb{R}^d$ at the price of a bounded error in the action. This does not modify the behavior in the limit; hence, the result follows. □

#### 2.2.2.4 *Optimality of Nesterov with vanishing damping if curvature vanishes*

Note that the bound on $|t_2 - t_1|$ in Prop. 2.2.1 gets loose as $\beta$ decreases. This is also predicted by the argument with Bessel functions in Section 2.2.2.1, and clear from the simulation in Fig. 3. As a result, as curvature vanishes, Nesterov's path becomes optimal for larger and larger time intervals. This setting is well described by polynomial losses $f(x) \propto (x - x^*)^p$, with $p > 2$. As Nesterov's path approaches the minimizer $x^*$, the curvature vanishes; hence, for every $\beta > 0$ there exists a time interval $(t_1, \infty)$ where the curvature along Nesterov's path is less then $\beta$. This suggest that, **for losses with vanishing curvature at the solution, there exists a time interval** $(t_*, \infty)$ **where Nesterov's path is actually a minimizer for the action**. While this claim is intuitive, it is extremely **hard to prove** formally since in this case the second variation of $J$ depends on the actual solution of Nesterov's equations — for which no closed-form formula is known in the polynomial case (Su et al., 2016).

However, we also note that the vanishing sharpness setting is only interesting from a theoretical perspective. Hence, it is safe to claim that in the machine learning setting Nesterov's path is only optimal for small time intervals, as shown in Theorem 2.2.5.

#### 2.2.2.5 *Analysis of the action under constant damping*

In the $\mu$-**strongly convex** case[10], it is well known that a **constant damping** $\alpha = 2\sqrt{\mu}$ yields **acceleration** compared to gradient descent[11]. This

10 Hessian eigenvalues lower bounded by $\mu > 0$.

11 The corresponding rate is linear and depends on $\sqrt{\mu/\beta}$, as opposed to $\mu/\beta$ (GD case).

choice completely changes the geometry of Nesterov's path and needs a separate discussion. The corresponding Lagrangian is

$$L_\alpha(Y, \dot{Y}, t) = e^{\alpha t}\left(\frac{1}{2}\|\dot{Y}\|^2 - f(Y)\right). \tag{20}$$

Again, we consider the quadratic function $f(x) = \frac{\beta x^2}{2}$ and examine Jacobi's ODE $\ddot{h}(t) + \alpha\dot{h}(t) + \beta h(t) = 0$. We have to determine whether there exists a non-trivial solution such that $h(t_1) = h(t_2) = 0$ and $h(t)$ vanishes also at a point $t \in (t_1, t_2)$, the conjugate point.
For the **critical damping case** $\alpha = 2\sqrt{\beta}$ the general solution such that $h(t_1) = 0$ is

$$h(t) = Ce^{-\sqrt{\beta}t}(t - t_1). \tag{21}$$

There is no non-trivial solution $h$ that vanishes also at $t \in (t_1, t_2)$ — **no conjugate points**. The **same holds** for the **overdamping case** $\alpha > 2\sqrt{\beta}$, where the solution that vanishes at $t_1$ is

$$h(t) = Ce^{-\frac{\alpha t}{2}}\left(e^{\frac{1}{2}\sqrt{\alpha^2 - 4\beta}t} - e^{\frac{1}{2}\sqrt{\alpha^2 - 4\beta}(2t_1 - t)}\right). \tag{22}$$

For the **underdamping case** $\alpha < 2\sqrt{\beta}$, the picture gets more similar to the vanishing damping case (Section 2.2.2.1). The solution under $h(t_1) = 0$ is

$$Ce^{-\frac{\alpha t}{2}}\left(\sin\left(\frac{\sqrt{4\beta - \alpha^2}}{2}t\right) - \tan\left(\frac{\sqrt{4\beta - \alpha^2}}{2}t_1\right)\cos\left(\frac{\sqrt{4\beta - \alpha^2}}{2}t\right)\right). \tag{23}$$

Hence all points $t > t_1$ conjugate to $t_1$ satisfy $t = t_1 + 2k\pi/\sqrt{4\beta - \alpha^2}$ for $k \in \mathbb{N}$. Therefore for any $t_2 > t_1 + 2\pi/\sqrt{4\beta - \alpha^2}$ there **exists a conjugate point** $t \in (t_1, t_2)$.

**Theorem 2.2.6** (Global optimality of overdamped Nesterov, suboptimality of accelerated Nesterov)**.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a strongly convex quadratic, and let $X : \mathbb{R} \to \mathbb{R}^d$ be a solution to the ODE $\ddot{X} + \alpha\dot{X} + \nabla f(X) = 0$. For $0 \le t_1 < t_2$, consider the action $J[Y] = \int_{t_1}^{t_2} L_\alpha(Y, \dot{Y}, t)dt$. If $\alpha \ge 2\sqrt{\beta}$, where $\beta$ is the max. eigenvalue of the Hessian of $f$, then Nesterov's path minimizes $J$ over all curves s.t. $Y(t_1) = X(t_1)$ and $Y(t_2) = X(t_2)$. Else (e.g. acceleration setting $\alpha \approx 2\sqrt{\mu}$), Nesterov's path is optimal only for $|t_2 - t_1| \le 2\pi/\sqrt{4\beta - \alpha^2}$ and otherwise is a saddle point.*

*Proof.* As for the proof of Theorem 2.2.5, the condition on conjugate points has to hold for each eigendirection separately. We conclude by noting that eigenvalues are in the range $[\mu, \beta]$. □

For the underdamping case, we give a **concrete example** for $\alpha = \beta = 1$, to show the saddle point nature. Consider the finite-norm perturbation $h(t) = \sin(k\pi(t - t_1)/(t_1 - t_2))$, where $k \in \mathbb{N}$. Then,

$$\delta^2 J[\gamma](\sigma h) = \sigma^2 e^{2t_1} \left( \frac{k^2 e^{-(t_2 - t_1)} \pi^2 (e^{t_2 - t_1} - 1)(2k^2\pi^2 - (t_2 - t_1)^2)}{(t_2 - t_1)^2 (4k^2\pi^2 + (t_2 - t_1)^2)} \right). \tag{24}$$

Hence, for any $t_2 - t_1 > \sqrt{2}k\pi$, it holds that $\delta^2 J[\gamma](\sigma h) < 0$.

EXTENDING CLAIMS TO $t_2 = \infty$ WITH $\Gamma$-CONVERGENCE. From an optimization viewpoint, the most interesting setting is to study the **action over the complete trajectory**, i.e. to consider $Y \in \mathcal{C}^1([t_1, \infty), \mathbb{R}^d)$ such that $Y(t_1) = X(t_1)$ and $Y(\infty) = x^*$, a minimizer. Prop. 2.2.1 and Theorem 2.2.6 show that the question of optimality in this case **deserves a discussion only in the extremely overdamped case** $\alpha \geq 2\sqrt{\beta}$, where minimality is guaranteed for any time interval. A careful study of the infinite-time setting would require the theory of $\Gamma$-convergence (Braides et al., 2002). The usual pipeline consists in defining a sequence of problems $J_k$, on intervals $[t_1, t_2^k]$, with $t_2^k \to \infty$ as $k \to \infty$. Under the assumption that each $J_k$ admits a global minimizer (only true for the overdamped case), one can study convergence of $J_k^* = \min\{J_k[Y] : Y \in \mathcal{C}^1([t_1, t_2^k], \mathbb{R}^d)\}$ to $J_\infty^* = \min\{J_\infty(Y) : Y \in \mathcal{C}^1([t_1, \infty), \mathbb{R}^d)\}$. While existence of $J_\infty^*$ and $J_\infty$ is not trivial in general, for our setting the pipeline directly yields minimality of overdamped Nesterov's path until $\infty$.

### 2.2.3 *Discussion of this Negative Result*

In this section, we summarize the results of Section 2.2.2 & 2.2.2.5 and discuss some implications that our analysis delivers on the geometry of accelerated flows in the convex and strongly convex setting.
We summarize below the main high-level findings of our analysis:

1. The **optimality** of Nesterov's path for minimization of the action corresponding to the Bregman Lagrangian is **strictly linked to the curvature** around the minimizer reached by the flow.

2. As the maximal curvature $\beta$ increases, **it gets increasingly difficult for accelerated flows to minimize the action over long integration intervals**: both the accelerated ODEs $\ddot{X} + 3/t\dot{X} + \nabla f(X) = 0$ and $\ddot{X} + 2\sqrt{\mu}\dot{X} + \nabla f(X) = 0$ are optimal only for intervals of length proportional to $1/\sqrt{\beta}$.

3. If Nesterov's path does not minimize the action, there does not exist a "better" (through the eyes of the action) algorithm, as the **functional gets unbounded** from below (Section 2.2.2.3).

4. This **suboptimality is due precisely to the oscillations** in the accelerated paths. As long as each coordinate in parameter space decreases monotonically (as it is the case for gradient descent), Nesterov's path is optimal. See also O'donoghue and Candes, 2015.

5. Hence, Nesterov's method with **very high damping** $\alpha > 2\sqrt{\beta}$ — which does not oscillate and hence **does not lead to acceleration** (see Fig. 6) — minimizes the action.

In a nutshell, locally Nesterov's method does indeed optimize a functional over curves. However, this property breaks down precisely when the geometry gets interesting — i.e. when the loss evolution is non-monotonic. Since acceleration is a global phenomenon, i.e. is the cumulative result of many consecutive oscillations (see Fig. 6), our results suggest that the essence of *acceleration cannot be possibly captured by the minimization of the action relative to the Bregman Lagrangian*.

NON-UNIQUENESS OF THE LAGRANGIAN. A possible reason for the non-optimality of Nesterov's path is, simply put — that we are not looking at the right action functional. Indeed, there are many Lagrangians that can generate Nesterov ODE. Let $F(Y, t)$ be any function which does not involve the velocity, then it is easy to see that the Lagrangian $L$ is equivalent to

$$\tilde{L}(Y, \dot{Y}, t) = L(Y, \dot{Y}, t) + \left\langle \dot{Y}, \frac{\partial F}{\partial Y}(X, t) \right\rangle + \frac{\partial F}{\partial t}(X, t). \tag{25}$$

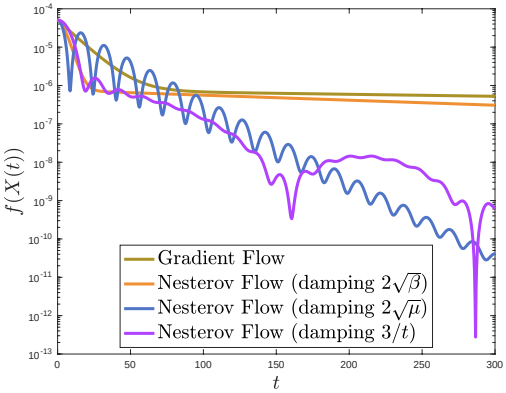


**Figure 6:** *Optimization of potential $f(x, y) = 0.02x^2 + 0.0004y^2$, where $2e - 2 = \beta \gg \mu = 3e - 4$. Non-monotonic trajectories (i.e. the accelerated curves) minimize the action only for short time intervals. Simulation with Runge-Kutta 4 integration.*

This simple fact opens up new possibilities for analyzing and interpreting Nesterov's method using different functionals — which perhaps have *both a more intuitive form and better properties*.

HIGHER ORDER ODES. On a similar note, it could be possible to *convexify the action functional* by considering a logical OR between symmetric ODEs, e.g $(\frac{d^2}{dt^2} + \alpha\frac{d}{dt} + \beta)(\frac{d^2}{dt^2} - \alpha\frac{d}{dt} + \beta)X = 0$. Such tricks are often used in the literature on dissipative systems (Szegleti and Márkus, 2020).

NOETHER THEOREM. In physics, the variational framework is actually never used to claim the minimality of the solution to the equations of motion. Its power relies almost completely in the celebrated Noether's Theorem (Noether, 1918), which laid the foundations for modern quantum field theory by linking the symmetries in the Lagrangian (and of the Hamiltonian) to the invariances in the dynamics. Crucially, for the application of Noether's Theorem, one only needs the ODE to yield a stationary point for the action (also saddle points work). Coincidentally, while finalizing this manuscript, two preprints (Tanaka and Kunin, 2021; Głuch and Urbanke, 2021) came out on some implications of Noether's Theo-

rem for optimization. However, we note that these works do not discuss the direct link between Noether's Theorem and acceleration, but instead study the interaction between the symmetries in neural network landscapes and optimizers. While some preliminary implications of Noether's theorem for time-rescaling of accelerated flows are discussed in Wibisono et al., 2016, we suspect that a more in-depth study could lead, in combination with recent work on the Hamiltonian formalism (Diakonikolas and Jordan, 2019), to substantial insights on the hidden invariances of accelerated paths. We note that finding these invariances might not be an easy task and requires dedicated work: indeed, even for simple linear damped harmonic oscillators (constant damping), invariance in the dynamics can be quite complex (Choudhuri et al., 2008).

## 2.3 UNDERSTANDING STOCHASTIC MOMENTUM WITH SDES

*Among all of the mathematical disciplines the theory of differential equations is the most important... It furnishes the explanation of all those elementary manifestations of nature which involve time.*

– Sophus Lie.

**Question:** In Section 2.2, we showed that the variational perspective on Nesterov's method by Wibisono et al., 2016 cannot be used in the current format to gain intuition. Does this mean that the continuous-time perspective cannot possibly provide insights in the mechanisms of acceleration and momentum?

**Answer (Orvieto et al., 2019):** The stochastic differential equation model for momentum can be used to gain insights into its gradient averaging and amplification features. In this section, we introduce a new class of algorithms generalizing Heavy-ball (HB) to different gradient memory mechanisms. We also show how the continous-time perspective can be leveraged to guide discrete-time analysis and to bring to light interesting features of (stochastic) Nesterov's method — which we show can be viewed as a time-changed linear memory equation.

Already in his 1964 paper (Polyak, 1964), Polyak motivated momentum as the discrete-time analogue of a second-order ODE:

$$\ddot{X}(t) + a(t)\dot{X}(t) + \nabla f(X(t)) = 0, \tag{HB-ODE}$$

which can be written in phase-space as

$$\begin{cases} \dot{V}(t) = -a(t)V(t) - \nabla f(X(t)) \\ \dot{X}(t) = V(t) \end{cases}. \tag{HB-ODE-PS}$$

This connection can be made precise: Polyak momentum is indeed the result of **semi-implicit Euler integration** (Zhang et al., 2021c).
If the viscosity parameter $\alpha = a(t)$ is time-independent, HB-ODE, with

initial condition $\dot{X}(0) = 0$ and $X(0) = x_0$, can be cast into an integro-differential equation[12]:

$$\dot{X}(t) = -\int_0^t e^{-\alpha\cdot(t-s)}\nabla f(X(s))ds. \tag{HB-ODE-INT}$$

### 2.3.1 *Memory interpretation of Heavy-ball*

Notice that the instantaneous update direction of HB-ODE-INT is a weighted average of the past gradients, namely

$$\dot{X}(t) = -\int_0^t w(s,t)\nabla f(X(s))ds, \tag{26}$$

with $w(s,t) := e^{\alpha(t-s)}$. However, the **weights do not integrate to one**. Indeed, for all $t$, we have $\int_0^t w(s,t)ds = (1-e^{-\alpha t})/\alpha$, which integrates to $1/\alpha$ as $t \to \infty$. As a result, in the constant gradient setting, the previous sum is a **biased estimator** of the actual gradient. This fact suggests a simple modification of HB-ODE-INT, for $t > 0$:

$$\dot{X}(t) = -\frac{\alpha}{1-e^{-\alpha t}}\int_0^t e^{-\alpha\cdot(t-s)}\nabla f(X(s))ds, \tag{27}$$

which we write as $\dot{X} = -\frac{\alpha}{e^{\alpha t}-1}\int_0^t e^{\alpha s}\nabla f(X(s))ds$. We note that this **normalization** step follows exactly the same motivation as bias correction in Adam (Kingma and Ba, 2014). If we define $\mathfrak{m}(t) := e^{\alpha t} - 1$, the previous formula takes the form:

$$\dot{X}(t) = -\int_0^t \frac{\dot{\mathfrak{m}}(s)}{\mathfrak{m}(t)}\nabla f(X(s))ds. \tag{MG-ODE-INT}$$

This **memory-gradient integro-differential equation** (MG-ODE-INT) provides a generalization of HB-ODE-INT, with bias correction. The following lemma is consequence of the fundamental theorem of calculus.

**Lemma 2.3.1.** *For any $\mathfrak{m} \in \mathcal{C}^1(\mathbb{R},\mathbb{R})$ s.t. $\mathfrak{m}(0) = 0$, MG-ODE-INT is normalized : $\int_0^t \frac{\dot{\mathfrak{m}}(s)}{\mathfrak{m}(t)}ds = 1$, for all $t > 0$.*

*Proof.* Since $\mathfrak{m}(0) = 0$, $\int_0^t \dot{\mathfrak{m}}(s)ds = \mathfrak{m}(t)$. □

12 Using the fundamental theorem of calculus and plugging in $\dot{X}(0) = 0$.

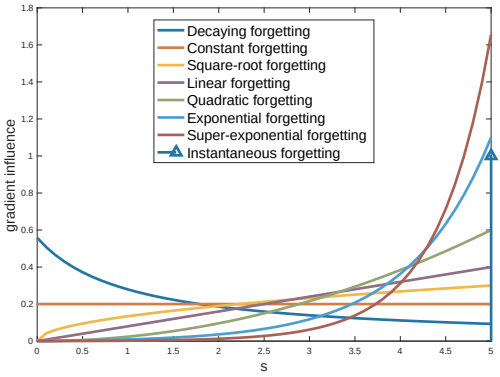


**Figure 7:** *Illustration of the influence of past gradients on $\dot{X}(6)$ (i.e. the right hand side of MG-ODE-INT with $t = 5$). The corresponding memory function can be found in Tb. 1. The influence is computed as $\dot{\mathfrak{m}}(s)/\mathfrak{m}(6)$. By Lemma 2.3.1, the area under all curves is 1.*

Based on Lemma 2.3.1, we will always set $\mathfrak{m}(0) = 0$. What other properties shall a general $\mathfrak{m}(\cdot)$ have? Requiring $\dot{\mathfrak{m}}(s) \neq 0$ for all $s \geq 0$ ensures that there does not exist a time instant where the gradient is systematically discarded. Hence, since $\mathfrak{m}(0) = 0$, $\mathfrak{m}(\cdot)$ should be either monotonically decreasing and negative or monotonically increasing and positive. In the latter case, without loss of generality, we can flip its sign. This motivates the following definition.

**Definition 2.3.1.** $\mathfrak{m} \in \mathcal{C}^1(\mathbb{R}_+, \mathbb{R})$ *is a* ***memory function*** *if it is non-negative, strictly increasing and s.t.* $\mathfrak{m}(0) = 0$.

For example, $e^{\alpha t} - 1$, from which we started our discussion, is a valid memory function. Crucially, we note that $\dot{\mathfrak{m}}(\cdot)$ plays the important role of controlling the **speed at which we forget previously observed gradients**. For instance, let $\mathfrak{m}(t) = t^3$; since $\dot{\mathfrak{m}}(s) = 3s^2$, the system forgets past gradients *quadratically fast*. In contrast, $\mathfrak{m}(t) = e^{\alpha t} - 1$ leads to *exponential forgetting*. Some interesting memory functions are listed in Tb. 1, and their respective influence on past gradients is depicted in Fig. 7. We point out that, in the limit $\alpha \to \infty$, the weights $w(s,t) = \frac{\dot{\mathfrak{m}}(s)}{\mathfrak{m}(t)}$ associated with exponential forgetting converge to a Dirac distribution $\delta(t - s)$. Hence, we recover the Gradient Descent ODE (Mertikopoulos and Staudigl, 2018):

$\dot{X}(t) = -\nabla f(X(t))$. For the sake of comparison, we will refer to this as *instantaneous forgetting*.
Finally, notice that MG-ODE-INT can be written as a second order ODE. Too see this, we just need to compute the second derivative. For $t > 0$ we have that

$$\ddot{X}(t) = \frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)^2} \int_0^t \dot{\mathfrak{m}}(s) \nabla f(X(s)) ds - \frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)} \nabla f(X(t)). \tag{28}$$

Plugging in the definition of $\dot{X}$ from the integro-differential equation, we get the memory-gradient ODE:

$$\ddot{X}(t) + \frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)} \dot{X}(t) + \frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)} \nabla f(X(t)) = 0. \tag{MG-ODE}$$

| **Forgetting** | **Memory $\mathfrak{m}$** | **ODE Coeff. $\dot{\mathfrak{m}}/\mathfrak{m}$** |
|---|---|---|
| Decaying | $\log(1+t)$ | $1/(t\log(t+1))$ |
| Constant | $t$ | $1/t$ |
| Square-root | $t^{1.5}$ | $1.5/t$ |
| Linear | $t^2$ | $2/t$ |
| Quadratic | $t^3$ | $3/t$ |
| Exponential | $e^{\alpha t} - 1$ | $\alpha e^{\alpha t} / \left(e^{\alpha t} - 1\right)$ |
| Super-exp | $e^{t^\alpha} - 1$ | $\alpha t^{\alpha-1} e^{t^\alpha} / \left(e^{t^\alpha} - 1\right)$ |
| Instantaneous | – | – |

**Table 1:** *Some examples of memory functions.*

Equivalently, we can transform this second order ODE into a system of two first order ODEs by introducing the variable $V(t) := \dot{X}(t)$ and noting that $\dot{V}(t) = -\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)} V(t) - \frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)} \nabla f(X(t))$. This is the *phase-space representation* of MG-ODE, which we use in Section 2.3.2 to provide the extension to the stochastic setting.

EXISTENCE AND UNIQUENESS. Readers familiar with ODE theory probably realized that, since by definition $\mathfrak{m}(0) = 0$, the question of existence and uniqueness of the solution to MG-ODE is not trivial. This is

why we stressed its validity for $t > 0$ multiple times during the derivation. Indeed, it turns out that solutions may not exist globally on $[0, \infty)$ (see appendix of Orvieto et al., 2019 for a counterexample). Nevertheless, if we allow to start integration from *any* $\epsilon > 0$ and assume $f(\cdot)$ to be $L$-smooth, standard ODE theory (see App. A) ensures that the sought solution exists and is unique on $[\epsilon, \infty)$. Since $\epsilon$ can be made as small as we like this apparent issue can be regarded as an artifact of the model.

### 2.3.2 *Memory of Stochastic Gradients*

In this section we introduce stochasticity in the MG-ODE model. As already mentioned in the introduction, at each step $k$, iterative stochastic optimization methods have access to an estimate $\mathcal{G}(x_k)$ of $\nabla f(x_k)$: the so called *stochastic gradient*. This information is used and possibly combined with previous gradient estimates $\mathcal{G}(x_0), \dots, \mathcal{G}(x_{k-1})$, to compute a new approximation $x_{k+1}$ to the solution $x^*$. There are many ways to design $\mathcal{G}(k)$: the simplest (Robbins and Monro, 1951) is to take $\mathcal{G}_{\text{MB}}(x_k) := \nabla f_{i_k}(x_k)$, where $i_k \in \{1, \dots, n\}$ is a uniformly sampled datapoint. This gradient estimator is trivially unbiased (conditioned on past iterates) and we denote its covariance matrix at point $x$ by $\Sigma(x) = \frac{1}{n}\sum_{i=1}^{n}(\nabla f_i(x) - \nabla f(x))(\nabla f_i(x) - \nabla f(x))^\top$.
Following Krichene and Bartlett, 2017 we model stochasticity adding a volatility term in MG-ODE:

$$\begin{cases} dX(t) = V(t)dt \\ dV(t) = -\dfrac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}V(t)dt - \dfrac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}[\nabla f(X(t))dt + \sigma(X(t))dB(t)] \end{cases}, \tag{MG-SDE}$$

where $\sigma(X(t)) \in \mathbb{R}^{d\times d}$ and $\{B(t)\}_{t\geq 0}$ is a standard **Brownian Motion**. For a primer on stochastic differential equations and Brownian Motion, please check Appendix A. Notice that this system of equations reduces to the phase-space representation of MG-ODE if $\sigma(X(t))$ is the null matrix. The connection from $\sigma(x)$ to the gradient estimator covariance matrix $\Sigma(x)$ can be made precise: Li et al., 2017 motivate the choice $\sigma(x) = \sqrt{h\Sigma(x)}$, where $\sqrt{\cdot}$ denotes the principal square root and $h$ is the discretization stepsize.

The proof of existence and uniqueness to the solution of this SDE[13] relies on the same arguments made for MG-ODE, with one additional crucial difference: Orvieto and Lucchi, 2019a showed that $f(\cdot)$ needs to additionally be three times continuously differentiable with bounded third derivative (i.e. $f \in \mathcal{C}_b^3(\mathbb{R}^d, \mathbb{R})$), in order for $\sigma(\cdot)$ to be Lipschitz continuous. Hence, we will assume this regularity and refer the reader to Orvieto and Lucchi, 2019a for further details.

### 2.3.3 *Insights on Nesterov's Method*

Su et al., 2016 showed that the continuous-time limit of Nesterov's Accelerated Gradient (NAG) for convex functions is HB-ODE with time-dependent viscosity $3/t$: $\ddot{X}(t) + \frac{3}{t}\dot{X}(t) + \nabla f(X(t)) = 0$, which we refer to as *Nesterov's ODE* (NAG-ODE in Section 2.2). Using Bessel functions, the authors were able to provide a new insightful description and analysis of this mysterious method. In particular, they motivated how the **vanishing viscosity is essential for acceleration**[14]. Indeed, the solution to the equation above is s.t. $f(X(t)) - f(x^*) \leq \mathcal{O}(1/t^2)$; in contrast to the solution to the GD-ODE $\dot{X}(t) = -\nabla f(X(t))$, which only achieves a rate $\mathcal{O}(1/t)$.
A closer look at Tb. 1 reveals that the choice of viscosity $3/t$ is related to MG-ODE with **quadratic forgetting**, that is $\ddot{X}(t) + \frac{3}{t}\dot{X}(t) + \frac{3}{t}\nabla f(X(t)) = 0$. However, in MG-SDE the gradient term is also pre-multiplied by $3/t$. Here we analyse the effects of this **intriguing difference** and its connection to acceleration.

GRADIENT AMPLIFICATION LEADS TO ACCELERATION. A **naïve way to speed up the convergence** of the GD-ODE $\dot{X}(t) = -\nabla f(X(t))$ is to consider

$$\dot{X}(t) = -t\nabla f(X(t)). \tag{29}$$

This can be seen by means of the Lyapunov function $\mathcal{E}(x,t) = t^2(f(x) - f(x^*)) + \|x - x^*\|^2$. Using convexity of $f(\cdot)$, we have

$$\dot{\mathcal{E}}(X(t),t) = -t^3\|\nabla f(X(t))\|^2 \leq 0, \tag{30}$$

13 See e.g. Theorem 5.2.1 in Øksendal, 2003, which gives sufficient conditions for (strong) existence and uniqueness.
14 Acceleration is not achieved for a viscosity of e.g. $2/t$.

and therefore the solution is s.t. $f(X(t)) - f(x^*) \leq \mathcal{O}(1/t^2)$. However, the Euler **discretization** of this ODE is the gradient-descent-like recursion $x_{k+1} = x_k - \eta k \nabla f(x_k)$ — which is *not* accelerated. Indeed, *gradient amplification* by a factor of $t$ is effectively changing the Lipschitz constant of the gradient field from $L$ to $kL$. Therefore, each step is going to yield a descent only if[15] $\eta \leq \frac{1}{kL}$. This iteration dependent learning rate effectively **cancels out gradient amplification**, which brings us back to the standard convergence rate $\mathcal{O}(1/k)$. It is thus natural to ask:

*"Is the mechanism of acceleration behind Nesterov's ODE related to a similar gradient amplification?"*

It is easy to show that $\{X_N(t), V_N(t)\}_{t\geq 0}$, the solution to Nesterov's SDE[16], is s.t. the infinitesimal update direction $V_N(t)$ of the position $X_N(t)$ can be written as

$$V_N(t) = -\int_0^t \frac{s^3}{t^3} \nabla f(X(s)) ds + \zeta_N(t), \tag{31}$$

where $\zeta_N(t)$ is a random vector with $\mathbb{E}[\zeta_N(t)] = 0$ and

$$\mathrm{Var}[\zeta_N(t)] = \frac{1}{7} t \sigma \sigma^\top. \tag{32}$$

In contrast, the solution $\{X_{\mathrm{m2}}(t), V_{\mathrm{m2}}(t)\})_{t\geq 0}$ of MG-SDE with quadratic forgetting satisfies

$$V_{\mathrm{m2}}(t) = -\int_0^t \frac{3s^2}{t^3} \nabla f(X(s)) ds + \zeta_{\mathrm{m2}}(t), \tag{33}$$

where $\zeta_{\mathrm{m2}}(t)$ is a random vector with $\mathbb{E}[\zeta_{\mathrm{m2}}(t)] = 0$ but

$$\mathrm{Var}[\zeta_{\mathrm{m2}}(t)] = \frac{9}{5t} \sigma \sigma^\top. \tag{34}$$

The reader might already have spotted an important difference in the noise covariances. To make our connection to gradient amplification even clearer, we consider the simpler setting of constant gradients: in this case,

15 See e.g. Bottou et al., 2018.

16 Nesterov's SDE is defined, as for MG-SDE by augmenting the phase space representation with a volatility term. The resulting system is then : $dX(t) = V(t)dt$; $dV(t) = -3/tV(t)dt - \sigma(X(t))dB(t)$.

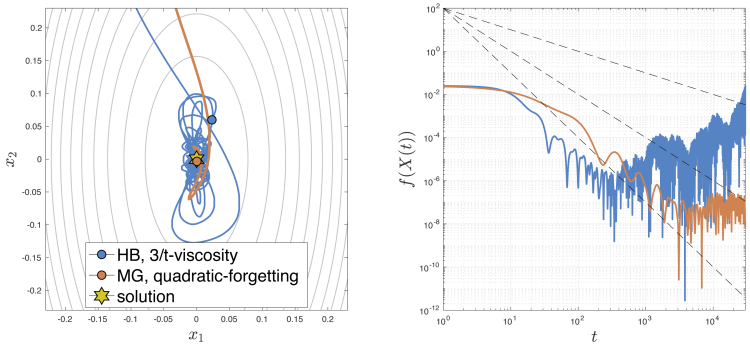


**Figure 8:** *HB-SDE with $\alpha(t) = 3/t$ (i.e. Nesterov's SDE) compared to MG-SDE with quadratic forgetting. Setting as in Su et al., 2016: $f(x) = 2 \times 10^{-2} x_1^2 + 5 \times 10^{-3} x_2^2$ starting from $X_0 = (1,1)$ and $\dot{X}(0) = (0,0)$. Both systems are exposed to the same noise volatility. Simulation using the Milstein scheme (Mil'shtejn, 1975) with stepsize $10^{-3}$.*

we have $V_N(t) = -\frac{1}{4} t \nabla f(X(t)) + \zeta_N(t)$, $V_{\mathfrak{m}2}(t) = -\nabla f(X(t)) + \zeta_{\mathfrak{m}2}(t)$. That is, stochastic algorithms with increasing momentum (i.e. decreasing viscosity, like the Nesterov's SDE) **are systematically amplifying the gradients over time**. Yet, at the same time they also **linearly amplify the noise variance**. This argument can easily be extended to the non-constant gradient case by noticing that $\mathbb{E}[V_{\mathfrak{m}2}(t)]$ is a weighted average of gradients where the weights integrate to 1 for all $t \geq 0$ (Lemma 2.3.1) . In contrast, in $\mathbb{E}[V_N(t)]$ these weights integrate to $t/4$. This behaviour is illustrated in Fig. 8: While the Nesterov's SDE is faster compared to MG-SDE with $\mathfrak{m}(t) = t^3$ at the beginning, it quickly **becomes unstable** because of the increasing noise in velocity and hence position. This gives multiple insights on the behavior of Nesterov's accelerated method for convex functions, both in for deterministic and the stochastic gradients:

1. **Deterministic gradients get linearly amplified overtime**, which counteracts the slow-down induced by the vanishing gradient problem around the solution. Interestingly Eq. (31) reveals that this amplification is *not* performed directly on the local gradient but on past history, with *cubic* forgetting. It is this feature that makes discretization stable compared to the naïve $\dot{X} = -t \nabla f(X(t))$.

2. **Stochasticity corrupts the gradient amplification** by an increasing noise variance (see Eq. (31)), which makes Nesterov's SDE unstable. This finding is in line with Allen-Zhu, 2017.

Furthermore, our analysis also gives an intuition as to why a constant momentum cannot in general yield acceleration in the non-strongly convex setting. Indeed, we saw already that HB-ODE-INT does not allow such persistent amplification, but at most a constant amplification inversely proportional to the (constant) viscosity.

TIME WARPING OF LINEAR MEMORY. We show here that Nesterov's path has a strong link to — surprisingly — linear forgetting. Consider speeding-up the **linear forgetting** ODE

$$\ddot{X}(t) + \frac{2}{t}\dot{X}(t) + \frac{2}{t}\nabla f(X(t)), \tag{35}$$

by introducing the **time change** (see also App. A)

$$\tau(t) = t^2/8, \tag{36}$$

and let $Y(t) = X(\tau(t))$ be the *accelerated* solution to linear forgetting. By the chain rule, we have

$$\dot{Y}(t) = \dot{\tau}(t)\dot{X}(\tau(t)) \quad \Longrightarrow \quad \ddot{Y}(t) = \ddot{\tau}(t)\dot{X}(\tau(t)) + \dot{\tau}(t)^2\ddot{X}(\tau(t)). \tag{37}$$

It can easily be verified that we recover $\ddot{Y}(t) + \frac{3}{t}\dot{Y}(t) + \nabla f(Y(t))$. However, in the stochastic setting, the behaviour is still quite different: as predicted by the theory, in Fig. 9 we see that — when gradients are large — the **trajectory of the two sample paths are almost identical** (yet, notice that **Nesterov moves faster**); however, as we approach the solution, **Nesterov diverges** while linear forgetting stably proceeds towards the minimizer along the Nesterov's ODE path, but at a different speed, until convergence to a neighborhood of the solution, as proved in Section 2.3.5. In the appendix of Orvieto et al., 2019 we prove that there are no other time changes which can cast MG-ODE into HB-ODE. This yields the following interesting conclusion: the *only* way to translate a memory system into a momentum method is by using a time change $\tau(t) = \mathcal{O}(t^2)$.

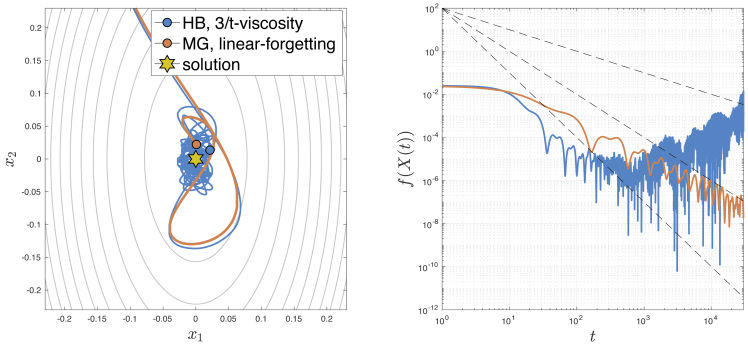


**Figure 9:** *Nesterov's ODE compared to MG-SDE with linear forgetting (i.e.* $\dot{\mathfrak{m}}(t)/\mathfrak{m}(t) = 2/t$*). Same settings as Fig. 8.*

### 2.3.4 *Convergence Analysis in Continuous-time*

In this section we first analyze convergence of MG-SDE under different memory functions. Next, **we use the Lyapunov analysis carried out in continuous-time** to derive an iterative discrete-time method that implements polynomial forgetting and has strong last-iterate convergence guarantees. We state a few assumptions:

**(Ho′)** $f \in \mathcal{C}_b^3(\mathbb{R}^d, \mathbb{R})$, $\sigma_*^2 := \sup_x \|\sigma(x)\sigma(x)^T\|_s < \infty$,

where $\mathcal{C}_b^3(\mathbb{R}^d, \mathbb{R})$ denotes the class of three times continuously differentiable functions with bounded derivates up to the third order, and $\|\cdot\|_s$ denotes the spectral norm. The definition of $\sigma_*^2$ nicely decouples the measure of noise magnitude to the problem dimension $d$ (which will then, of course, appear explicitly in all our rates).

**(H1′)** $f : \mathbb{R}^d \to \mathbb{R}$ is $L$-smooth and there exist $x^* \in \mathbb{R}^d$ s.t. for all $x \in \mathbb{R}^d$, $\langle \nabla f(x), x - x^* \rangle \geq \tau(f(x) - f(x^*))$.

The last condition is known as weak-quasi-convexity and generalizes convexity (convex functions are weakly-quasi-convex with constant 1 (Hardt et al., 2018)). For background on smoothness and convexity, the reader can check Appendix A.

We start with a general result about the convergence of the memory SDE (see Chapter 2) for arbitrary memory $\mathfrak{m}(\cdot)$ in the stochastic setting. We will then specialize this result to **polynomial forgetting**.

**Lemma 2.3.2** (Continuous-time Master Inequality)**.** *Assume **(Ho')** and **(H1')** hold. Let $\{X(t), V(t)\}_{t\geq 0}$ be a solution to MG-SDE with memory $\mathfrak{m}(\cdot)$, define $\lambda(t) := -\mathfrak{m}(t)\int \frac{1}{\mathfrak{m}(t)}dt$ (where $\int$ denotes the antiderivative[17]) and $r(t) := \frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\lambda(t)^2$. If $\mathfrak{m}(\cdot)$ is such that $\dot{r}(t) \leq \tau\lambda(t)\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}$, then for any $t > 0$*

$$\mathbb{E}[f(X(t)) - f(x^*)] \leq \frac{r(0)(f(x_0) - f(x^*)) + \frac{1}{2}\|x_0 - x^*\|^2}{r(t)} + \frac{d\sigma_*^2}{2}\frac{\int_0^t \lambda(s)^2 \left(\frac{\dot{\mathfrak{m}}}{\mathfrak{m}}(s)\right)^2 ds}{r(t)}.$$

*Proof.* The proof uses the following Lyapunov function, inspired from Su et al., 2016:

$$\mathcal{E}(x, v, t) = r(t)(f(x) - f(x^*)) + \frac{1}{2}\|x - x^* + \lambda(t)v\|^2.$$

Details can be found in in Appendix B.1. □

Using this lemma, we can prove a last-iterate convergence result for the memory SDE. The proof showcases **simplicity and beauty when working with SDEs**.

**Theorem 2.3.1.** *Under the conditions of Lemma 2.3.2, let $\mathfrak{m}(t) = t^p$, with $p \geq 1 + \frac{1}{\tau}$. Then, for any $t > 0$,*

$$\mathbb{E}[f(X(t)) - f(x^*)] \leq \underbrace{\frac{(p-1)^2\|x_0 - x^*\|^2}{2pt}}_{\text{rate of convergence to suboptimal sol.}} + \underbrace{\frac{p\, d\, \sigma_*^2}{2}}_{\text{suboptimality}}.$$

*Proof.* Since $p \neq 1$, we have $\lambda(t) = -\mathfrak{m}(t)\int \frac{1}{\mathfrak{m}(t)}dt = -t^p \int \frac{1}{t^p}dt = -t^p\left(\frac{t^{1-p}}{1-p} - C\right)$. Let us choose $C = 0$, then $\lambda(t) = \frac{t}{p-1}$ and $r(t) = \frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\lambda(t)^2 = \frac{p}{t}\frac{t^2}{(p-1)^2} = \frac{pt}{(p-1)^2}$. Thanks to the Lemma, we get a rate if $\dot{r}(t) \leq \tau\lambda(t)\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}$; that is, $\frac{p}{(p-1)^2} \leq \tau\frac{t}{p-1}\frac{p}{t}$, which is true if and only if $p \geq 1 + \frac{1}{\tau}$. Plugging in $h, \mathfrak{m}$ and $r$, we get the desired rate. □

17 Equivalently, $\lambda$ is such that $\dot{\lambda}(t) = \frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\lambda(t) - 1$.

### 2.3.5 *Convergence Analysis in Discrete-time*

Reasoning in continuous-time is much easier compared to discrete-time. Following insights from the continuous-time analysis, we derive in this subsection a discrete version of MG-SDE and a corresponding rate.
We start from the algorithm below, a generalization of HB that builds a sequence of iterates $\{x_k\}_{k\in\mathbb{N}}$ as well as moments $\{m_k\}_{k\in\mathbb{N}}$ starting from $m_0 = 0$ using the recursion

$$m_{k+1} = \beta_k(x_k - x_{k-1}) - \delta_k \eta \nabla f_{i_k}(x_k) \tag{38}$$

$$x_{k+1} = x_k + m_{k+1}, \tag{39}$$

where $i_k \in \{1, \dots, n\}$ is the index of a random data point sampled at iteration $k$, $\beta_k$ is an iteration-dependent positive momentum parameter and $\delta_k$ is a positive iteration-dependent discount on the learning rate $\eta$. For each $x \in \mathbb{R}^d$, the sample loss $f_{i_k}(x)$ is a random variable with mean zero and finite covariance matrix, which we call $\Sigma(x)$.
We list below two important assumptions for our convergence guarantee

**(Ho)** $\varsigma_*^2 := \sup_{x\in\mathbb{R}^d} \|\Sigma(x)\|_s < \infty$.

**(H1)** $f : \mathbb{R}^d \to \mathbb{R}$ is convex and $L$-smooth.

Note that, compared to the continuous-time setting, we have to assume convexity and unfortunately **cannot work with the weaker weak-quasi-convexity assumption**. We start with a rather abstract result inspired by Ghadimi et al., 2015, which we will then use for algorithm design.

**Lemma 2.3.3** (Discrete-time Master Inequality)**.** *Assume **(H1)** and **(Ho)** hold. Let $\{\lambda_k\}_{k\in\mathbb{N}}$ be any sequence such that $\lambda_k \leq k$ for all $k$ and define $r_k = \eta(\lambda_k + 1)$. If $\beta_k = \frac{\lambda_k}{\lambda_{k+1}+1}$, $\delta_k = \frac{1}{\lambda_{k+1}+1}$ and $\eta \leq 1/L$. Then we have for all iterates $\{x_k\}_{k\in\mathbb{N}}$ given by Eq. (38) and (39) that*

$$\mathbb{E}[f(x_k) - f(x^*)] \leq \frac{2r_0(f(x_0) - f(x^*)) + \frac{1}{2}\|x_0 - x^*\|^2}{r_k} + \frac{d\eta^2\varsigma_*^2(k+1)}{2r_k}.$$

*Proof.* The proof is based on the following Lyapunov function **inspired by the continuous-time** setting in Lemma 2.3.1.

$$\mathcal{E}_k = r_k(f(x_k) - f(x^*)) + \frac{1}{2}\|x_{k+1} - x^* + \lambda_k m_{k+1}\|^2. \tag{40}$$

Details can be found in Appendix B.1. □

We now apply the lemma above to get an algorithm and a convergence rate. In particular, we want to implement **polynomial memory** of past gradients and still get the rate found in Theorem 2.3.1.

**Theorem 2.3.2.** *Assume **(H1)** and **(Hod)** hold. Consider the following method*

$$x_{k+1} = x_k + \frac{k}{k+p}(x_k - x_{k-1}) - \frac{p}{k+p}\eta \nabla f_{i_k}(x_k),$$

*with $p \geq 2$ and $\eta \leq \frac{p-1}{pL}$. We have*

$$\mathbb{E}[f(x_k) - f(x^*)] \leq \underbrace{\frac{2\eta(p-1)(f(x_0) - f(x^*)) + \frac{1}{2}\|x_0 - x^*\|^2}{\eta(k+p-1)}}_{\text{rate of convergence to suboptimal sol.}} + \underbrace{\frac{d(p-1)\eta\zeta_*^2}{2}}_{\text{suboptimality}}.$$

*Moreover, the resulting update direction can be written as*

$$x_{k+1} - x_k = -\eta \sum_{i=0}^{k} w(i,k) \nabla f(x_i),$$

*with $\sum_{i=0}^{k} w(i,k) = 1$ and $w(\cdot, k) : \{0, \dots, k\} \to \mathbb{R}$ behaving like a $(p-1)$-th order polynomial, that is $w(i,k) \sim i^{p-1}$, for all $k$.*

We present the proof to showcase exactly the point where we leverage intuition from continuous-time. Note that, compared to the SDE setting, here **reasoning is more tedious** (1 page of proof vs. 5 lines): sums are often harder to work with, compared to integrals.

*Proof.* In the context of Lemma 2.3.3, **pick the continuous-time inspired** function $\lambda_k = \frac{k}{p-1} \leq k$ (see $\lambda(t)$ in Prop. 2.3.1). If $r_k = \eta(\lambda_k + 1) = \eta\frac{k+p-1}{p-1}$, $\beta_k = \frac{\lambda_k}{\lambda_{k+1}+1} = \frac{k}{k+p}$, $\delta_k = \frac{1}{\lambda_{k+1}+1} = \frac{p-1}{k+p}$, the iterates defined by

$$x_{k+1} = x_k + \frac{k}{k+p}(x_k - x_{k-1}) - \frac{p-1}{k+p}\eta\nabla f_{i_k}(x_k), \tag{41}$$

are such that, under $\eta \leq 1/L$ (see fundamental lemma)

$$\begin{aligned}
&\mathbb{E}[f(x_k) - f(x^*)] \\
&\leq \frac{2r_0(f(x_0) - f(x^*)) + \frac{1}{2}\|x_0 - x^*\|^2}{r_k} + \frac{d\eta^2\varsigma_*^2(k+1)}{2r_k} \\
&= \frac{2\eta(p-1)(f(x_0) - f(x^*)) + \frac{p-1}{2}\|x_0 - x^*\|^2}{\eta(k+p-1)} + \frac{d(p-1)\eta^2\varsigma_*^2(k+1)}{2\eta(k+p-1)} \\
&\leq \frac{2\eta(p-1)(f(x_0) - f(x^*)) + \frac{p-1}{2}\|x_0 - x^*\|^2}{\eta(k+p-1)} + \frac{d(p-1)\eta\varsigma_*^2}{2}
\end{aligned} \tag{42}$$

where we also used the fact that $p \geq 2$. Now we have to get back to the algorithm defined in our statement, to do this we simply need $\eta \leftarrow \frac{p}{p-1}\eta$ (hence the stability condition $\eta \leq \frac{p-1}{pL}$). The rate becomes

$$\mathbb{E}[f(x_k) - f(x^*)] \leq \frac{2\eta(p-1)^2(f(x_0) - f(x^*)) + \frac{p^2}{2}\|x_0 - x^*\|^2}{\eta p(k+p-1)} + \frac{pd\eta\varsigma_*^2}{2}. \tag{43}$$

To reveal the hidden averaging structure in the algorithm (after change of variable $\eta$), let us first expand the iterates:

$$\begin{aligned}
x_{k+1} - x_k &= -\eta\sum_{j=0}^{k-1}\left(\prod_{h=j+1}^{k}\frac{h}{h+p}\right)\frac{p}{j+p}\nabla f(x_j) - \eta\frac{p}{k+p}\nabla f(x_k) \\
&= -\eta\sum_{j=0}^{k} w(j,k)\nabla f(x_j),
\end{aligned} \tag{44}$$

therefore, using some simple formulas [18] from number theory, its easy to check that the weights behave polynomially and sum to 1:

$$\begin{aligned}
\sum_{j=0}^{k} w(j,k) &= \sum_{j=0}^{k-1} \left( \prod_{h=j+1}^{k} \frac{h}{h+p} \right) \frac{p}{j+p} + \frac{p}{k+p} \\
&= \sum_{j=0}^{k-1} \frac{(j+1)(j+2)\cdots\cancel{(j+p)}}{(k+1)(k+2)\cdots(k+p)} \frac{p}{\cancel{(j+p)}} + \frac{p}{k+p} \\
&= \sum_{j=0}^{k-1} \frac{(j+1)(j+2)\cdots\cancel{(j+p)}}{(k+1)(k+2)\cdots(k+p)} \frac{p}{\cancel{(j+p)}} + \frac{p}{k+p} \\
&= \frac{p}{(k+1)(k+2)\cdots(k+p)} \sum_{j=0}^{k} (j+1)(j+2)\cdots(j+p-1),
\end{aligned} \tag{45}$$

From the last formula, we see that indeed the weights behave like a $(p-1)$-order polynomial. To see that the weights some to 1, notice that

$$\sum_{j=0}^{k} (j+1)(j+2)\cdots(j+p-1) = \frac{(k+1)(k+2)\cdots(k+p)}{p}. \tag{46}$$

□

18 The reader can check the formula in Wolphram Alpha®: `http://tinyurl.com/y3quchcd`

## 2.4 SHADOWING OF DISCRETE TRAJECTORIES

*The precise definition of "hyperbolic" is a little complicated, and so we defer it until next semester, but roughly speaking it means that the dynamical system expands in some directions and contracts in others.*

– Will J. Merry

As evident from Section 2.3, working in continuous time provides **profound insights and simplifies reasoning** when analyzing gradient-based methods. Therefore, it comes as no surprise that the relationship between continuous-time models and their discrete counterparts has become a focal point of interest in the contemporary optimization literature (Li et al., 2017; Shi et al., 2019; Muehlebach and Jordan, 2020; Muehlebach and Jordan, 2021; Zhang et al., 2021c). Specifically, much research is focused on understanding the error resulting from the discretization of a gradient-based ODE into an optimizer. The conceptual challenge is that without assuming heavy regularity (e.g. the system is Hamiltonian, as in Benettin and Giorgilli, 1994; Hairer et al., 2003), the **approximation error** of any (even high-order) numerical integrator **grows exponentially**[19] as a function of the integration interval (Chow and Van Vleck, 1994; Hairer et al., 2003). As a result, convergence rates derived for ODEs **can not be straightforwardly translated to algorithmic guarantees**. While continuous-time is helpful to guide reasoning, obtaining convergence guarantees for the discrete case often requires analyzing a **separate discrete-time Lyapunov function**, which at times cannot be easily recovered from the one used for the continuous-time analysis (we were lucky in Section 2.3, see instead Shi et al., 2021; Shi et al., 2019).

**Question:** Under which conditions optimization algorithms are well approximated by their limiting continuous-time models?

19 The error between the numerical approximation and the actual trajectory with the same initial conditions is, for a $p$-th order method, $e^{Ct}h^p$ at time $t$ with $C \gg 0$.

**Answer (Orvieto and Lucchi, 2019b):** Under strong convexity or near well-conditioned saddles, models of gradient descent (with momentum) are guaranteed to be faithful to algorithms: trajectories are uniformly close — one *shadows* the other. This finding empirically holds true also on simple neural networks.

In this work, we study conditions on the loss function under which the flow of an **ODE model is shadowed** by (i.e. is uniformly close[20] to) the iterates of an optimization algorithm. The key difference with previous work, which makes our analysis possible, is that we allow the algorithm — i.e. the *shadow*— to be initialized at a **slightly perturbed point** compared to the ODE (see Fig. 14 for an illustration). We rely on tools from numerical analysis (Hairer et al., 2003) as well as concepts from dynamical systems (Brin and Stuck, 2002), where solutions to ODEs and iterations of algorithm are viewed as the same object, namely maps in a topological space (Brin and Stuck, 2002). Specifically, **our analysis builds on the theory of hyperbolic sets**, which grew out of seminal works ny Anosov (Anosov, 1967) and Smale (Smale, 1967) in the 1960's and plays a fundamental role in several branches of the area of dynamical systems but has not yet been seen to have a relationship with optimization for machine learning.

To the best of our knowledge, our work is the **first to focus on a** (Lyapunov function independent) **systematic and quantitative comparison of ODEs and algorithms** for optimization. Also, we believe the tools we describe in this work can be used to advance our understanding of related machine learning problems, perhaps to better characterize the attractors of neural ordinary differential equations (Chen et al., 2018b).

### 2.4.1 *Background on Pseudo-orbits and Shadowing*

This section provides an overview of fundamental concepts in the theory of dynamical systems, which we will use heavily in the rest of the section. Many of the definitions and results used here are also included in the background material (App A).

20 Formally defined in Sec. 2.4.1.

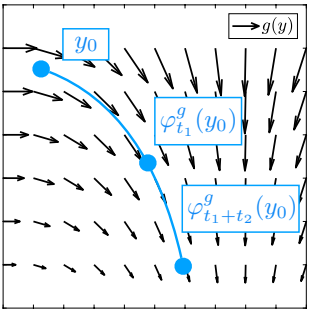


**Figure 10:** *Flow of the vector field $g$.*

#### 2.4.1.1 *Differential equations and flows*

Consider the autonomous differential equation $\dot{y} = g(y)$. Every $y$ represents a point in $\mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^n$ is a vector field which, at any point, prescribes the velocity of the solution $y$ that passes through that point. Formally, the curve $y : \mathbb{R} \to \mathbb{R}^n$ is a solution passing through $y_0$ at time 0 if $\dot{y}(t) = g(y(t))$ for $t \in \mathbb{R}$ and $y(0) = y_0$. We call this the solution to the initial value problem (IVP) associated with $g$ (starting at $y_0$). The following results can be found in Perko, 2013; Khalil, 2002.

**Theorem 2.4.1.** *Assume $g$ is Lipschitz continuous and $\mathcal{C}^k$. The IVP has a unique $\mathcal{C}^{k+1}$ solution in $\mathbb{R}$.*

This fundamental theorem tells us that, if we integrate for $t$ time units from position $y_0$, the final position $y(t)$ is uniquely determined. Therefore, we can define the family $\{\varphi^g_t\}_{t\in\mathbb{R}}$ of maps — the **flow** of $g$ — such that $\varphi^g_t(y_0)$ is the solution at time $t$ of the IVP. Intuitively, we can think of $\varphi^g_t(y_0)$ as determining the location of a particle starting at $y_0$ and moving via the velocity field $g$ for $t$ seconds. Since the vector field is static, we can move along the solution (in both directions) by iteratively applying this map (or its inverse). This is formalized below.

**Proposition 2.4.1.** *Assume $g$ is Lipschitz continuous and $\mathcal{C}^k$. For any $t \in \mathbb{R}$, $\varphi^g_t \in \mathcal{C}^{k+1}$ and, for any $t_1, t_2 \in \mathbb{R}$, $\varphi^g_{t_1+t_2} = \varphi^g_{t_1} \circ \varphi^g_{t_2}$. In particular, $\varphi^g_t$ is a diffeomorphism[21] with inverse $\varphi^g_{-t}$.*

21 A $\mathcal{C}^1$ map with well-defined and $\mathcal{C}^1$ inverse.

In a way, the flow allows us to **exactly discretize** the trajectory of an ODE. Indeed, let us fix a stepsize $h > 0$; the associated **time-$h$ map** $\varphi_h^g$ integrates the ODE $\dot{y} = g(y)$ for $h$ seconds starting from some $y_0$, and we can apply this map recursively to compute a sampled ODE solution. In this section, we study how flows can approximate the iterations of some optimization algorithm using the language of dynamical systems and the concept of shadowing.

#### 2.4.1.2 *Dynamical systems and shadowing*

A dynamical system on $\mathbb{R}^n$ is a map $\Psi : \mathbb{R}^n \to \mathbb{R}^n$. For $p \in \mathbb{N}$, we define $\Psi^p = \Psi \circ \cdots \circ \Psi$ ($p$ times). If $\Psi$ is invertible, then $\Psi^{-p} = \Psi^{-1} \circ \cdots \circ \Psi^{-1}$ ($p$ times). Since $\Psi^{p+m} = \Psi^p \circ \Psi^m$, the powers of $\Psi$ form a group if $\Psi$ is invertible, and a semigroup otherwise. We proceed with more definitions.

**Definition 2.4.1.** *A sequence $(x_k)_{k=0}^{\infty}$ is a (positive) **orbit** of $\Psi$ if, for all $k \in \mathbb{N}$, $x_{k+1} = \Psi(x_k)$.*

For the rest of this subsection, the reader may think of $\Psi$ as an optimization algorithm (such as GD, which maps $x$ to $x - \eta \nabla f(x)$ ) and of the orbit $(x_k)_{k=0}^{\infty}$ as its iterates. Also, the reader may think of $(y_k)_{k=0}^{\infty}$ as the sequence of points derived from the iterative application of $\varphi_h^g$, which is itself a dynamical system, from some $y_0$. The latter sequence represents our ODE approximation of the algorithm $\Psi$. Our goal in this subsection is to understand when a sequence $(y_k)_{k=0}^{\infty}$ is "close to" an orbit of $\Psi$. The first notion of such similarity is local.

**Definition 2.4.2.** *The sequence $(y_k)_{k=0}^{\infty}$ is a $\delta-$**pseudo-orbit** of $\Psi$ if, for all $k \in \mathbb{N}$, $\|y_{k+1} - \Psi(y_k)\| \leq \delta$.*

If $(y_k)_{k=0}^{\infty}$ is locally similar to an orbit of $\Psi$ (i.e. it is a pseudo-orbit of $\Psi$), then we may hope that such similarity extends globally. This is captured by the concept of shadowing.

**Definition 2.4.3.** *A pseudo-orbit $(y_k)_{k=0}^{\infty}$ of $\Psi$ is $\epsilon-$**shadowed** if there exists an orbit $(x_k)_{k=0}^{\infty}$ of $\Psi$ such that, for all $k \in \mathbb{N}$, $\|x_k - y_k\| \leq \epsilon$.*

It is crucial to notice that, as depicted in the figure above, we allow the shadowing orbit (a.k.a. the shadow) to start at a *perturbed point* $x_0 \neq y_0$. A natural question is the following: *which properties must $\Psi$ have such that*

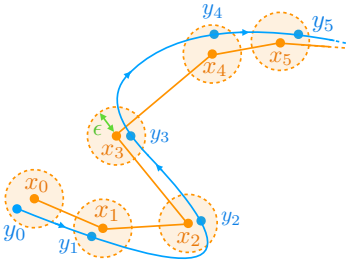


**Figure 11:** *Orbit of the map $\varphi_h^g$ (blue) is shadowed by the algorithm $\Psi$ (orange).*

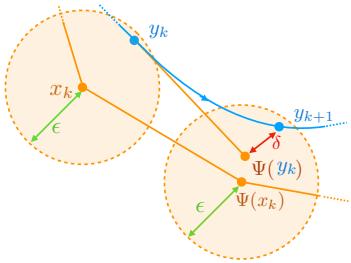


**Figure 12:** *Illustration of the shadowing proof in Theorem 2.4.2.*

*a general pseudo-orbit is shadowed?* A lot of research has been carried out in the last decades on this topic (see e.g. Palmer, 2013; Lanford, 1985 for a comprehensive survey).

SHADOWING FOR CONTRACTIVE/EXPANDING MAPS.
A straight-forward sufficient condition is related to contraction. $\Psi$ is said to be *uniformly contracting* if there exists $\rho < 1$ (*contraction factor*) such that for all $x_1, x_2 \in \mathbb{R}^n$, $\|\Psi(x_1) - \Psi(x_2)\| \leq \rho\|x_1 - x_2\|$. The next result can be found in Hayes and Jackson, 2005.

**Theorem 2.4.2.** *(Contraction map shadowing theorem) Assume $\Psi$ is uniformly contracting. For every $\epsilon > 0$ there exists $\delta > 0$ such that every $\delta-$pseudo-orbit $(y_k)_{k=0}^{\infty}$ of $\Psi$ is $\epsilon-$shadowed by the orbit $(x_k)_{k=0}^{\infty}$ of $\Psi$ starting at $x_0 = y_0$, that is $x_k := \Psi^k(x_0)$. Moreover, $\delta \leq (1-\rho)\epsilon$.*

The idea behind this result is simple: **since all distances are contracted, errors that are made along the pseudo-orbit vanish asymptotically**. For instructive purposes, we report the full proof.

*Proof.* We proceed by induction: the proposition is trivially true at $k = 0$, since $\|x_0 - y_0\| \leq \epsilon$; next, we assume the proposition holds at $k \in \mathbb{N}$ and we show validity for $k+1$. We have

$$\begin{aligned} \|x_{k+1} - y_{k+1}\| &\overset{\text{subadditivity}}{\leq} \|\Psi(x_k) - \Psi(y_k)\| + \|\Psi(y_k) - y_{k+1}\| \\ &\overset{\delta\text{-pseudo-orbit}}{\leq} \|\Psi(x_k) - \Psi(y_k)\| + \delta \\ &\overset{\text{contraction}}{\leq} \rho\|x_k - y_k\| + \delta \\ &\overset{\text{induction}}{\leq} \rho\epsilon + \delta. \end{aligned} \tag{47}$$

Finally, since $\delta \leq \epsilon(1-\rho)$, $\rho\epsilon + \delta = \epsilon$. □

Next, assume $\Psi$ is invertible. If $\Psi$ is **uniformly expanding** (i.e. $\rho > 1$), then $\Psi^{-1}$ is uniformly contracting with contraction factor $1/\rho$ and we can apply the **same reasoning backwards**: consider the pseudo-orbit $\{y_0, y_1, \cdots, y_K\}$ up to iteration $K$, and set $x_K = y_K$ (before, we had $x_0 = y_0$); then, apply the same reasoning as above using the map $\Psi^{-1}$ and find the initial condition $x_0 = \Psi^{-K}(y_K)$. In App. B.2.1.2 we show that this reasoning holds in the limit, i.e. that $\Psi^{-K}(y_K)$ converges to a suitable $x_0$ to construct a shadowing orbit under $\delta \leq (1 - 1/\rho)\epsilon$ (precise statement in Thm. B.2.3).

SHADOWING IN HYPERBOLIC SETS. In general, for machine learning problems, the algorithm map $\Psi$ can be a combination of the two cases above: an example is the pathological contraction-expansion behavior of Gradient Descent around a saddle point (Du et al., 2017). To illustrate the shadowing properties of such systems, we shall start by taking $\Psi$ to be linear[22], that is $\Psi(x) = Ax$ for some $A \in \mathbb{R}^{n\times n}$. $\Psi$ is called **linear hyperbolic** if its spectrum $\sigma(A)$ does not intersect the unit circle $\mathbb{S}^1$. If this is the case, we call $\sigma_s(A)$ the part of $\sigma(A)$ inside $\mathbb{S}^1$ and $\sigma_u(A)$ the part of

22 This case is restrictive, yet it includes the important cases of the dynamics of GD and HB when $f$ is a quadratic (which is the case in, for instance, least squares regression).

$\sigma(A)$ outside $\mathbb{S}^1$. The spectral decomposition theorem (see e.g. Corollary 4.56 in Irwin, 2001) ensures that $\mathbb{R}^n$ decomposes into two $\Psi$-invariant closed subspaces (a.k.a the stable and unstable manifolds) in direct sum : $\mathbb{R}^n = E_s \oplus E_u$. We call $\Psi_s$ and $\Psi_u$ the restrictions of $\Psi$ to these subspaces and $A_s$ and $A_u$ the corresponding matrix representations; the theorem also ensures that $\sigma(A_s) = \sigma_s(A)$ and $\sigma(A_u) = \sigma_u(A)$. Moreover, using standard results in spectral theory (see e.g. App. 4 in Irwin, 2001) we can equip $E_s$ and $E_u$ with norms equivalent to the standard Euclidean norm $\|\cdot\|$ such that, w.r.t. the new norms, $\Psi_u$ is uniformly expanding and $\Psi_s$ uniformly contracting. If $A$ is symmetric, then it is diagonalizable and the norms above can be taken to be Euclidean. To wrap up this paragraph — we can think of a linear hyperbolic system as a map that allows us to decouple its stable and unstable components *consistently* across $\mathbb{R}^n$. Therefore, a shadowing result directly follows from a combination of Thm. 2.4.2 and B.2.3 (Hayes and Jackson, 2005).

An important further question is — whether the result above for linear maps can be generalized. From the classic theory of nonlinear systems (Khalil, 2002; Araújo and Viana, 2009), we know that in a neighborhood of an **hyperbolic point** p for $\Psi$, that is $D\Psi(p)$ has no eigenvalues on $\mathbb{S}^1$, $\Psi$ behaves like[23] a linear system. Similarly, in the analysis of optimization algorithms, it is often used that, near a saddle point, an Hessian-smooth function can be approximated by a quadratic (Levy, 2016; Daneshmand et al., 2018; Ge et al., 2015). Hence, it should not be surprising that pseudo-orbits of $\Psi$ are shadowed in a neighborhood of an hyperbolic point, if such set is $\Psi$-invariant (Lanford, 1985): this happens for instance if $p$ is a stable equilibrium point (see Khalil, 2002).

Starting from the reasoning above, the celebrated *shadowing theorem*, which has its foundations in the work of Ansosov (Anosov, 1967) and was originally proved in Bowen, 1975, provides the strongest known result of this line of research: near an **hyperbolic set** of $\Psi$, pseudo-orbits are shadowed. Informally, $\Lambda \subset \mathbb{R}^n$ is called hyperbolic if it is $\Psi$-invariant and has clearly marked local directions of expansion and contraction which make $\Psi$ behave similarly to a linear hyperbolic system.We provide the precise definition of hyperbolic set and the statement of the shadowing theorem in App. B.2.1.1.

23 For a precise description of this similarity, we invite the reader to read on *topological conjugacy* in (Araújo and Viana, 2009).

Unfortunately, **despite the ubiquity of hyperbolic sets, it is practically infeasible to establish this property analytically** (Aulbach and Colonius, 1996). Therefore, an important part of the literature (Coomes, Palmer, et al., 1995; Sauer and Yorke, 1991; Chow and Van Vleck, 1994; Van Vleck, 1995) is concerned with the numerical verification of the existence of a shadowing orbit *a posteriori*, i.e. given a particular pseudo-orbit.

### 2.4.2 *Shadowing Gradient Descent*

We assume some regularity on the objective function $f : \mathbb{R}^d \to \mathbb{R}$ which we seek to minimize.

**(H1)** $f \in \mathcal{C}^2(\mathbb{R}^d, \mathbb{R})$ is coercive[24], bounded from below and $L$-smooth ($\forall a \in \mathbb{R}^d$, $\|\nabla^2 f(a)\| \leq L$).

In this chapter, we study the well-known continuous-time model for GD, that is $\dot{y} = \nabla f(y)$ (GD-ODE). Under **(H1)**, Thm. 2.4.1 ensures that the solution to GD-ODE exists and is unique. We denote by $\varphi_h^{\mathrm{GD}}$ the corresponding time-$h$ map. We show that, under some additional assumptions, the orbit of $\varphi_h^{\mathrm{GD}}$ (which we indicate as $(y_k)_{k=0}^{\infty}$) is **shadowed by an orbit of the GD map** with learning rate $h$: $\Psi_h^{\mathrm{GD}}$.

**Remark 2.4.1** (Bound on the ODE gradients)**.** *Under **(H1)** let $G_y = \{p : p = \|\nabla f(y(t))\|, t \geq 0\}$ be the set of gradient magnitudes experienced along the GD-ODE solution starting at any $y_0$. It is easy to prove, using an argument similar to Prop. 2.2 in Cabot et al., 2009, that coercivity implies $\sup G_y < \infty$. A similar argument holds for the iterates of Gradient Descent. Hence, for the rest of this chapter it is safe to assume that gradients are bounded: $\|\nabla f(y(t))\| \leq \ell$ for all $t \geq 0$. For instance, if $f$ is a quadratic centered at $x^*$, then we have $\ell = L\|y_0 - x^*\|$.*

The next result follows from the fact that GD implements the explicit Euler method on GD-ODE.

**Proposition 2.4.2.** *Assume **(H1)**. $(y_k)_{k=0}^{\infty}$ is a $\delta$-pseudo-orbit of $\Psi_h^{GD}$ with $\delta = \frac{\ell L}{2} h^2$: $\forall k \in \mathbb{N}$,*

$$\|y_{k+1} - \Psi_h^{GD}(y_k)\| \leq \delta.$$

24 $f(x) \to \infty$ as $\|x\| \to \infty$.

*Proof.* Thanks to Thm. 2.4.1, since the solution $y$ of GD-ODE is a $\mathcal{C}^2$ curve, we can write $y(kh+h) = \varphi_h^{\mathrm{GD}}(y(kh)) = y_{k+1}$ using Taylor's formula with Lagrange's Remainder in Banach spaces (see e.g. Thm 5.2. in Coleman, 2012) around time $t = kh$. Namely: $y(kh+h) = y(kh) - \dot{y}(kh)h + \mathcal{R}(2,h)$, where $\mathcal{R}(2,\cdot) : \mathbb{R}_{>0} \to \mathbb{R}^d$ is the approximation error as a function of $h$, which can be bounded as follows:

$$\begin{aligned}
\|\mathcal{R}(2,h)\| &\leq \frac{h^2}{2} \sup_{0\leq\lambda\leq 1} \|\ddot{y}(t+\lambda h)\| \\
&= \frac{h^2}{2} \sup_{0\leq\lambda\leq 1} \|\nabla^2 f(y(t+\lambda h))\nabla f(y(t+\lambda h))\| \\
&\leq \frac{h^2}{2} \sup_{0\leq\lambda\leq 1} \|\nabla^2 f(y(t+\lambda h))\| \sup_{0\leq\lambda\leq 1} \|\nabla f(y(t+\lambda h))\| \\
&\leq \frac{h^2}{2} L\ell. \qquad (48)
\end{aligned}$$

Since $y(kh) - \dot{y}(kh)h = \Psi_h^{\mathrm{GD}}(y_k)$, $\|y_{k+1} - \Psi_h^{\mathrm{GD}}(y_k)\| \leq \frac{\ell L}{2} h^2$. □

#### 2.4.2.1 *Shadowing under strong convexity*

As seen in Sec. 2.4.1.2, the last proposition provides the first step towards a shadowing result. We also discussed that if, in addition, $\Psi_h^{\mathrm{GD}}$ is a contraction, we directly have shadowing (Thm. 2.4.1). Therefore, we start with the following assumption that will be shown to **imply contraction**.

**(H2)** $f$ is $\mu$-**strongly convex**: for all $a \in \mathbb{R}^d$, $\|\nabla^2 f(a)\| \geq \mu$.

The next result follows from standard techniques in convex optimization (see e.g. Jung, 2017).

**Proposition 2.4.3.** *Assume* ***(H1)***, ***(H2)***. *If* $0 < h \leq \frac{1}{L}$, $\Psi_h^{GD}$ *is uniformly contracting with* $\rho = 1 - h\mu$.

We provide the proof in App. B.2.3 and sketch the idea using a quadratic form: let $f(x) = \langle x, Hx\rangle$ with $H$ symmetric s.t. $\mu I \preceq H \preceq LI$; if $h \leq \frac{1}{L}$ then $(1 - Lh) \leq \|I - hH\| \leq (1 - \mu h)$. Prop. 2.4.3 follows: $\|\Psi_h^{\mathrm{GD}}(x) - \Psi_h^{\mathrm{GD}}(y)\| = \|(I - hH)(x - y)\| \leq \|I - hH\|\|x - y\| \leq \rho\|x - y\|$.
The shadowing result for strongly convex functions is then a simple application of Thm. 2.4.2.

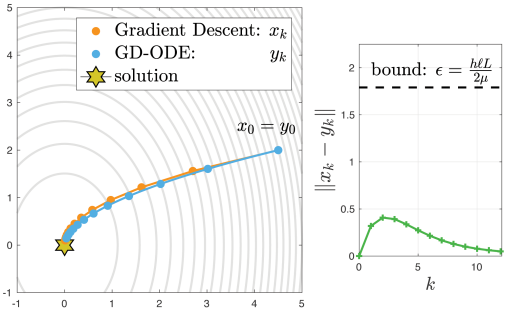


**Figure 13:** *Orbit of $\Psi_h^{GD}$, $\varphi_h^{GD}$ on a strongly convex quadratic. $h = 0.2$.*

**Theorem 2.4.3.** *Assume **(H1)**, **(H2)** and let $\epsilon$ be the desired accuracy. Fix $0 < h \leq \min\{\frac{2\mu\epsilon}{L\ell}, \frac{1}{L}\}$; the orbit $(y_k)_{k=0}^{\infty}$ of $\varphi_h^{GD}$ is $\epsilon$-shadowed by any orbit $(x_k)_{k=0}^{\infty}$ of $\Psi_h^{GD}$ with $x_0$ such that $\|x_0 - y_0\| \leq \epsilon$.*

*Proof.* From Thm. 2.4.2, we need $(y_k)_{k=0}^{\infty}$ to be a $\delta$-pseudo-orbit of $\Psi_h^{\mathrm{GD}}$ with $\delta \leq (1-\rho)\epsilon$. From Prop. 2.4.2 we know $\delta = \frac{\ell L}{2}h^2$, while from Prop. 2.4.3 we have $\rho \leq (1 - h\mu)$. Putting it all together, we get $\frac{\ell L}{2}h^2 \leq h\mu\epsilon$, which holds if and only if $h \leq \frac{2\mu\epsilon}{L\ell}$. □

Notice that we can formulate the theorem in a dual way: namely, **for every learning rate we can bound the ODE approximation error** (i.e. find the shadowing radius).

**Corollary 2.4.1.** *Assume **(H1)**, **(H2)**. If $0 < h \leq \frac{1}{L}$, $(y_k)_{k=0}^{\infty}$ is $\epsilon$-shadowed by any orbit $(x_k)_{k=0}^{\infty}$ of $\Psi_h^{GD}$ starting at $x_0$ with $\|x_0 - y_0\| \leq \epsilon$, with $\epsilon = \frac{h\ell L}{2\mu}$.*

This result ensures that if the objective is smooth and strongly convex, then GD-ODE is a theoretically sound approximation of GD. We validate this in Fig. 13 by integrating GD-ODE analytically.

SHARPNESS OF THE RESULT. First, we note that the bound for $\delta$ in Prop. 2.4.2 cannot be improved; indeed it coincides with the well-known **local truncation error** of Euler's method (Hairer and Wanner, 1996). Next, pick $f(x) = x^2/2$, $x_0 = 1$ and $h = 1/L = 1$. For $k \in \mathbb{N}$, gradients are smaller than 1 for both GD-ODE and GD, hence $\ell = L = \mu = 1$. Our formula for the **global shadowing radius** gives $\epsilon = hL\ell/(2\mu) = 0.5$, equal

to the local error $\delta = \ell L h^2/2$ — i.e. as tight the well-established local result. In fact, GD jumps to 0 in one iteration, while $y(t) = e^{-t}$; hence $y(1) - x_1 = 1/e \approx 0.37 < 0.5$. For smaller steps, like $h = 0.1$, our formula predicts $\epsilon = 0.05 = 10\delta$. In simulation, we have maximum deviation at $k = 10$, $0.02 = 4\delta$: only 2.5 times smaller than our prediction.

THE CONVEX CASE. If $f$ is convex but not strongly convex, GD is non-expanding and the error between $x_k$ and $y_k$ cannot be bounded by a constant[25] but grows slowly : in App. B.2.2.1 we show the error $\epsilon_k$ it is upper bounded by $\delta k = \ell L k h^2/2 = \mathcal{O}(kh^2)$.

EXTENSION TO STOCHASTIC GRADIENTS. We extend Thm. 2.4.3 to account for **perturbations**: let $\Psi_h^{\mathrm{SGD}}(x) = x - h\tilde{\nabla} f(x)$, where $\tilde{\nabla} f(x)$ is a stochastic gradient s.t. $\|\tilde{\nabla} f(x) - \nabla f(x)\| \leq R$. Then, for $\epsilon$ big enough, we can (proof in App. B.2.2.2) choose $h \leq \frac{2(\mu\epsilon - R)}{\ell L}$ so that the orbit of $\varphi_h^{\mathrm{GD}}$ (deterministic) is shadowed by the stochastic orbit of $\Psi_h^{\mathrm{SGD}}(x)$ starting from $x_0 = y_0$. So, if the $h$ is small enough, GD-ODE can be used to study SGD. This result is well known from stochastic approximation (Kushner and Yin, 2003).

### 2.4.2.2 *Towards non-convexity: behaviour around local maxima and saddles*

We first study the strong-concavity case and then combine it with strong-convexity to assess the shadowing properties of GD around a saddle.

STRONG-CONCAVITY. In this case, it follows from the same argument of Prop. 2.4.3 that GD is uniformly expanding with $\rho = 1 + \gamma h > 1$, with $-\gamma := \max(\sigma(H)) < 0$. As mentioned in the background section (see Thm. B.2.3 for the precise statement) this case is conceptually identical to strong-convexity once we **reverse the arrow of time** (so that expansions become contractions). We are allowed to make this step because, under **(H1)** and if $h \leq \frac{1}{L}$, $\Psi_h^{\mathrm{GD}}$ is a **diffeomorphism** (see e.g. Lee et al., 2016, Prop. 4.5). In particular, the backwards GD map $(\Psi_h^{\mathrm{GD}})^{-1}$ is contracting with factor $1/\rho$. Consider now the initial part of an orbit of GD-ODE such that the gradient norms are still bounded by $\ell$ and let $y_K = (\varphi_h^{\mathrm{GD}})^K(y_0)$

25 This in line with the requirement of hyperbolicity in the shadowing theory: a convex function might have zero curvature, hence the corresponding gradient system is going to have an eigenvalue on the unit circle.

be the last point of such orbit. It is easy to realize that $(y_k)_{k=0}^K$ is a pseudo-orbit, with reversed arrow of time, of $(\Psi_h^{\mathrm{GD}})^{-1}$. Hence, Thm. 2.4.3 directly ensures shadowing choosing $x_K = y_K$ and $x_k = (\Psi_h^{\mathrm{GD}})^{k-K}(y_K)$. Crucially — the initial condition of the shadow $(x_k)_{k=0}^K$ we found are **slightly**[26] **perturbed**: $x_0 = (\Psi_h^{\mathrm{GD}})^{-K}(y_K) \cong y_0$. Notice that, if we instead start GD from exactly $x_0 = y_0$, the iterates will diverge from the ODE trajectory, since every error made along the pseudo-orbit is amplified. We show this for the unstable direction of a saddle in Fig. 14.

QUADRATIC SADDLES. As discussed in Sec. 2.4.1, if the space can be **split into stable (contracting) and unstable (expanding) invariant subspaces** ($\mathbb{R}^d = E_s \oplus E_u$), then **every pseudo-orbit is shadowed**. This is a particular case of the shadowing theorem for hyperbolic sets (Bowen, 1975). In particular, we saw that if $\Psi_h^{\mathrm{GD}}$ is linear hyperbolic such splitting exists and $E_s$ and $E_u$ are the subspaces spanned by the stable and unstable eigenvalues, respectively. It is easy to realize that $\Psi_h^{\mathrm{GD}}$ is linear if the objective is a quadratic; indeed $f(x) = \langle x, Hx\rangle$ is such that $\Psi_h^{\mathrm{GD}}(x) = (I - hH)x$. It is essential to note that hyperbolicity requires $H$ to have no eigenvalue at 0 — i.e. that the saddle has only directions of strictly positive or strictly negative curvature. This splitting allows to study shadowing on $E_s$ and $E_u$ separately: for $E_s$ we can use the shadowing result for strong-convexity and for $E_u$ the shadowing result for strong-concavity, along with the computation of the initial condition for the shadow in these subspaces. We summarize this result in the next theorem, which we prove formally in App. B.2.2.3. To enhance understanding, we illustrate the procedure of construction of a shadow in Fig. 14.

**Proposition 2.4.4.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be quadratic centered at $x^*$ with Hessian $H$ with no eigenvalues in the interval $(-\gamma, \mu)$, for some $\mu, \gamma > 0$. Assume the orbit $(y_k)_{k=0}^\infty$ of $\varphi_h^{GD}$ is s.t. **(H1)** holds up to iteration $K$. Let $\epsilon$ be the desired tracking accuracy; if $0 < h \leq \min\left\{\frac{\mu\epsilon}{L\ell}, \frac{\gamma\epsilon}{2L\ell}, \frac{1}{L}\right\}$, then $(y_k)_{k=0}^\infty$ is $\epsilon$-shadowed by an orbit $(x_k)_{k=0}^\infty$ of $\Psi_h^{GD}$ up to iteration $K$.*

GENERAL SADDLES. We take inspiration from the literature on the **approximation of stiff ODEs** near stationary points (Lubich et al., 1995; Beyn, 1987; Larsson and Sanz-Serna, 1994) and use **Banach fixed-point**

26 Indeed, Thm. 2.4.3 applied backward in time from $y_K$ ensures that $\|(\Psi_h^{\mathrm{GD}})^{-K}(y_K) - x_0\| \leq \epsilon$.

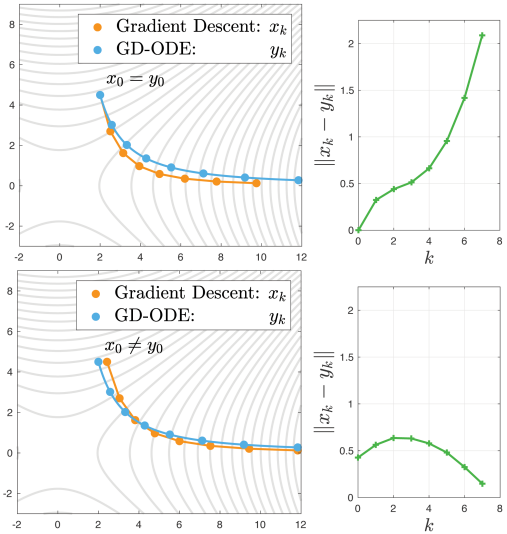


**Figure 14:** ***Top****: The orbit of GD is not a shadow: error blows up.* ***Bottom****: A few iterations of the maps $\Psi_h^{GD}$ and $\varphi_h^{GD}$ with different initializations on a quadratic saddle. GD-ODE was solved analytically and $h = 0.2$. On the bottom plot, the coordinates of $x_0$ are $x_{0,1} = \Psi^{-7}(y_{7,1})$ (expanding direction, need to reverse time) and $x_{0,2} = y_{0,2}$ (contracting direction).*

**theorem** to generalize the result above to perturbed quadratic saddles $f + \phi$, where $\phi$ is required to be $L_\phi$-smooth with $L_\phi \leq \mathcal{O}(\min\{\gamma, \mu\})$. This condition is intuitive, since $\phi$ effectively counteracts the contraction/expansion.

**Theorem 2.4.4.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a quadratic centered at $x^*$ with Hessian $H$ with no eigenvalues in the interval $(-\gamma, \mu)$, for some $\mu, \gamma > 0$. Let $g : \mathbb{R}^d \to \mathbb{R}$ be our objective function, of the form $g(x) = f(x) + \phi(x)$ with $\phi : \mathbb{R}^d \to \mathbb{R}$ a $L_\phi-$smooth perturbation such that $\nabla\phi(x^*) = 0$. Assume the orbit $(y_k)_{k=0}^{\infty}$ of $\varphi_h^{GD}$ on $g$ is s.t.* ***(H1)*** *(stated for $g$) holds, with gradients bounded by $\ell$ up to iteration $K$. Assume $0 < h \leq \frac{1}{L}$ and let $\epsilon$ be the desired tracking accuracy, if also*

$$h \leq \frac{\epsilon\left(\min\left\{\frac{\gamma}{2}, \mu\right\} - 4L_\phi\right)}{2\ell L},$$

*then* $(y_k)_{k=0}^{\infty}$ *is* $\epsilon$*-shadowed by a orbit* $(x_k)_{k=0}^{\infty}$ *of* $\Psi_h^{GD}$ *on* $g$ *up to iteration* $K$*.*

The proof of this theorem is quite involved, hence is not reported in this thesis; the interested reader can find it in the appendix of Orvieto and Lucchi, 2019b. In the special case of strongly convex quadratics, the theorem above recovers the shadowing condition of Cor. 2.4.1 up to a factor $1/2$ which is due to the different proof techniques.

GLUING LANDSCAPES. The last result can be combined with Thm. 2.4.3 to capture the dynamics of GD-ODE where directions of negative curvature are encountered during the early stage of training followed by a strongly convex regions as we approach a local minimum (such as the one in Fig. 15). Note that, since under **(H1)** the objective is $\mathcal{C}^2$, there will be a few iterations in the "transition phase" (non-convex to convex) where the curvature is very close to zero. These few iterations are not captured by Thm. 2.4.3 and Thm. 2.4.4; indeed, the error behaviour in Fig. 15 is pathological at $k \approxeq 10$. Nonetheless, as we showed for the convex case in Sec. 2.4.2.1, the approximation error during these iterations only grows as $\mathcal{O}(kh)$. In the numerical analysis literature, the procedure we just sketched was made precise in Chow and Van Vleck, 1994, who proved that a gluing argument is successful if the number of unstable directions on the ODE path is non-increasing.

### 2.4.3 *Shadowing Heavy-ball*

We now turn our attention to analyzing Heavy-ball whose continuous representation is $\ddot{q} + \alpha\dot{q} + \nabla f(q) = 0$, where $\alpha$ is a positive number called the viscosity parameter. Following Maddison et al., 2018, we introduce the velocity variable $p = \dot{q}$ and consider the dynamics of $y = (q, p)$ (i.e. in phase space).

$$\begin{cases} \dot{p} = -\alpha p - \nabla f(q) \\ \dot{q} = p \end{cases} . \tag{HB-ODE}$$

Under **(H1)**, we denote by $\varphi_{\alpha,h}^{\text{HB}} : \mathbb{R}^{2d} \to \mathbb{R}^{2d}$ the corresponding joint time-$h$ map and by $(y_k)_{k=0}^{\infty} = ((p_k, q_k))_{k=0}^{\infty}$ its orbit (i.e. the sampled HB-ODE trajectory). First, we show that a **semi-implicit** (Hairer et al., 2003) integration of Eq. (HB-ODE) yields HB.

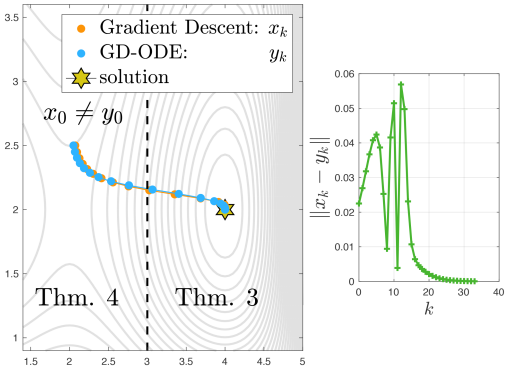


**Figure 15:** *Dynamics on the Hosaki function, $h = 0.3$ and lightly perturbed initial condition in the unstable subspace. ODE numerical simulation with Runge-Kutta 4 method Hairer et al., 2003.*

Given a point $x_k = (v_k, z_k)$ in phase space, this integrator computes $(v_{k+1}, z_{k+1}) \approxeq \varphi^{\mathrm{HB}}_{\alpha,h}(x_k)$ as

$$\begin{cases} v_{k+1} = v_k + h(-\alpha v_k - \nabla f(z_k)) \\ z_{k+1} = z_k + h v_{k+1} \end{cases} . \tag{HB-PS}$$

Notice that $v_{k+1} = (z_{k+1} - z_k)/h$ and $z_{k+1} = z_k - (1 - \alpha h)(z_k - z_{k-1}) - h^2 \nabla f(z_k)$, which is exactly one iteration of HB, with $\beta = 1 - h\alpha$ and $\eta = h^2$. We therefore have established a numerical link between HB and HB-ODE, similar to the one presented in Shi et al., 2019. In the following, we use $\Psi^{\mathrm{HB}}_{\alpha,h}$ to denote the one step map in phase space defined by HB-PS. Similarly to Remark 2.4.1, by Prop. 2.2 in Cabot et al., 2009, **(H1)** implies that gradients are bounded by a constant $\ell$. Hence, we can get an analogue to Prop. 2.4.2 (see App. B.2.4 for the proof).

**Proposition 2.4.5.** *Assume **(H1)** and let $y_0 = (0, z_0)$. Then, $(y_k)_{k=0}^{\infty}$ is a $\delta$-pseudo-orbit of $\Psi^{HB}_{\alpha,h}$ with $\delta = \ell(\alpha + 1 + L)h^2$.*

STRONG-CONVEXITY. The next step, as done for GD, would be to consider strongly convex landscapes and derive a **formula for the shadowing radius** (see Thm. 2.4.3). However — it is easy to realize that, in this

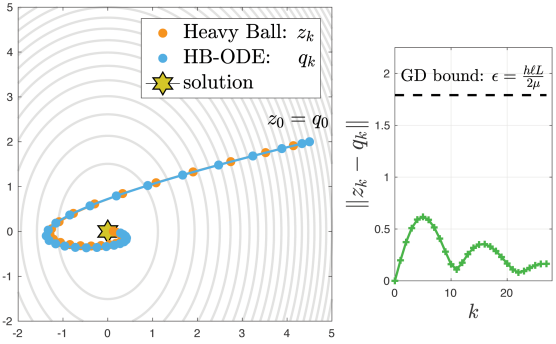


**Figure 16:** *Orbit of the space variable in $\Psi_h^{HB}$, $\varphi_h^{HB}$ (sampled ODE solution) on a strongly convex quadratic with $h = 0.2$ and $\alpha = 1$ . Solution to HB-ODE was computed analytically.*

setting, **HB is not uniformly contracting**. Indeed, it notoriously is **not a descent method**. Hence, it is unfortunately difficult to state an equivalent of Thm. 2.4.3 using similar arguments. We believe that the reason behind this difficulty lies at the **very core of the acceleration phenomenon**. Indeed, as noted by Ghadimi et al., 2015, the current known bounds for HB in the strongly convex setting might be loose due to the tediousness of its analysis (Shi et al., 2021)— which is also reflected here. Hence, we leave this theoretical investigation (as well as the connection to acceleration and symplectic integration (Shi et al., 2019)) to future research, and show instead experimental results in Sec. 2.4.4 and Fig. 16.

QUADRATICS. In App. B.2.4.1 we show that, if $f$ is quadratic, then $\Psi_{\alpha,h}^{\mathrm{HB}}$ is linear hyperbolic. Hence, as discussed in the introduction and thanks to Prop. 2.4.5, **there exists a norm**[27] **under which we have shadowing**, and we can recover a result analogous to Prop. 2.4.4 and to its perturbed variant. We show this empirically in Fig. 16 and compare with the GD formula for the shadowing radius.

27 For GD, this was the Euclidean norm. For HB the norm we have to pick is different, since (differently from GD) $\varphi_{\alpha,h}^{\mathrm{HB}}(x) = Ax$ with $A$ non-symmetric. The interested reader can find more information in App. 4 of Irwin, 2001.

### 2.4.4 *Experiments on Neural Networks*

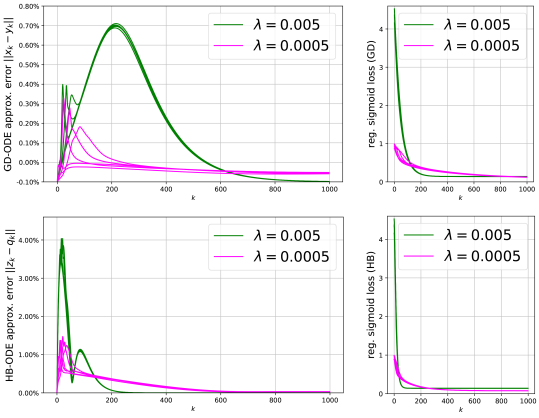


**Figure 17:** *Shadowing results under the sigmoid loss in MNIST (2 digits). We show 5 runs for the ODE and for the algorithm, with same (random) initialization. ODEs are simulated with 4th-order Runge-Kutta: our implementation uses 4 back-propagations and an integrator-step of* 0.1*. When trying higher precisions, results do not change. Shown are also the strictly decreasing (since we use full gradients) losses for each run the algorithms. The loss of the discretized ODEs are indistinguishible (because of shadowing) and are therefore not shown. We invite the reader to compare the results (in particular, for high $\lambda$) to the ones obtained in synthetic examples in Fig. 13 and 16.*

We consider the problem of binary classification of digits 3 and 5 from the MNIST data-set (LeCun, 1998). We take $n = 10000$ training examples $\{(a_i, l_i)\}_{i=1}^n$, where $a_i \in \mathbb{R}^d$ is the i-th image (in $\mathbb{R}^{785}$ adding a bias) and $l_i \in \{-1, 1\}$ is the corresponding label. We use the regularized sigmoid loss (non-convex) $f(x) = \frac{\lambda}{2}\|x\|^2 + \frac{1}{n}\sum_{i=1}^n \phi(\langle a_i, x\rangle l_i)$, $\phi(t) = \frac{1}{1+e^t}$. Compared to the cross-entropy loss (convex), this choice of $f$ often leads to better generalization (Shalev-Shwartz et al., 2011). For 2 different choices of $\lambda$, using the *full gradient*, we simulate GD-ODE using fourth-order Runge-Kutta (Hairer et al., 2003) (high-accuracy integration) and run GD with learning rate $h = 1$, which yields a steady decrease in the loss. We simulate HB-ODE and run HB under the same conditions, using $\alpha = 0.3$ (to induce a significant momentum). In Fig. 17, we show the behaviour

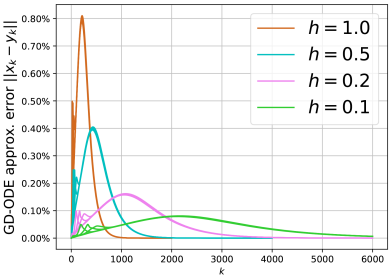


**Figure 18:** *Approx. error under same setting of Fig. 17 for $\lambda = 0.005$. Experiment validates the linear dependency on h in Cor. 2.4.1.*

of the approximation error, measured in percentage w.r.t. the discretized ODE trajectory, until convergence (with accuracy around 95%). We make a few comments on the results.

1. Heavy regularization (in green) increases the contractiveness of GD around the solution, yielding a **small approximation error** (it converges to zero) after a few iterations — exactly as in Fig. 13. For a small $\lambda$ (in magenta), the error between the trajectories is bounded but is slower to converge to zero, since local errors tend not to be corrected (cf. discussion for convex objectives in Sec. 2.4.2.1).

2. Locally, as we saw in Prop. 2.4.2, large gradients make the algorithm deviate significantly from the ODE. Since **regularization** increases the norm of the gradients experienced in early training, a larger $\lambda$ will **cause the approximation error to grow rapidly at the first iterations** (when gradients are large). Indeed, Cor. 2.4.1 predicts that the shadowing radius is proportional[28] to $\ell$.

3. Since HB has momentum, we notice that indeed it converges faster than GD (Sutskever et al., 2013). As expected, (see point 2) this has a **bad effect on the global shadowing radius**, which is 5 times bigger. On the other hand, the error from HB-ODE is also much quicker to decay to zero when compared to GD.

28 Alternatively, looking at the formula $\epsilon = \frac{hL\ell}{2\mu}$ in Cor. 2.4.1 and noting $\ell \leq L\|x_0 - x^*\|$, we get $\epsilon = \mathcal{O}(L^2/\mu)$. Hence, regularization, which increases $L$ and decreases $\mu$ by the same amount, actually increases $\epsilon$.

Last in Fig. 18 we explore the effect of the shadowing radius $\epsilon$ on the learning rate $h$ and find a **good match with the prediction** of Cor. 2.4.1. Indeed, the **experiment confirms that such relation is linear**: $\epsilon = \mathcal{O}(h)$, with *no dependency on the number of iterations* (as opposed to the classical results discussed in the introduction).
All in all, we conclude from the experiments that the intuition developed from our analysis can potentially explain the behaviour of the GD-ODE and the HB-ODE approximation in simple machine learning problems.

# 3

# UNDERSTANDING ADAPTIVE METHODS IN DEEP NETWORKS

*Ok, so, who is the most important author of the last twenty years? Careful now, not the best; virtuosity is for the arrogant.*

– Paolo Sorrentino, *The Young Pope*, 2016

In Chapter 2 we studied the first component of adaptive momentum methods — i.e. momentum, from a continuous-time perspective. In this chapter, we turn our attention to their second component: adaptive stepsizes. As we already discussed in the introduction, Adam (Kingma and Ba, 2014) and RMSprop (Tieleman and Hinton, 2012), as well as explicitly regularized variants such as AdamW (Loshchilov and Hutter, 2017), are known to **perform extremely well compared to SGD when training attention** models (Zhang et al., 2020b; Liu et al., 2019b; Wolf et al., 2020; Brown et al., 2020; Biggio et al., 2021), generative models (Karras et al., 2020) and RNNs (Hochreiter and Schmidhuber, 1997b). In this chapter, we seek the precise reason for this success — with the hope of further boosting performance and of guiding the design of the next generation of optimizers (see next chapter). Our analysis will be performed **in combination with a study of the landscape and signal propagation** of Multilayer Perceptrons (MLPs, Section 3.2) and Transformers (Section 3.3) at initialization. While the sections on MLPs and Transformers are separate, the findings are similar and highlight the power of Adam in adapting to complex high-dimensional landscapes.

## 3.1 RELATED WORKS ON THE ANALYSIS OF ADAM

Before diving into our contributions, in this section, we review some related work on the analysis of adaptive stepsizes.

STOCHASTIC NON-CONVEX OPTIMIZATION APPROACH TO ADAM.
The first correct[1] proof of convergence for (a modified variant of) Adam in the stochastic nonconvex case was given by Reddi et al., 2018, under a few assumptions (e.g. bounded gradients) in the framework of online optimization. Perhaps the most recent simple, up-to-date, complete, and elegant proof of convergence of Adam is the one recently given by Défossez et al., 2022, where the authors show that a rate $\mathcal{O}(\log(k)/\sqrt{k})$ can be obtained in expectation with iterate averaging. **The same rate is also achieved by vanilla SGD** (Ghadimi and Lan, 2013), which is well-known to be asymptotically optimal for non-convex stochastic programming with bounded variance (Arjevani et al., 2023), if one does not rely on variance reduction (else, one can achieve slightly faster convergence (Cutkosky and Orabona, 2019)). It is therefore clear that worst-case first-order complexity bounds which can be derived using the standard **non-convex optimization methodology** (at least for first-order stationary points) are **not yet able to explain the success of Adam** in the context of optimization of deep neural networks. Inspired by the issues above, in Chapter 4, we will revisit the convergence of adaptive methods and propose new stepsizes that have **provable adaptive behavior**.

GEOMETRY ADAPTATION, NOISE RESCALING, & OTHER CONJECTURES.
Some papers on Adam do not take the standard non-convex optimization approach discussed above. Chronologically, the first was Balles and Hennig, 2018, which "dissects" Adam highlighting a **variance-dependent preconditioning effect**. This insight was taken one step further by Staib et al., 2019, which shows how the variance-adaptation of **Adam effectively scales the gradient variance to be isotropic**; the authors claim this effect leads to fast escape from saddle points since all eigendirections become equally affected by stochastic perturbations.
Other papers take instead a geometric approach and motivate how Adam and RMSprop provide a cheap **diagonal approximation of the empirical Fisher preconditioner**, which is sometimes related to the Hessian (Martens, 2020). As a result, Adam can be thought of as an approximate Gauss-Newton method (Nocedal and Wright, 2006). While this interpretation could in principle explain the performance of Adam, it was re-

1 Quite interestingly, the original proof by Kingma and Ba, 2014 contains a few mistakes due to the problematic non-monotonic decrease of the effective stepsize.

cently shown (Kunstner et al., 2019) that very often the RMSprop preconditioner[2] can be far from the true Hessian, which is instead provably related to the non-empirical (a.k.a. true) Fisher. However, in Dauphin et al., 2015, the authors show that RMSprop effectively regularizes the landscape, making it more "equilibrated" (well-conditioned).
Finally, Zhang et al., 2019c relate the success of adaptive methods to an **underlying gradient clipping effect**: if big stochastic gradients are observed, those are smoothed out. The authors are able to show that clipped SGD, under quite uncommon assumptions on the cost function, is able to slightly improve (by a constant factor) over the rate of SGD, in some particular cases. While this does provide a quantitative improvement on SGD, the result is arguably not strong enough to motivate the success of adaptive methods in deep learning.

2 i.e. the diagonal matrix of per-dimension stepsizes, see Chapter 1.

## 3.2 POWER OF ADAM IN DEEP MLPS

*You don't understand anything until you learn it more than one way.*
– Marvin Minsky.

Multi-layer perceptrons (MLPs) with nonlinear activations are the bedrock upon which modern neural networks are constructed. It wasn't until 2015, when He et al. revolutionized the field with their pioneering work on **initializing ReLU MLPs effectively** (He et al., 2015), that training deep neural networks became significantly more feasible. Prior to this and to the advent of the Adam optimizer in 2014 (Kingma and Ba, 2014), practitioners had to rely heavily on intricate strategies and techniques to train neural networks effectively (Krizhevsky et al., 2012; Simonyan and Zisserman, 2014). With the introduction of proper initializatin (He et al., 2015) **residual connections** (He et al., 2016) and **batch normalization** (Ioffe and Szegedy, 2015), a significant transformation occurred. These innovations marked a turning point, enabling neural networks to be successfully trained using the more traditional Stochastic Gradient Descent (SGD) optimization method. The combination of residual connections and batch normalization tackled issues such as **vanishing gradients**, making the training process much more accessible and less reliant on advanced optimization techniques. Despite this, it's important to note that the **advantage of the Adam optimizer persists** in scenarios where networks are exceptionally deep (Tan and Le, 2019). In such cases, Adam can still offer benefits, streamlining the training process and contributing to faster convergence. It is also worth mentioning that, years ago, it was believed that **Adam had at times worse generalization than SGD** Wilson et al., 2017. This issue was **solved by the AdamW variant**, which correctly implements weight decay into the method (Loshchilov and Hutter, 2017).

In this section, we investigate the factors contributing to the remarkable success of the Adam optimization algorithm in effectively training deep MLPs and convolutional networks (CNNs) faced with **potentially suboptimal initialization or lack of normalization**. As such, our investigation is in some sense retrospective, yet highlights interesting properties of neural landscapes and intriguing features of adaptive methods.

**Question:** Initialization is known to be crucial for unlocking training of deep multi-layer perceptions (MLPs) and Convolutional Networks (CNNs) (Glorot and Bengio, 2010; He et al., 2015) with SGD. Which landscape features makes training with SGD challenging when initialization is not correct? Is Adam able to train even in contexts where no initialization can provide "nice" initialization landscapes?

**Answer (Orvieto et al., 2022b):** In deep MLPs and CNNs, suboptimal initialization creates flat saddle points that pose a significant challenge for SGD. Increasing the network width allows the design of specific initialization schemes (Glorot and Bengio, 2010; He et al., 2015) to solve the issue. However, in the case of very deep models, the expectation analysis of (Glorot and Bengio, 2010; He et al., 2015) becomes problematic and again SGD fails to train effectively. Our study demonstrates how Adam solves this challenge, enabling surprisingly rapid convergence on such difficult initialization landscapes. In essence, our findings provide theoretical support for the observation that training deep networks without normalization favors an adaptive stepsize.

### 3.2.1 *Notation and Background on Initialization*

In our theoretical analysis, we consider the L2 loss associated with a multilayer perceptron (MLP)

$$\begin{aligned} \mathcal{L}(\mathbf{W}) &= \frac{1}{2n}\sum_{i=1}^{n} \|\mathbf{y}_i - \mathbf{B}\mathbf{D}^L\mathbf{W}_\phi^{L:1}\mathbf{A}\mathbf{x}_i\|_2^2, \\ \mathbf{W}_\phi^{L:1} &:= \mathbf{W}^L\mathbf{D}^{L-1}\mathbf{W}^{L-1}\cdots\mathbf{W}^2\mathbf{D}^1\mathbf{W}^1\mathbf{D}^0, \end{aligned} \tag{49}$$

where $\mathbf{x}_i \in \mathbb{R}^{d_{in}}$, $\mathbf{y}_i \in \mathbb{R}^{d_{out}}$, $\mathbf{A} \in \mathbb{R}^{d\times d_{in}}$,$\mathbf{B} \in \mathbb{R}^{d_{out}\times d}$, and $\mathbf{W}^\ell \in \mathbb{R}^{d\times d}, \forall \ell = 1,\dots,L$ . $\mathbf{D}^\ell$ is the diagonal matrix of activation gates w.r.t the nonlinearity $\phi$ at layer $\ell$, which we consider to be either $\phi(x) = x$ (linear networks) or $\phi(x) = \max\{x, 0\}$ (ReLU networks).

**Assumption 3.2.1** (Random initialization). *Each entry of $\mathbf{W}^\ell$ ($\ell = 1,\dots,L$) is initialized i.i.d. with some distribution $\mathcal{P}$ symmetric around zero with variance $\sigma^2 < \infty$ and fourth moment $\mu_4 < \infty$.*

For more than a decade, the standard choice of initialization variance was $\sigma^2 = \frac{1}{3d}$ (LeCun et al., 1998). Motivated by repeated observations of the **vanishing gradient problem**, an **improved initialization** was suggested by Glorot and Bengio, 2010 and He et al., 2015. The following theorem summarizes the reasoning in He et al., 2015. We define the parameter $p = 1$ for the linear case and $p = 1/2$ for the ReLU case

**Proposition 3.2.1** (Glorot and Bengio, 2010, He et al., 2015)**.** *Under Assumption 3.2.1, $Var(\partial \mathcal{L}(\mathbf{W})/\partial \mathbf{W}^\ell)$ scales as $(pd\sigma^2)^L$ across all layers $\ell = 1, \dots, L$. When initializing with $\sigma^2 = \frac{1}{3d}$ (LeCun init.), this quantity vanishes in depth. Instead, choosing $\sigma^2 = 1/d$ in the linear case (Xavier init.), and $\sigma^2 = 2/d$ in the ReLU case (He init), stabilizes the variance.*

*Proof.* Let $\mathbf{a}^{\ell+1} = \mathbf{W}^\ell \mathbf{h}^\ell$ be the preactivation of layer $\ell + 1$, $\mathbf{h}^\ell = \mathbf{D}^\ell \mathbf{a}^\ell$, the activation at layer $\ell$ . At random initialization, clearly, the components of $\mathbf{h}, \mathbf{a}$ at each layer are identically distributed. Let $a^{\ell+1}$, $w^\ell$ and $h^\ell$ represent the random variables corresponding to each element in $\mathbf{a}^{\ell+1}$, $\mathbf{W}^\ell$ and $\mathbf{h}^\ell$ respectively. Since $w^\ell$ is zero mean, we have that $\text{Var}[a^{\ell+1}] = d \cdot \text{Var}[w^\ell] \cdot \mathbb{E}[(h^\ell)^2]$. Finally, since $\mathbb{E}[(h^\ell)^2] = p\text{Var}[a^\ell]$, we end up with $\text{Var}[a^{\ell+1}] = d\sigma^2 p \cdot \text{Var}[a^\ell]$, which yields $\text{Var}[a^{\ell+1}] = \text{Var}[a^\ell]$ for $\sigma^2 = \frac{1}{dp}$. □

For example, for the uniform initialization $\mathcal{P} = \mathcal{U}[-\tau, \tau]$ we have $\sigma^2 = \tau^2/3$ and hence the "optimal" initialization range amounts to $\tau = \sqrt{3/d}$ in the linear - and $\tau = \sqrt{6/d}$ in the ReLU case.

### 3.2.2 *Adam on the Neural Chain*

To illustrate an **important shortcoming in the analyses** of Glorot and Bengio, 2010 and He et al., 2015, we consider a deep linear network of unit width (henceforth called **neural chain**). While these networks are utterly useless for practical applications, they are sufficient to **exhibit some critical properties of the loss landscape**, that generalize to wider nets (see next subsection) and can make us appreciate the power of adaptive methods such as Adam (Kingma and Ba, 2014).
In the neural chain case, if $\mathbf{A} = \mathbf{B} = 1$, Eq.(49) simplifies to

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2n} \sum_{i=1}^{n} (y_i - w_L \dots w_1 x_i)^2. \tag{50}$$

We consider each $w_i \in \mathbb{R}$ to be drawn uniformly at random in $[-\tau, \tau]$.

SHORTCOMINGS OF THE EXPECTATION ANALYSIS. Prop. 3.2.1 suggests that both forward pass and **gradient remain stable** in magnitude when choosing $\tau = \sqrt{3}$. While this **is true in expectation**, it is not the case when initializing individual models, where the expected value becomes an **increasingly atypical event** (see Thm. 3.2.1) as the chain grows in depth ($L$). Indeed, in Fig. 20 we see that all quantities vanish under the "optimal" initialization. Perhaps the most intuitive indication for this pathological behavior comes from writing down the population quantities for the absolute value of the input-output map.

**Proposition 3.2.2** (Forward pass statistics chain)**.** *Consider the absolute value of a forward pass on the chain, i.e. the random variable* $v_{\tau,L} := \prod_{k=1}^{L}(\tau w_k)$, *with* $w_k \overset{iid}{\sim} \mathcal{U}(0,1]$. *Then,*

$$\mathbb{E}\left[v_{\tau,L}\right] = \left(\tfrac{\tau}{2}\right)^L, \quad \mathbb{E}[v_{\tau,L}^2] = \left(\tfrac{\tau^2}{3}\right)^L, \quad \mathbb{E}[v_{\tau,L}^3] = \left(\tfrac{\tau^3}{4}\right)^L. \tag{51}$$

*Clearly,* $\tau = \sqrt{3}$ *(Xavier init.) leads to (as* $L \to \infty$*)*

$$\mathbb{E}\left[v_{\tau,L}\right] \to 0, \quad \mathbb{E}\left[v_{\tau,L}^2\right] = 1, \quad \mathbb{E}[v_{\tau,L}^3] \to \infty.$$

The result follows directly from the moments of the uniform distribution. It might be tempting to conclude from Eq. (51) that picking $\tau = 2$ instead of $\sqrt{3}$ solves the problem. Yet, this is not the case since then $\mathbb{E}[v_{\tau,L}^2] \to \infty$ and by Mallows inequality (Mallows and Richter, 1969) the mean becomes an unreliable predictor for the median, as their difference is bounded by one standard deviation (exploding). In fact, the above proposition reveals that **one cannot stabilize any pair of moments** of $v_{\tau,L}$ **simultaneously** and hence in both cases $\tau = \sqrt{3}$ and $\tau = 2$, the distribution of $v_{\tau,L}$ becomes **fat-tailed** as $L \to \infty$, which leads to slow convergence of the central limit theorem[3]. As we show in Sec. 3.2.3, this basic moment trade off prevails in wider nets (see Eq. (62)).

3 The speed of convergence in the CLT, as bounded by the Berry-Esseen inequality, is proportional to $\mathbb{E}[|v|^3]$.

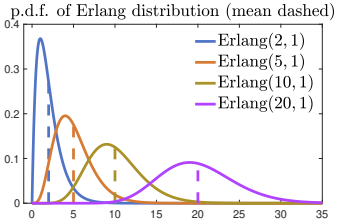


**Figure 19:** *Density of the Erlang Distribution Erlang*$(L, 1)$.

BETTER CHARACTERIZATION THROUGH A MEDIAN ANALYSIS. One can go beyond the population analysis in order to better understand this phenomenon. In the first step, we characterize the distribution of the magnitude of the input-output map in log scale.

**Lemma 3.2.1** (Distribution of chain input-output)**.** *In the setting of Prop. 3.2.2,*

$$-\ln(v_{\tau,L}) = z - L\ln(\tau), \quad z \sim \text{Erlang}(L, 1).$$

*Hence* $\Pr(-\ln(v_{\tau,L}) \leq \zeta) = 1 - e^{-\xi}\sum_{k=0}^{L-1}\frac{\xi^k}{k!}$, $\xi := \zeta + L\ln\tau$.

*Proof.* Basic logarithm identities allow us to write

$$-\ln(v_{\tau,L}) = -\ln\left(\prod_{k=1}^{L}(\tau w_k)\right) = -L\ln(\tau) - \sum_{k=1}^{L}\ln(w_k). \tag{52}$$

Clearly, $-\ln(w_k))$ is exponentially distribution with parameter 1. Furthermore, if random variables $V_k \sim \text{Exp}(\lambda)$ are independent, then $\sum_{k=1}^{L} V_k \sim \text{Erlang}(L, \lambda)$ (Temme, 1996; Devroye, 2006). The CDF follows from the properties of the Erlang distribution. □

This lemma allows to characterize the **median** and provide an asymptotic[4] bound on the forward pass norm.

**Proposition 3.2.3** (Expectation is not predictive for input-output map magnitude)**.** *We have*

$$\text{median}\,[v_{\tau,L}] = e^{L\ln(\tau)-\tilde{L}},$$

4 In the context of asymptotic expansions, we write $f \sim g$ if $\lim_{L\to\infty} f(L)/g(L) = 1$.

*with* $L - \frac{1}{3} \leq \tilde{L} \leq L - 1 + \ln(2)$. *Therefore, if* $\tau = 2$, $\text{median}\left[v_{\tau,L}\right] \to 0$ *while* $\mathbb{E}\left[v_{\tau,L}\right] = 1$ *and* $\mathbb{E}\left[v_{\tau,L}^2\right] \to \infty$. *For* $\tau = \sqrt{3}$, *also* $\text{median}\left[v_{\tau,L}\right] \to 0$. *Yet, the median is stabilized for* $\tau = e$, *since*

$$\lim_{L\to\infty} \left(v_{\tau,L}\right)^{\frac{1}{L}} \overset{a.s.}{=} \tau/e,$$

*which implies* $v_{\tau,L} \sim \exp\left(-L(1 - \ln\tau)\right)$.

*Proof.* The moments follow from Prop. 3.2.2. For the median, we solve $\Pr(-\ln v_{\tau,L} \leq \zeta) = 1/2$ for $\zeta$, which by Lemma 3.2.1 is equivalent to solving $1 - e^{-\xi}\sum_{k=0}^{L-1}\frac{\xi^k}{k!} = 1/2$ w.r.t. $\xi := \zeta + L\ln\tau$. The solution, termed $\tilde{L}$, is approximated with a Ramanujan formula (1913), as in Choi, 1994. Since $v_{\tau,L} = e^{L\ln\tau - z}$, then $\left(v_{\tau,L}\right)^{\frac{1}{L}} = \tau e^{-z/L}$. We conclude with the strong law of large numbers. □

We now apply the idea behind the last result to analyze the first and second order partial derivatives.

**Theorem 3.2.1.** *Assume bounded data and* $w_i \sim \mathcal{U}[-\tau,\tau]$, *with fixed* $\tau$. *For each* $k, \ell \leq L$ *we asymptotically (as* $L \to \infty$*) have almost surely that*

$$\left|\frac{\partial \mathcal{L}_{chain}(\mathbf{w})}{\partial w_k}\right|, \left|\frac{\partial^2 \mathcal{L}_{chain}(\mathbf{w})}{\partial w_k \partial w_{\ell\neq k}}\right| = \mathcal{O}\left(e^{-(L-1)(1-\ln\tau)}\right),$$
$$\left|\frac{\partial^2 \mathcal{L}_{chain}(\mathbf{w})}{\partial w_k \partial w_k}\right| = \mathcal{O}\left(e^{-2(L-1)(1-\ln\tau)}\right).$$

*In particular, as for* $v_{\tau,L}$, *all these quantities asymptotically vanish if* $\tau < e$ *and explode if* $\tau > e$.
*In the case of Xavier init.* $\tau = \sqrt{3}$, *the Hessian vanishes in norm (hence eigenvalues vanish) and becomes hollow, i.e. diagonal elements become exponentially smaller than off-diagonal elements.*

The proof is presented in Appendix C.1 of this thesis. Empirical simulations in Fig. 20 show that the result is very precise.

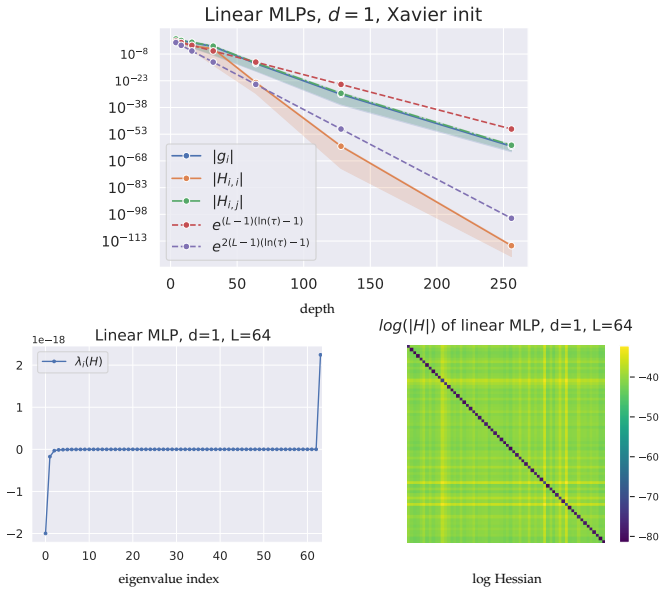


**Figure 20:** *Numerical verification of Theorem 3.2.1.* ***Top:*** *Gradient and Hessian entry magnitudes for deep neural chains (no activations, Xavier init). Plotted is mean and 95% confidence interval of 10 random seeds)* ***Bottom:*** *Eigenvalues and log Hessian entry maginute at init. for* $L = 64$.

IMPLICATIONS ON LANDSCAPE AND OPTIMIZATION. In narrow networks, our results show vanishing gradients and hollow Hessians with positive and negative eigenvalues of decreasing magnitude. Hence, the initialization landscape is a plateau that resembles **a barely curved saddle** (see Fig. 21). As discussed next, this is particularly **bad for optimization with plain SGD but adaptive methods escape the plateau quickly** due to a notable curvature adaptation capability.

To illustrate this point, we consider a single datapoint $(x, y)$ with $x, y > 0$ and study the **gradient flow** on a neural chain of depth $L$ with initialization $0 < w_1(0) = w_2(0) = \cdots = w_L(0) := w_0 \in \mathbb{R}$. The gradient

$$\nabla_{w_i} \mathcal{L}(\mathbf{w}) = x \prod_{j \neq i} w_j \left( \prod_r w_r x - y \right) \tag{53}$$

is invariant w.r.t. any permutation of the $w_i$'s. Hence, each coordinate of the gradient flow solution will satisfy

$$w_1(t) = w_2(t) = \cdots = w_L(t) =: w(t) \tag{54}$$

and

$$w(t) \to w^* = (y/x)^{1/L}, \text{ as } L \to \infty. \tag{55}$$

Therefore, the gradient flow is

$$\frac{d}{dt}w(t) = -w(t)^{2L-1}x^2 + w(t)^{L-1}xy. \tag{56}$$

To simplify this we can take the special case $x = y$ (which leads to $w^* = 1$) and drop the first term (negative). Hence we get an upper bounding solution (since $w(t)$ is increasing) which explodes in finite time $t_e$ (see Fig. 21):

$$w(t) \leq [(L-2)(t_e - t)]^{-\frac{1}{L-2}}, \quad t_e = w_0^{2-L}/(L-2). \tag{57}$$

In this case, the upper bound for $w(t)$ reaches $w^* = 1$ at time

$$t^* = t_e - \frac{1}{L-2} \tag{58}$$

which is **exponential in the network depth** $L$. This provides a proof for the following proposition.

**Proposition 3.2.4** (Slow convergence of Gradient Flow on the chain)**.** *On neural chains, in the worst case, gradient flow takes exponential (in depth) time to reach an $\epsilon$-neighbour of the solution.*

CURVATURE ADAPTATION OF RMSPROP. As can be seen in Figure 21, RMSprop (Tieleman and Hinton, 2012) is able to optimize the neural chain in a number of iterations **independent of depth**. This finding is also observable in wider MLPs (Fig. 24) and deep convnets (Fig.25).
To provide some intuition around this phenomenon, we apply RMSprop to the neural chain gradient flow approximation $\dot{w}(t) = w(t)^{L-1}$. This approximation is tight during the first steps of the optimizer if $L$ is big. The RMSprop flow solves $\dot{w}(t) = w(t)^{L-1}/\sqrt{v(t)}$, where $v(t)$ is a low-pass filter on the approximate square gradient $w(t)^{2L-2}$. Since $w(t)^{L-1}$ is

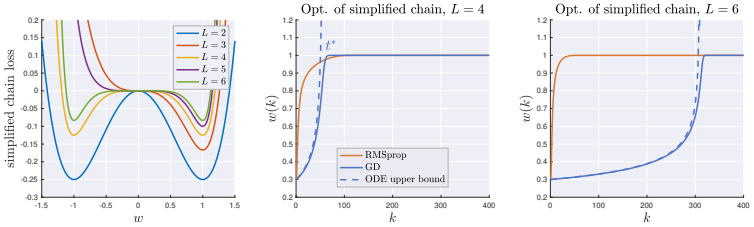


**Figure 21:** *Chain setting of Prop. 3.2.4. Fast convergence of RMSprop with $\beta_2 = 0.9$ and stepsize decay. For GD, $\eta = 0.1$ is used since bigger $\eta$ leads to instability). Plotted is also the loss corresponding to the integrated gradient $w^{2L-1} + w^{L-1}$. To discretize the bound, we use Euler discretization with equivalence $\eta k \equiv t$.*

increasing, $v(t)$ is also increasing and the filter delay guarantees $\sqrt{v(t)} \leq w(t)^{L-1}$. It follows that $\dot{w}(t) > 1$ for $t$ small, *regardless of the network depth*, which allows RMSprop to quickly escape the flat plateau (see Figure 22). In the vanishing curvature setting predicted by Thm. 3.2.1 and confirmed in Fig.20 & 21 (leftmost plot), this speedup is not extremely surprising. Many recent works report an **improved curvature adaptation of adaptive methods** compared to SGD (Dauphin et al., 2015; Kunstner et al., 2019). For instance, Staib et al., 2019 recently showed that RMSprop is provably faster than SGD around flat saddle points (see Section 5.2 of their paper). This result, in combination with our findings on the flatness of the initialization landscape, gives an **explanation for the historical difficulties of training deep nets with SGD and for the success of adaptive methods**.

INSIGHTS FROM THE QUADRATIC SETTING. Towards a motivation of the results in Figure 22, consider optimizing a one-dimensional quadratic with curvature $\lambda$. Unrolling the update of $v_k$ in RMSprop:

$$\begin{aligned} v_{k+1} &= \beta_2 v_k + (1-\beta_2)\gamma^2 w_{k+1}^2 \\ \implies v_k &= \beta_2^k \gamma^2 w_0^2 + (1-\beta_2)\sum_{j=0}^{k} \beta_2^{k-j}\gamma^2 w_k^2. \end{aligned} \tag{59}$$

Therefore, ignoring the negligible effect of $\epsilon$, we have

$$w_k - \frac{\eta}{\sqrt{v_k}}\gamma w_k = w_k - \frac{\eta}{\sqrt{\beta_2^k \gamma^2 w_0^2 + (1-\beta_2)\sum_{j=0}^{k}\beta_2^{k-j}\gamma^2 w_k^2}}\gamma w_k. \tag{60}$$

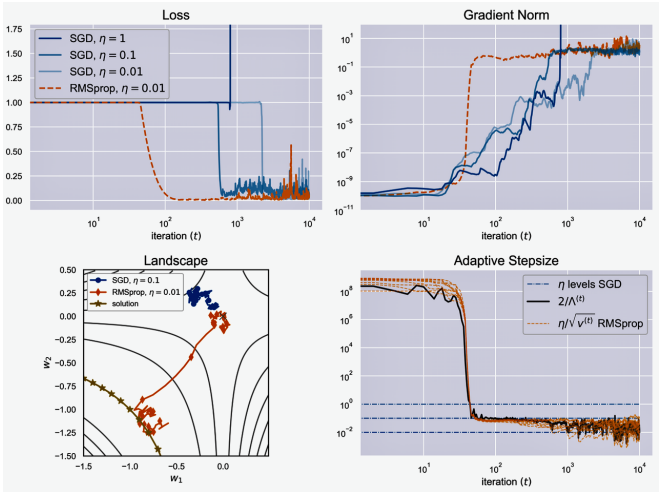


**Figure 22:** *Dynamics of RMSprop and gradient descent on a 10-dimensional neural chain. For RMSprop is able to optimize the loss, while GD can't escape fast the initial saddle point, regardless of the magnitude of stepsize. We also show how the adaptive stepsizes chosen by RMSprop effectively adapt to the maximum hessian eigenvalue $\Lambda$.*

Hence, the curvature $\gamma$ influences the update of $v_k$ but does not influence the update of $w_k$.

EFFECT OF NOISE. While it is known that the inherent sampling noise of SGD is anisotropic and in many settings aligned with negative curvature (Daneshmand et al., 2018; Zhu et al., 2019; Li et al., 2020), the saddle escape time still depends inversely on the magnitude of the smallest eigenvalue (Daneshmand et al., 2018; Curtis and Robinson, 2019). As a result, similar to gradient flow on the chain, **SGD is unable to train networks with vanishing gradients/curvature** despite the presence of inherent noise (see also Fig. 24 & 25). Another possibility is to directly add noise to the updates (Du et al., 2017; Du et al., 2019). In addition, in Fig. 63-64 (Appendix C.1), we provide evidence that **noise can accelerate GD** on the chain, but it is still **orders of magnitude slower than RMSprop**, for any noise level and learning rate.

### 3.2.3 *Adam on wide MLPs*

In analogy with Prop. 3.2.2 for the chain, we here show that also in the general MLP case, **different population quantities cannot be jointly stabilized** using standard i.i.d. initialization.

**Proposition 3.2.5** (Forward pass statistics MLP)**.** *Let* $\mathbf{z} = \mathbf{A}\mathbf{x}$*. Let* $\kappa = \mu_4/\sigma_4$ *be the kurtosis (fourth standardized moment) of the initialization distribution (see Assumption 3.2.1). Let* $p = 1$ *in the linear case and* $p = 1/2$ *in the ReLU case. Then we have*

$$\begin{aligned} \mathbb{E}\|\mathbf{W}_\phi^{k:1}\mathbf{z}\|_2^2 &= (d\sigma^2 p)^k \mathbb{E}\|\mathbf{z}\|_2^2. \\ \begin{pmatrix} \mathbb{E}\|\mathbf{W}_\phi^{k:1}\mathbf{z}\|_2^4 \\ \mathbb{E}\|\mathbf{W}_\phi^{k:1}\mathbf{z}\|_4^4 \end{pmatrix} &= \left(p^2 d\sigma^4\right)^k \mathbf{Q}^k \begin{pmatrix} \mathbb{E}\|\mathbf{z}\|_2^4 \\ \mathbb{E}\|\mathbf{z}\|_4^4 \end{pmatrix}, \\ \textit{with} \quad \mathbf{Q} &:= \begin{pmatrix} d+2 & \frac{\kappa-3+(1-p)(d+2)}{p} \\ 3 & \frac{\kappa-3p}{p} \end{pmatrix}. \end{aligned} \tag{61}$$

We do not present a proof for result in the appendix, since this would require several pages of notation and technical calculations. The interested reader can check (Orvieto et al., 2022b) for a complete discussion. We instead present a compelling empirical verification for this result in Appendix C.1.4.

For deep linear nets of arbitrary width $d$ and Gaussian initialization ($\kappa = 3$) the result above simplifies to

$$\begin{aligned} \mathbb{E}\|\mathbf{W}^{L:1}\mathbf{A}\mathbf{x}\|_2^2 &= (d\sigma^2)^L \mathbb{E}\|\mathbf{A}\mathbf{x}\|_2^2, \\ \mathbb{E}\|\mathbf{W}^{L:1}\mathbf{A}\mathbf{x}\|_2^4 &= \left(d\sigma^4\right)^L (d+2)^L\, \mathbb{E}\|\mathbf{A}\mathbf{x}\|_2^4. \end{aligned} \tag{62}$$

Hence, as for the neural chain (see Prop. (51)), picking the Xavier initialization $\sigma^2 = \frac{1}{d}$ stabilizes $\mathbb{E}\|\mathbf{W}^{L:1}\mathbf{A}\mathbf{x}\|_2^2$, but $\mathbb{E}\|\mathbf{W}^{L:1}\mathbf{A}\mathbf{x}\|_2^4 = \left(\frac{d+2}{d}\right)^L \mathbb{E}\|\mathbf{A}\mathbf{x}\|_2^4$ explodes *unless* $d$ grows faster than $L$. This points to an important shortcoming of the initialization proposed in Glorot and Bengio, 2010 & He et al., 2015, which — as we note next — is only guaranteed to **prevent vanishing gradients and curvature in networks that are wider than deep**. The next result is verified empirically in Figure 23.

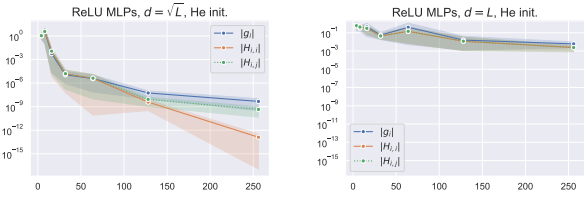


**Figure 23:** ***Effect of width in ReLU MLPs:*** *Gradient and curvature scaling on Fashion-MNIST over depth. While quantities vanish on the left ($d = \sqrt{L}$), the right shows stable magnitudes. Mean and 95% CI of 15 runs.*

**Theorem 3.2.2.** *The initialization in Glorot and Bengio, 2010 & He et al., 2015 is guaranteed to stabilize both the mean and the median of the squared forward pass norm for $d = \Omega(L)$.*

*Proof.* We carry out the proof for the Gaussian linear case, but the other settings are conceptually equivalent. First, in the case $\sigma^2 = 1/d$, we have

$$\mathrm{Var}\|\mathbf{W}^{L:1}\mathbf{A}\mathbf{x}\|_2^2 = \mathbb{E}\|\mathbf{W}^{L:1}\mathbf{A}\mathbf{x}\|_2^4 - \left(\mathbb{E}\|\mathbf{W}^{L:1}\mathbf{A}\mathbf{x}\|_2^2\right)^2 \overset{Eq.\ (62)}{=} \left(\frac{d+2}{d}\right)^L - 1. \tag{63}$$

By Mallows inequality Mallows and Richter, 1969, the square root of this quantity provides a upper bound on

$$\left|\mathrm{median}\left(\|\mathbf{W}^{L:1}\mathbf{A}\mathbf{x}\|_2^2\right) - \mathbb{E}\|\mathbf{W}^{L:1}\mathbf{A}\mathbf{x}\|_2^2\right|. \tag{64}$$

If we want to guarantee a *non-vanishing median*, say in $[1-\alpha, 1+\alpha]$, for $1 > \alpha > 0$, we need to have $d$ s.t. $\left(\frac{d+2}{d}\right)^L - 1 \leq \alpha^2$, which implies $d \geq \frac{2}{(\alpha^2+1)^{\frac{1}{L}}-1}$. A Maclaurin's series expansion gives

$$(\alpha^2+1)^{\frac{1}{L}} = 1 + \ln(\alpha^2+1)/L + \mathcal{O}(1/L^2), \tag{65}$$

which concludes the proof for large enough $L$. □

Before discussing the regime $d \ll L$, we consolidate our core claim about vanishing curvature by generalizing the results of Glorot and Bengio, 2010; He et al., 2015 to second-order derivatives.

**Theorem 3.2.3** (Gradient and Hessian in exp.). *Under Ass. 3.2.1, the **expected** norm of any Hessian diag block* $\mathbb{E}[\| \frac{\partial^2 \mathcal{L}(\mathbf{W})}{\partial \mathbf{W}_k \partial \mathbf{W}_k} \|_F]$ *in linear nets scales as* $\mathcal{O}\left((d\sigma^2)^L\right)$*, while off-diag. blocks* $\mathbb{E}[\| \frac{\partial^2 \mathcal{L}(\mathbf{W})}{\partial \mathbf{W}_k \partial \mathbf{W}_\ell} \|_F]$ *and the gradient* $\mathbb{E}[\| \frac{\partial \mathcal{L}(\mathbf{W})}{\partial \mathbf{W}_k} \|_F]$ *scale as* $\mathcal{O}\left((d\sigma^2)^{\frac{L}{2}}\right)$*. In ReLU nets the scaling amounts to* $\mathcal{O}\left(\left(\frac{d}{2}\sigma^2\right)^L\right)$ *and* $\mathcal{O}\left(\left(\frac{d}{2}\sigma^2\right)^{\frac{L}{2}}\right)$ *respectively.*

This result is (to the best of our knowledge) the first to study the effects of depth on second-order derivatives at random initialization. Its proof, which mainly builds upon Prop. 3.2.5, and a sketch can be found in the appendix. A simple application of Gershgorin's theorem yields the following bound on the eigenvalues.

**Corollary 3.2.1.** *Under Assumption 3.2.1, the **expected** magnitude of the largest eigenvalue* $\lambda_{\max}$ *is upper bound as* $\mathbb{E}[|\lambda_{\max}^{linear}|] \leq Ld \cdot \mathcal{O}\left((d\sigma^2)^{\frac{L}{2}}\right)$ *and* $\mathbb{E}[|\lambda_{\max}^{ReLU}|] \leq Ld \cdot \mathcal{O}\left((\frac{d}{2}\sigma^2)^{\frac{L}{2}}\right)$ *respectively.*

Our results in this subsection can be compressed in the next theorem.

**Theorem 3.2.4** (Main result). *For* $d = \Omega(L)$ *the initialization in Glorot and Bengio, 2010 & He et al., 2015 is guaranteed to stabilize both the gradient and the Hessian **with high probability**. Instead, if* $d \ll \mathcal{O}(L)$*, then both these initializations can yield vanishing gradients/curvature almost surely.*

The theorem above simply illustrates that there must be a transition in behavior from the $d = 1$ case to the $d = \Omega(L)$ case. Arguably, the precise characterization of this transition is of little theoretical interest — as the fundamental insight is the existence of such transition. Empirically, we observe that $d = \sqrt{L}$ still yields vanishing gradients.

Our result, combined with the analysis of RMSprop in the vanishing curvature regime in Section 3.2.2, justifies the performance observed in deep networks in Figures 24 and 25: without additional tricks such as residual connections (He et al., 2016) or batch normalization (Ioffe and Szegedy, 2015), **very deep networks of moderate width are not trainable with SGD, but they are with RMSprop**.


**Figure 24:** ***(Top row)*** *Training Fashion-MNIST (Xiao et al., 2017) on a* ***narrow*** *(32 hidden units per layer) 128 layer ReLU MLP with He init. Learning rates are grid-searched (best is shown). Plotted are mean and 95% confidence interval of 10 random seeds. See the appendix in Orvieto et al., 2022b for hyperparameters and test accuracy. Note that RMSProp successfully trains despite the fact that both gradients and curvature vanished. Yet, as can be deduced from how gradient norms evolve over time, SGD struggles to escape the flat plateau which is not surprising given that the negative eigenvalues are very small.* ***(Bottom row)*** *Training Fashion-MNIST on a* ***wide*** *(128 hidden units per layer) 128 layer ReLU MLP (He init.) Training accuracy and gradient magnitude over epochs as well as eigenvalues at initialization. Increased width prevents vanishing and allows SGD to train.*


**Figure 25:** *CIFAR-10 on a 500 layer convolutional network with He init. Test accuracy and gradient norm over epochs as well as eigenvalue histogram at initialization. Plotted is mean and 95% confidence interval of 10 random seeds.*

## 3.3 POWER OF ADAM IN TRANSFORMERS

*You look around. Your mind seeks,*
*makes harmonies, falls apart*
*in the perfume, expands*
*when the day wearies away.*
*There are silences in which one watches*
*in every fading human shadow*
*something divine let go.*

– Eugenio Montale.

Since its first appearance in Vaswani et al., 2017, the Transformer architecture — hinging on the ingenious **attention mechanism** — has brought a revolution in the realm of Natural Language Processing (NLP), as known e.g. from OpenAI, 2023. This architectural marvel has achieved remarkable success across a spectrum of tasks, including text classification (Yang et al., 2019), machine translation (Conneau et al., 2019), reading comprehension (Brown et al., 2020), and question answering (Raffel et al., 2020). Recent endeavors have masterfully extended its applicability beyond the confines of NLP. Remarkably, this groundbreaking architecture has seamlessly extended its influence into the domain of computer vision (Dosovitskiy et al., 2020), as well as other diverse domains (Baevski et al., 2020; Huang et al., 2018; Biggio et al., 2021; Polu et al., 2022), further cementing its popularity and impact.

**Question:** Evidence shows that **SGD cannot successfully train deep attention-based models**, even after tricks such as normalization and learning rate warm-up. State-of-the-art transformers are indeed trained almost exclusively with Adam variants (Touvron et al., 2023). In Section 3.2 we showed that, on vanilla MLPs and CNNs, SGD becomes ineffective when width is substantially smaller than depth due to vanishing curvature. Is the poor performance of SGD in these settings related to similar issues? If not, what's happening there?!

**Answer (Noci et al., 2022):** The poor performance of SGD is rooted in more fundamental properties of the Transformer architecture. Namely, the landscape curvature at initialization is not isotropic: some directions are less curved than others. Through an in-depth analysis of signal propagation, we are able to precisely characterize this issue and propose a fix that makes SGD work. While the heart of this section is motivating the performance of Adam, part of the discussion is dedicated to the rank collapse phenomenon, which makes deep transformers hard to train in general. We propose a fix for this issue too.

The Transformer operates on inputs comprising a sequence of tokens. At its core, it relies on stacked attention layers, which compute a measure of relevance for the whole sequence by assigning token-wise importance weights — obtained by matrix multiplication of the *queries* and *keys*, and finally normalized with the softmax function. The output of an attention layer is then a linear combination of the importance weights and the so-called *values*. Then, the architecture includes fully-connected sub-layers, residual connections (He et al., 2016), and layer normalization (LN) (Ba et al., 2016), as illustrated in Fig. 26.

RANK COLLAPSE. In the absence of residual connections, Dong et al., 2021 proved that at initialization the **rank of the sequence representation** collapses doubly exponentially with depth, and both layer normalization and fully connected layers can only partially alleviate the speed of degeneracy. Under *rank collapse*, the model does not distinguish between representations of different tokens, which are perfectly aligned in feature space at initialization. However, the precise implications of rank collapse in Transformers are not fully understood.

In this chapter, we show that a high alignment of the tokens' representations at initialization — corresponding to rank collapse in the extreme case of perfect alignment — **affects training by causing vanishingly small gradients of the queries and keys'**. This problem severely diminishes the capabilities of the model to learn meaningful attention weights and is further exacerbated in very deep networks, where the rank deficiency — and hence the vanishing gradient problem of the queries and keys — affects several layers (see Fig. 27). In order to shed light on this problem, we take inspiration from the flourishing literature on signal

propagation in random networks and start our analysis by computing the expected gradients of an attention layer with respect to the queries, keys, and values, which leads to Theorem 3.3.1 on the vanishing gradients for the queries and keys. From here, we pursue **two different directions**.

We first investigate under which **conditions rank collapse can be avoided** by studying the evolution of the input sequence in a Transformer at initialization. Our theory reveals that a **depth-dependent scaling of the residual branches**, beyond stabilizing the norm of the activations at initialization, also approximately preserves the cosine of the angle between tokens, hence stabilizing the rank of the propagating sequence. We show that this holds even in the **infinite-depth limit**.

Secondly, we illustrate that there are factors, other than the average tokens' correlation, that affect differently the gradient norm of the queries and keys compared to the values. The **propagating sequence's squared norm has a linear dependence in the values, while a cubic one in the queries and keys**, justifying the use of layer normalization and Adam. We also highlight a different dependence on the embedding dimension and the length of the input sequence, implying that the **gradient norm of a subset of parameters can potentially be of different orders of magnitude**, as empirically hinted by previous works (Liu et al., 2020a). Our analysis brings to light fundamental issues in the signal propagation in Transformers, opening the way for new, well-founded, and motivated approaches to improve optimization in these models.

In this chapter, we are going to first study signal propagation and rank collapse (Sections 3.3.2), and then **discuss performance of Adam and SGD** in Section 3.3.3.

### 3.3.1 *Notation and Background on Rank Collapse*

The Transformer architecture stacks $L$ attention blocks, as shown in Fig. 26. Layer normalization is usually applied token-wise either after the residual connections or to the inputs of the self-attention and position-wise feed-forward sub-layers, leading to the POST-LN (Vaswani et al., 2017) and PRE-LN (Wang et al., 2019; Xiong et al., 2020) variants respectively.

Formally, given an input sequence $\mathbf{X} \in \mathbb{R}^{n \times d_v}$, with $n$ tokens of dimension $d_v$, the single-head unmasked scaled dot-product self-attention[5] is defined as:

$$\mathbf{S}^\ell := \mathbf{A}^\ell \mathbf{X}^\ell \mathbf{W}^V, \tag{66}$$

with

$$\mathbf{A}^\ell = \text{softmax}\left( \frac{1}{\sqrt{d_k}} \mathbf{X}^\ell \mathbf{W}^Q \left( \mathbf{X}^\ell \mathbf{W}^K \right)^\top \right), \tag{67}$$

where the softmax function is applied independently across each row, and the superscript $\ell$ indexes the $\ell$-th layer.

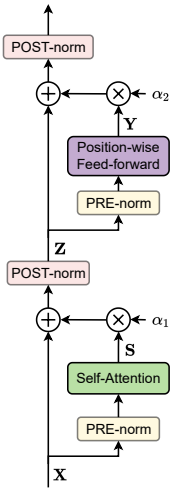


**Figure 26:** *A single Transformer block.*

The matrices $\mathbf{W}^Q, \mathbf{W}^K \in \mathbb{R}^{d_v \times d_k}$ and $\mathbf{W}^V \in \mathbb{R}^{d_v \times d_v}$ are learnable parameters, and each layer is initialized with an independent set of weights. In the literature, the matrices $\mathbf{X}^\ell \mathbf{W}^Q, \mathbf{X}^\ell \mathbf{W}^K, \mathbf{X}^\ell \mathbf{W}^V$ are referred to as queries, keys and values, respectively. The complete Transformer block, in the absence of layer normalization, can be written recursively as:

5 Our analysis also easily generalizes to the case of cross-attention.

$$\mathbf{Z}^\ell = \alpha_1 \mathbf{S}^\ell + \mathbf{X}^\ell \tag{68}$$

$$\mathbf{Y}^\ell = \sigma(\mathbf{Z}^\ell \mathbf{W}^{F_1})\mathbf{W}^{F_2} \tag{69}$$

$$\mathbf{X}^{\ell+1} = \alpha_2 \mathbf{Y}^\ell + \mathbf{Z}^\ell, \tag{70}$$

where the introduced $\alpha_1, \alpha_2$ parameters indicate the strength of the residual block, $\mathbf{W}^{F_1}, \mathbf{W}^{F_2} \in \mathbb{R}^{d_v \times d_v}$ [6] are matrices of learnable parameters; we set $\mathbf{X}^0 := \mathbf{X}$, and $\sigma : \mathbb{R} \to \mathbb{R}$ is an activation function. In our case, $\sigma$ is the ReLU function, but we relax this assumption to the linear activation from Section 3.3.2.2 on. At initialization, each weight is sampled independently from a distribution with zero-mean and variance $\sigma_v^2 = \frac{1}{d_v}$ for the values and feedforward weights[7], and $\sigma_k^2 = \frac{1}{d_k}$ for the queries and keys. This is the standard "Xavier" (Glorot and Bengio, 2010) or "He" (He et al., 2015) initialization, commonly used in deep learning.

RANK COLLAPSE IN TRANSFORMERS. Dong et al., 2021 proved that **when the residual branches are omitted**, the matrix of the tokens' representations $\mathbf{X}^\ell$ **converges to a rank-1 matrix** in which all the representations are the same and equal to a vector $\mathbf{x} \in \mathbb{R}^{d_v}$, i.e. $\mathbf{X}^\ell \to \mathbf{1}_n \mathbf{x}^\top$, where $\mathbf{1}_{d_v}$ is the vector with all ones in $\mathbb{R}^{d_v}$. One of our main contributions is to provide an explanation of **how rank collapse affects the gradients** of a Transformer at initialization.

VANISHING GRADIENT PROBLEM. Traditionally considered one of the core issues that prevent successful training, the vanishing gradient problem has a long and rich history that dates back to before the popularization of deep learning (Hochreiter, 1991; Bengio et al., 1994). In its essence, given a loss function $\mathcal{L} : \mathbb{R}^{n \times d_v} \to \mathbb{R}$, vanishing gradients occur when the norm of the gradient of the loss $\mathcal{L}$ with respect to the parameters of the network $\mathbf{W}$ — which we indicate as $\left\| \frac{\partial \mathcal{L}}{\partial \mathbf{W}} \right\|$ — is too small to provide enough backpropagating signal, thus hindering gradient-based optimization methods. Despite extensive research toward understanding and

6 In practice, one commonly uses $\mathbf{W}^{F_1} \in \mathbb{R}^{d_v \times d_F}$, $\mathbf{W}^{F_2} \in \mathbb{R}^{d_F \times d_v}$ where $d_F = \gamma d_v$, with $\gamma \in \{2, 4, 8\}$. Our results then hold up to a constant factor that depends on $\gamma$.

7 One should explicitly write the layer dependence $\mathbf{W}^{Q,\ell}, \mathbf{W}^{K,\ell}, \mathbf{W}^{V,\ell}, \mathbf{W}^{F_1,\ell}, \mathbf{W}^{F_2,\ell}$. We at times suppress the $\ell$ index to improve readability. In case $\sigma$ is the ReLU function, we set $\mathbf{W}^{F,\ell}$ to have variance $\frac{2}{d_v}$.

overcoming the problem (Glorot and Bengio, 2010; He et al., 2015; Zhang et al., 2019b), a formal explanation of its role in relatively new architectures such as Transformers is largely missing in the literature, with a few exceptions (Xiong et al., 2020; Wang et al., 2022a; Huang et al., 2020). In our work (Section 3.3.2.1), we show how **vanishing gradient occurs in conjunction with the rank collapse issue** identified by Dong et al., 2021.

SIGNAL PROPAGATION IN RANDOM NETWORKS AT INITIALIZATION. After addressing the question on the effects of rank collapse, we take a step back and rigorously analyze **its causes** by looking at how the properties of the input sequence $\mathbf{X}$ are lost/preserved as it propagates through a randomly initialized Transformer. More specifically, we focus on two aspects of the propagating sequence: the expected Frobenius norm $\mathbb{E}\left\|\mathbf{X}^{\ell}\right\|^2$ and the **expected inner product** between different tokens $\mathbb{E}\langle \mathbf{X}_k, \mathbf{X}'_k \rangle$, with $k \neq k'$. The former is linked to studies on the initialization of neural networks at the edge of chaos (Poole et al., 2016; Schoenholz et al., 2016), and vanishing/exploding gradients (Hanin, 2018). The latter quantity describes how the geometry of the feature space changes after applying a Transformer block, and is related to the concept of dynamical isometry (Saxe et al., 2013). To understand the evolution of the inner product, we analyze the following measure of correlation (Nachum et al., 2021):

$$\rho^{\ell}_{kk'} := \frac{\mathbb{E}\langle \mathbf{X}^{\ell}_k, \mathbf{X}^{\ell}_{k'} \rangle}{\sqrt{\mathbb{E}\left\|\mathbf{X}^{\ell}_k\right\|^2 \mathbb{E}\left\|\mathbf{X}^{\ell}_{k'}\right\|^2}}. \tag{71}$$

Note that $\rho^{\ell}_{kk'} = 1$ if and only if the $k$-th and $k'$-th tokens are perfectly aligned ($\cos\theta_{kk'} = 1$). We stress that in our case — differently from the aforementioned works — instead of analyzing the relationship between two different data points, **we study the relationship between tokens** of the same sequence.

### 3.3.2 *Analysis of Signal Propagation*

The goal of this section is twofold. In Lemma 3.3.1 and Theorem 3.3.1, we provide an explanation of the possible cause of **vanishingly small gradients for queries and keys at initialization**, namely the high correlations

between the tokens representations in $\mathbf{X}^\ell$ as the **depth increases**. In Section 3.3.2.2, we show that the problem can be mitigated with an appropriate choice of the residual branch parameters $\alpha_1$ and $\alpha_2$ that inversely scales with the depth of the network $L$. Under the proposed scaling, the correlations are well behaved even in the infinite depth limit. Finally, in Section 3.3.2.3 we analyze the scaling of the gradients with respect to other network's parameters, and in 3.3.3 we draw some connections between our findings and optimization of Transformers.

#### 3.3.2.1 *Vanishing Gradients for Queries and Keys under Rank Collapse*

To investigate the problem of vanishing gradients in the attention layers, we make use of the framework of matrix calculus (Magnus and Neudecker, 2019; Singh et al., 2021). In particular, we compare the expected Frobenius norm of the gradient of a self-attention layer with respect to its parameters $\mathbb{E}\left\|\frac{\partial \mathbf{S}^\ell}{\partial \mathbf{W}}\right\|_F^2$, where here $\mathbf{W}$ indicates one of the keys, queries or values weight matrices. Due to the well-known difficulty of computing expectations of the softmax (Daunizeau, 2017; Shekhovtsov and Flach, 2018), throughout this manuscript, we make the simplifying assumption that the **softmax** output is the **uniform distribution at initialization**, i.e. the $n \times n$ matrix containing $\frac{1}{n}$ in each entry.

**Assumption 3.3.1** (Uniform attention)**.** *We assume that* $\mathbf{A}^\ell = \frac{1}{n}\mathbf{1}_{n\times n}$,

where $\mathbf{1}_{n\times n}$ is the matrix with all entries equal to 1. Crucially, in Appendix C.2, we formally show that **this assumption holds almost surely** in the limit $d_k \to \infty$. There, we also experimentally show that even in the more realistic case where $d_k = d_v \approx 512$, the empirical simulations provide a surprisingly faithful approximation of the theoretical insights presented in this chapter.
We define the mean token $\bar{\mathbf{x}}^\ell$ through its components $\bar{\mathbf{x}}_i^\ell = \frac{1}{n}\sum_{k=1}^n \mathbf{X}_{ki}^\ell$, $i \in [d_v]$. In the following theorem, we compute the expected gradients of an attention layer at initialization and set the basis for our following analysis. We provide the results only for the queries, as the case for the keys is analogous.

**Lemma 3.3.1** (Gradient Norms). *Let $\mathbf{X}^\ell$ be the representations of the input sequence at the $\ell$-th layer. Under the uniform-attention assumption, we have*

$$\mathbb{E}\left\|\frac{\partial \mathbf{S}^\ell}{\partial \mathbf{W}^{V,\ell}}\right\|_F^2 = d_v n \mathbb{E}\|\bar{\mathbf{x}}^\ell\|^2 \,; \tag{72}$$

$$\mathbb{E}\left\|\frac{\partial \mathbf{S}^\ell}{\partial \mathbf{W}^{Q,\ell}}\right\|_F^2 = \frac{\sigma_v^2\sigma_k^2 d_v}{n^2}\cdot \mathbb{E}\left[\|\mathbf{X}^\ell\|_F^2 \cdot \|(\mathbf{X}^\ell)^\top \mathbf{X}^\ell - n\bar{\mathbf{x}}^\ell(\bar{\mathbf{x}}^\ell)^\top\|_F^2\right] \,; \tag{73}$$

$$\mathbb{E}\left\|\frac{\partial \mathbf{S}^\ell}{\partial \mathbf{X}^\ell}\right\|_F^2 \leq \frac{8\sigma_q^2\sigma_k^2\sigma_v^2 d_k d_v}{n}\cdot \mathbb{E}\left\|(\mathbf{X}^\ell)^\top \mathbf{X}^\ell - n\bar{\mathbf{x}}^\ell(\bar{\mathbf{x}}^\ell)^\top\right\|_F^2 + 2d_v^2\sigma_v^2 \,. \tag{74}$$

We defer the precise study of the scaling of these quantities as a function of $n$ and $d_v, d_k$, to Section 3.3.2.3. At this stage, it is crucial to note that $\frac{1}{n}(\mathbf{X}^\ell)^\top \mathbf{X}^\ell - \bar{\mathbf{x}}^\ell(\bar{\mathbf{x}}^\ell)^\top$ is the centered empirical covariance matrix of the tokens' representations. It is easy to see that if $\mathbf{X}^\ell$ is a rank-1 matrix, then all the rows of $\mathbf{X}^\ell$ are proportional to a fixed $d_v$-dimensional vector, and the empirical covariance matrix has all zero entries. Introducing a differentiable loss function $\mathcal{L} : \mathbb{R}^{n\times d_v} \to \mathbb{R}$, we make the statement on vanishing gradients more formal in the following theorem:

**Theorem 3.3.1** (Vanishing gradients under rank collapse)**.** *Suppose that the uniform-attention assumption holds. If additionally $\mathbf{X}^\ell$ for any $l \in [L]$ has rank-1, and there exists a vector $\mathbf{x} \in \mathbb{R}^d$ such that $\mathbf{X}^\ell = \mathbf{1}_n \mathbf{x}^T$, then:*

$$\mathbb{E}\left\|\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{Q,\ell}}\right\|_F^2 = 0, \quad \mathbb{E}\left\|\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{K,\ell}}\right\|_F^2 = 0, \quad \mathbb{E}\left\|\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{V,\ell}}\right\|_F^2 \gg 0, \tag{75}$$

*where the expectation is taken over the weight matrices. This implies that these quantities are vanishing almost surely, due to the non-negativeness of the norm.*

We provide in the appendix of this thesis (Appendix C.2) a numerical verification of Lemma 3.3.1 and a sketch for the proof of Theorem 3.3.1. For a complete discussion, we invite the reader to check Noci et al., 2022. In light of Theorem 3.3.1, we can conclude that the issue of **rank collapse** originally identified in Dong et al., 2021 **corresponds to an initialization in a region of vanishing gradient signal in the subspace of parameters identified by the queries and keys**. How can this affect training? One may argue that if rank collapse does not happen in the very first layer,

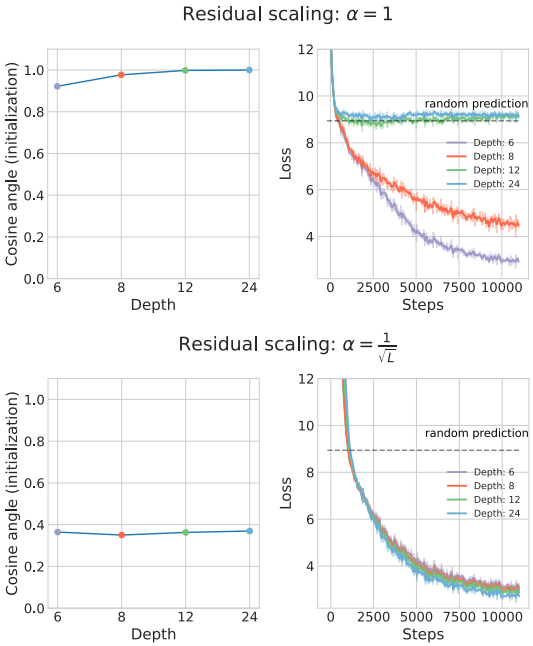


**Figure 27:** *Evolution of the cosine of the angle between tokens for training POST-LN Transformers of increasing depth, with the Adam optimizer, for the IWSLT'14 De-En translation task. Unless adequate residual scaling is used at initialization, increasing depth leads to an increase in the tokens' alignment at initialization, which can inhibit training.*

then the corresponding gradients are non-zero and the rank of the subsequent layers — affected by rank collapse — can be increased with the first few steps of gradient descent. In practice, we show empirically in Fig. 27 that **escaping this pathological landscape is harder in deeper nets** where rank collapse persists across several layers.

In Section 3.3.3 we discuss the **implications** of the results above **for training with adaptive methods** such as Adam (Kingma and Ba, 2014). In the next section, we show how scaling residual branches helps to avoid rank collapse.

#### 3.3.2.2 *Signal Propagation and Importance of Scaling the Residual Branches*

We now turn our attention to the study of the **influence of skip connections** in Transformers. Dong et al., 2021 showed that simply adding this architectural trick prevents rank collapse. Somewhat surprisingly, we show that while the claim holds for any finite depth, the average angle between different tokens quickly increases with just a few layers, and as $L \to \infty$ a Transformer **can still lose rank** unless the residual branches are adequately initialized. As Dong et al., 2021 showed that **layer normalization does not avoid rank collapse**, we omit it in our analysis. Firstly, we introduce two lemmas on the propagation of inner products (Lemma 3.3.2) and the norm (Lemma 3.3.3) of the tokens' representations.

**Lemma 3.3.2** (Propagation of inner products)**.** *Let* $C(\mathbf{X}^\ell) = \sum_{k,k'} \langle \mathbf{X}^\ell_k, \mathbf{X}^\ell_{k'} \rangle$ *and* $\mathbf{X}$ *the input sequence. Under the Assumption 3.3.1 and if* $\sigma$ *is the linear activation function, we have that:*

$$\mathbb{E}\left[C(\mathbf{X}^L)\right] = (\alpha_2^2 + 1)^L (\alpha_1^2 + 1)^L C(\mathbf{X}). \tag{76}$$

*hence, under the depth scaling for the residual block parameters* $\alpha_1^2 = \frac{\tilde{\alpha}_1}{L}, \alpha_2^2 = \frac{\tilde{\alpha}_2}{L}$ *with* $\tilde{\alpha}_1, \tilde{\alpha}_2 \in \mathbb{R}$ *independent of* $L$*, we have that:*

$$\lim_{L\to\infty} \mathbb{E}[C(\mathbf{X}^L)] = e^{\tilde{\alpha}_1 + \tilde{\alpha}_2} C(\mathbf{X}). \tag{77}$$

Note that $C(\mathbf{X}^\ell) = n^2 \left\|\bar{\mathbf{x}}^\ell\right\|^2$. The lemma on the propagation of the norm is slightly more involved:

**Lemma 3.3.3** (Propagation of the norm)**.** *Let* $\mathbf{X}^L$ *be the representations of the input sequence at the final layer. Under the assumptions of Lemma 3.3.2, we have that:*

$$\mathbb{E}\left\|\mathbf{X}^L\right\|_F^2 = n(\alpha_2^2 + 1)^L \alpha_1^2 \sum_{k=0}^{L-1} (\alpha_1^2 + 1)^k \left\|\bar{\mathbf{x}}\right\|^2 + (\alpha_2^2 + 1)^L ||\mathbf{X}||_F^2, \tag{78}$$

*hence, under the depth scaling for the residual block parameters* $\alpha_1^2 = \frac{\tilde{\alpha}_1}{L}, \alpha_2^2 = \frac{\tilde{\alpha}_2}{L}$ *with* $\tilde{\alpha_1}, \tilde{\alpha_2} \in \mathbb{R}$ *independent of* $L$*, we have that:*

$$\lim_{L\to\infty} \mathbb{E}\left\|\mathbf{X}^L\right\|_F^2 = n e^{\tilde{\alpha}_2}(e^{\tilde{\alpha}_1} - 1) \left\|\bar{\mathbf{x}}\right\|^2 + e^{\tilde{\alpha}_2} ||\mathbf{X}||_F^2. \tag{79}$$

The previous Lemma provides theoretical justification that scaling the residual branches by setting the alpha parameters to be $\mathcal{O}(1/\sqrt{L})$ allows both the norm of the propagating input and the inner products between different tokens to be approximately preserved. Hence, the information contained in the input is not lost, even in the infinite depth limit.

RESIDUAL SCALING PRESERVES CORRELATIONS. We now prove that **without the depth-dependent residual scaling** (i.e. with $\alpha_1 = \alpha_2 = 1$) the **correlation between the tokens quickly increases**, and reaches perfect alignment in the infinite depth limit. More specifically, our argument shows that in this limit, the correlation between different tokens $\rho^{\ell}_{k,k'}$ as in Eq. (71) converges to 1, implying rank collapse. Furthermore, we show how setting the residual parameters $\alpha_1$ and $\alpha_2$ as dictated by Theorem 3.3.3, ensures that the correlation measure is dependent on the input in a non-trivial way even at infinite depth. To this end, we introduce the average correlation at layer $\ell$:

$$\rho^{\ell} = \frac{1}{n(n-1)} \sum_{k \neq k'} \rho^{\ell}_{kk'}. \tag{80}$$

Note that $\rho^{\ell} = 1$ if and only if every pair of tokens is perfectly aligned. We are now ready to formalize the influence of the $1/\sqrt{L}$-scaling on the correlation between tokens' representations by stating Theorem 3.3.2.

**Theorem 3.3.2** (Expected cosine similarity). *Let the input tokens have the same norm, i.e.* $\|\mathbf{X}_k\| = \|\mathbf{x}\| \ \ \forall k \in [n]$ *for some* $\mathbf{x} \in \mathbb{R}^{d_v}$. *Under the depth scaling for the residual block parameters* $\alpha_1^2 = \frac{\tilde{\alpha}_1}{L}, \alpha_2^2 = \frac{\tilde{\alpha}_2}{L}$ *with* $\tilde{\alpha}_1, \tilde{\alpha}_2 \in \mathbb{R}$ *independent of L, we have that:*

$$\lim_{L\to\infty} \rho^{\ell} = \frac{n e^{\tilde{\alpha}_1} C(\mathbf{X})}{(n-1)[(e^{\tilde{\alpha}_1} - 1)C(\mathbf{X}) + n\,\|\mathbf{X}\|_F^2]} - \frac{1}{n-1}. \tag{81}$$

*On the other hand, if* $\alpha_1, \alpha_2 \neq 0$ *are some constants independent of L, we have that:*

$$\lim_{L\to\infty} \rho^{\ell} = 1. \tag{82}$$

The proof can be found in the Appendix of (Noci et al., 2022). Note that under the $1/\sqrt{L}$-scaling, the correlation term is one if and only if $C(\mathbf{X}) = n\,\|\mathbf{X}\|^2$, which holds in the degenerate case where all the input

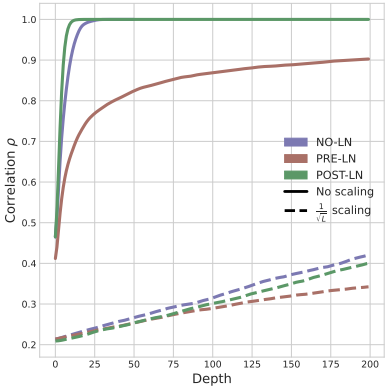


**Figure 28:** *Evolution of Correlation in Transformers with (dashed lines) and without (solid lines) $1/\sqrt{L}$-scaling for PRE-LN, POST-LN and without layer normalization (No-LN).*

tokens are perfectly aligned. In the appendix of Noci et al., 2022, we give precise formulas for the expected correlations at any depth, showing that $\rho^{\ell}$ reaches values close to one even for relatively shallow networks when the $1/\sqrt{L}$-scaling is not adopted (see also Fig. 29 (left)). Additionally, in Fig. 28, we empirically show that in the presence of the $1/\sqrt{L}$-scaling, layer normalization (either PRE or POST) does not significantly affect the evolution of the correlations. On the other hand, without the residual scaling, PRE-LN seems to alleviate the rate of increase of $\rho^{\ell}_{kk'}$. It is intriguing that most deep Transformer models use this configuration (Brown et al., 2020).

It is also worth noting that the $1/\sqrt{L}$ scaling for the residual branches has been previously studied in the context of stabilization of residual networks (see Section 3.3.4), here we extend these results to Transformers and provide new insights on its role in the context of rank preservation. Finally, note that by setting $\tilde{\alpha}_1, \tilde{\alpha}_2 = 0$, we recover the so called "ReZero" initialization (Bachlechner et al., 2021). In this context, the $1/\sqrt{L}$ scaling extends this framework as it allows for wider range of values for $\tilde{\alpha}_1, \tilde{\alpha}_2$ while still guaranteeing stability.

RELU EXTENSION. We mention here that extending these results from the linear activation to the ReLU case is known to be a hard problem, due to the technical difficulty of propagating the inner products across ReLU layers that are shared among the tokens (this is the case in the position-wise feed-forward layers in Transformers). Exact formulas can be found only in the case of one ReLU layer with Gaussian inputs in Cho and Saul, 2009. However, in the context of rank collapse analyzed here, the linear activation function provides a bound on the correlation with respect to the ReLU case. In fact, **correlations** are exactly preserved in expectation in the linear case, but **increase in the ReLU case** (for instance, see the contraction argument in Nachum et al., 2021 below Equation (2)). Hence, the perfect alignment (a.k.a rank collapse) that affects the linear case affects the ReLU case as well (in which case the rank collapses even faster with depth, as we show in the appendix of Noci et al., 2022).

#### 3.3.2.3 *Dependence on the Angle between Tokens and the Input Norm*

In this section, we drop the superscript $\ell$ as it is obvious from context and assume for simplicity that $d_k = d_v$. To gain a better intuition on the factors that affect the gradients and provide additional insights, we study the case in which **every pair of distinct tokens are zero-mean Gaussian** random variables, correlated in the same way, i.e $\rho^{\ell}_{ii'} = \rho$ for $i \neq i'$ or more precisely

$$\mathbb{E}\left[\mathbf{X}_{i,j}\mathbf{X}_{i',j'}\right] = \begin{cases} 0 & j \neq j' \text{ (independent dimensions)} \\ \sigma_x^2 & i = i', j = j' \\ \rho\sigma_x^2 & i \neq i', j = j' \end{cases} . \tag{83}$$

To see that this equation satisfies our definition of the correlation metric, note that $\mathbb{E}[\|\mathbf{X}_i\|^2] = d\sigma_x^2$ and $\mathbb{E}\langle \mathbf{X}_i, \mathbf{X}_{i'}\rangle = d\sigma_x^2\rho$, for $i \neq i'$. Then, the expected norm of the gradients for the values (Eq. (72)) simplifies to

$$\mathbb{E}\left\| \frac{\partial \mathbf{S}}{\partial \mathbf{W}^V} \right\|_F^2 = \sigma_x^2 d^2 \left(1 + \rho(n-1)\right). \tag{84}$$

By making the additional assumption that the norm and the correlation propagate independently, the respective norm for the queries (Eq. (73)) — and symmetrically the keys — reduces to:

$$\mathbb{E}\left\|\frac{\partial \mathbf{S}}{\partial \mathbf{W}^Q}\right\|_F^2 = \sigma_x^6 \frac{(n-1)}{n}(1-\rho)^2 d(n+d). \tag{85}$$

In the appendix of Noci et al., 2022 we provide a rigorous proof, that relies on Isserlis theorem (Isserlis, 1918) to compute higher-order moments. The above expressions reveal the different dependencies on four main actors, that we inspect separately here. The **gradients of the queries depend via a cubic function on the variance of the input**, $\sigma_x^2$, compared to a **linear for the values**. This provides an additional interpretation of the successful use of layer normalization, as in Xiong et al., 2020, either in the POST-LN or PRE-LN format, that standardizes the input variance $\sigma_x^2$ to the value 1.
Next, we emphasize the dependence on the correlation between the tokens, also illustrated in Fig. 29. Importantly, note how the **queries/keys have opposite monotonic functional dependence** with respect to $\rho$ **compared to the values**. As revealed by Theorem 3.3.2 and Fig. 29 (center), inappropriate scaling of the residual branches can already lead to this phenomenon even in a relatively shallow network. Finally, Eq. (84) and (85) reveal a different scaling in terms of the *embedding size* $d$ and the *sequence length* $n$ due to the self-attention operation itself. We hope that the identification of the different dependencies in the gradients of the parameters will inspire a new line of works aimed at solving some of the difficulties in training Transformers.

### 3.3.3 *Why Does Adam Work on Transformers?*

The existence of the discrepancy in the magnitude of the gradients with respect to the weights $\mathbf{W}^Q, \mathbf{W}^K$ and $\mathbf{W}^V$, might explain the success of adaptive optimization algorithms, as shown in Fig. 30, where we plot the **effective learning rate computed by Adam** (Kingma and Ba, 2014) in a toy encoder task: given a sequence of 20 numbers in the range $0-9$, we predict the same tokens in the inverted order. We use an embedding layer of size 16, initializes with variance 1, and sinusoidal positional encodings to initially embed the input. We use a 5-layer POST-LN Transformer

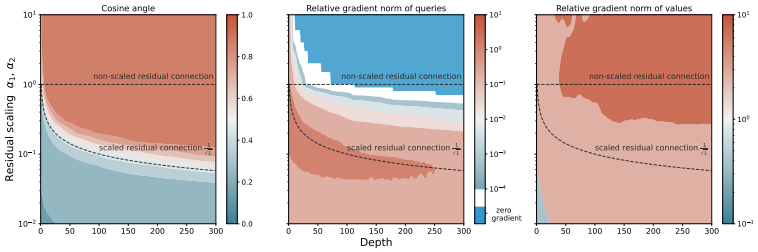


**Figure 29:** *Effect of the residual scaling to the norm of the gradients of the network at initialization with respect to some loss. From left to right: (a) the cosine of the angle between tokens increases with depth. Note how larger values of $\alpha_1, \alpha_2$ imply a faster token alignment with depth (Theorem 3.3.2). Subplots (b) and (c) show the gradients of the queries-keys and values parameters respectively by increasing depth, compared to the norms of the first layer. Gradients for the queries-keys diminish with depth, while the opposite happens for the values. We use POST-LN to disentangle the effect of the variance of the input.*

encoder model, with a single head attention operation and a two-layer feed-forward layer with a ReLU nonlinearity. We use residual scaling, in this case, equal to $\alpha_1 = \alpha_2 = 1$ — leading to moderate rank collapse and hence vanishing gradients. We train using Adam with betas parameters $(0.9, 0.999)$, learning rate 0.01 and weight decay 0. Notice that the **effective learning rate is increasingly larger (with depth) for the queries compared to the values** – as postulated by our theory – and this **gap is remarkably constant throughout training**.

Hence, we conjecture that the **success of adaptive methods** in Transformers' training can be partially explained by the **need to fix this unbalanced gradient's magnitude**. To test this hypothesis, we propose a simple architectural modification, an inverse **temperature scaling** $\tau \in \mathbb{R}$ inside the softmax:

$$\mathbf{S}_{\tau}^{\ell} := \text{softmax}\left(\frac{\tau}{\sqrt{d_k}}\mathbf{X}^{\ell}\mathbf{W}^{Q}\left(\mathbf{X}^{\ell}\mathbf{W}^{K}\right)^{\top}\right)\mathbf{X}^{\ell}\mathbf{W}^{V}. \tag{86}$$

A direct consequence of our analysis is that $\tau$ allows **controlling the magnitude of the gradients** for the queries and keys' parameters. In the appendix of Noci et al., 2022, we detail how one can choose $\tau$ such that

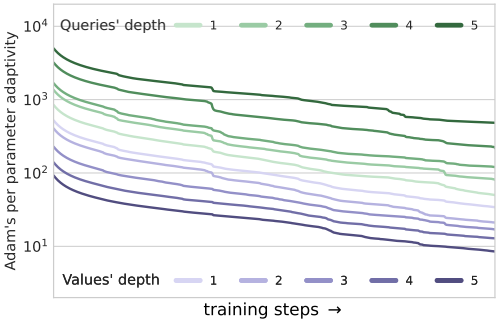


**Figure 30:** *Adaptive learning rates computed by Adam in Transformers.*

the magnitude of the gradients as derived in Equation (84) and (85) is approximately matched at initialization:

$$\tau^2 \approx \frac{dn(1+\rho(n-1))}{\sigma_x^4(1-\rho)^2(n+d)(n-1)}. \tag{87}$$

We evaluate our proposal, consisting of residual scaling and the aforementioned inverse temperature parameters, on the widely used IWSLT14 German-to-English (De-En) benchmark translation task. All details regarding the experimental setup and the choice of inverse temperature used are provided in the Appendix. We train a Transformer encoder-decoder of varying depth with stochastic gradient descent (SGD), after removing all normalization layers and adequately initializing the residual connections. For our **training with SGD**, we avoid using any learning rate warm-up, as commonly done for Adam, and instead use a step-scheduler to decrease the learning rate at 40% and 80% of training. We compare against the following methods that make use of Adam; POST-LN and PRE-LN refer to the aforementioned alternatives to apply layer normalization. We also compare against other successful techniques that rely on specific initializations to avoid layer normalization, such as ReZero (Bachlechner et al., 2021) and T-Fixup (Zhang et al., 2019b). We report the average BLEU score (Papineni et al., 2002) across 5 runs in Fig. 31 and Table 2.

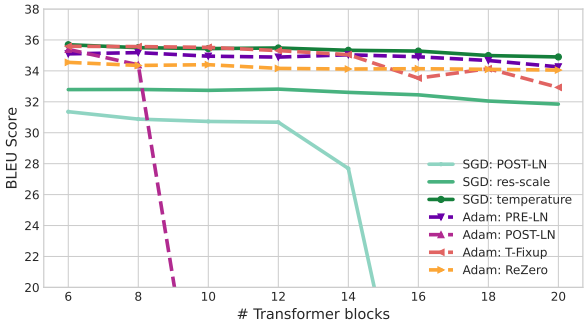


**Figure 31:** *BLEU scores by increasing the number of transformers blocks. 'X' Transformer blocks implies in total 'X' encoder self-attention, 'X' decoder self-attention, and 'X' decoder cross-attention layers.*

Our proposed method considerably improves training with SGD, keeping up and in some cases surpassing any results achieved by the Adam optimizer. We are also able to train deeper networks without the use of layer normalization. We leave for future work to further investigate modifications or alternatives to the self-attention operation.

| Method (6L-Encoder / 6L-Decoder) | BLEU ↑ |
|---|---|
| SGD POST-LN | 31.36 |
| SGD res-scale | 32.79 |
| SGD temperature | **35.69** |
| Adam POST-LN (Vaswani et al., 2017) | 35.39 |
| Adam PRE-LN (Vaswani et al., 2017) | 35.10 |
| ReZero (Bachlechner et al., 2021) | 34.55 |
| T-Fixup (Zhang et al., 2019b) | 35.59 |

**Table 2:** *BLEU scores for the IWSLT14 German-to-English translation task. SGD res-scale refers to the training of SGD without layer normalization and initialization of the residual scaling* $a_1 = a_2 = \frac{1}{\sqrt{L}}$. *SGD temperature additionally employs an inverse temperature inside the softmax.*

### 3.3.4 *Discussion of Related Works*

Our work builds upon the rich literature on forward and backward signal propagation in random neural networks (Poole et al., 2016; Schoenholz et al., 2016; Xiao et al., 2018; Pennington et al., 2017; Orvieto et al., 2022b; Noci et al., 2021). The $1/\sqrt{L}$ scaling scheme has been investigated in the literature for the stabilization of residual networks (Hanin and Rolnick, 2018; Arpit et al., 2019; Hayou et al., 2021).

Our work draws inspiration from a series of recent works studying the rank of the representations of random feed-forward neural networks at initialization (Daneshmand et al., 2020). In the context of Transformers, Dong et al., 2021 has recently identified the rank collapse issue object of study of the present work. Thanks to our analysis of the backward pass, we are able to demonstrate that rank collapse in Transformer architectures leads to vanishingly small gradients of queries and keys, thereby preventing effective training and allowing us to complete the analysis of Dong et al., 2021.

Among the architectural components in Transformers, layer normalization is, arguably, one of the most important – and debated – ones (Chen et al., 2018a; Wang et al., 2019; Nguyen et al., 2010; Xiong et al., 2020). In the original architecture (Vaswani et al., 2017), layer normalization is used to stabilize the forward pass by reducing the variance of the inputs to the following sublayer. Our analysis of the forward pass shows that its inclusion is not strictly necessary for the purpose of controlling the norm of the representations. For a theoretical analysis of signal propagation in the presence of layer norm, we refer the reader to Xiong et al., 2020.

Additionally, our theoretical study of the backward pass provides a rigorous explanation of the empirically observed discrepancy between the magnitude of the gradients of the queries and the values, which Liu et al., 2020a hypothesize to be one of the causes of the success of adaptive methods in training Transformers (Liu et al., 2019b; Zhang et al., 2020b; Huang et al., 2020).

Finally, properly rescaled residual connections have been found to be beneficial for training Transformers by a number of recent research works (Zhang et al., 2019b; Bachlechner et al., 2021; Wang et al., 2022a). However, none of these studies characterize the impact of skip connections on rank propagation, while our analysis suggests a theoretically-grounded way to stabilize it.

# 4

# DESIGNING NEW ADAPTIVE METHODS

*"As a rule, when many options are available, man's actions are guided by the need to choose the best possible way. Human activity, indeed, implicates solving (consciously or unconsciously) optimization problems. Moreover, many laws of nature are of a variational character, even if it is inappropriate in this case to speak of the existence of a purpose. "*

– Boris Polyak.

In Chapter 3, we discussed adaptive stepsizes in the context of deep learning, showing their effectiveness in training complex neural networks such as transformers. While our discussion involved heavy calculations, the analysis of adaptive stepsizes was often empirical or **driven by toy examples**. This necessity is due to the **complexity of dealing with the Adam update**, a known issue in the literature (Dauphin et al., 2015).
Treasuring the lessons learned from the last chapter, in the next section we revisit the foundations of adaptive methods and show interesting **new approaches** for the **design of novel** — simpler — **adaptive methods** with solid theoretical guarantees and strong empirical performance.

To do so, we go back to the stochastic optimization problem formulation:

$$\min_{x\in\mathbb{R}^d}\left[f(x)=\frac{1}{n}\sum_{i=1}^{n}f_i(x)\right], \tag{88}$$

where each $f_i$ is lower bounded. We denote by $\mathcal{X}^*$ the non-empty set of optimal points $x^*$ of equation (88). We set $f^* := \min_{x\in\mathbb{R}^d} f(x)$, and $f_i^* := \min_{x\in\mathbb{R}^d} f_i(x)$.
The most basic algorithm, Stochastic Gradient Descent (SGD), is

$$x_{k+1} = x_k - \gamma_k \nabla f_{\mathcal{S}_k}(x_k) \tag{SGD}$$

where $\gamma_k > 0$ is the stepsize at iteration $k$, $\mathcal{S}_k \subseteq [n]$ a random subset of datapoints (minibatch) with cardinality $B$ sampled independently at each iteration $k$, and $\nabla f_{\mathcal{S}_k}(x) := \frac{1}{B}\sum_{i\in\mathcal{S}_k}\nabla f_i(x)$ is the minibatch gradient.

THE IDEAL ADAPTIVE STEPSIZES. As it is clear at this point of the thesis, a careful choice of $\gamma_k$ is crucial for successful training (Bottou et al., 2018; Goodfellow et al., 2016). Let us **refresh the basics**. The simplest option is to pick $\gamma_k$ to be constant, with its value inversely proportional to the **Lipschitz constant** of the gradient. While this choice yields fast convergence to the neighborhood of a minimizer, two main problems arise: (a) the optimal $\gamma$ depends on (often unknown) problem parameters — hence often requires heavy tuning ; and (b) it cannot be guaranteed that $\mathcal{X}^*$ is reached in the limit (Ghadimi and Lan, 2013; Gower et al., 2019; Gower et al., 2021). A simple fix for the last problem is to allow polynomially decreasing stepsizes (Nemirovski et al., 2009): this choice leads to convergence to $\mathcal{X}^*$, but hurts the overall algorithm speed. Ideally, a **theoretically grounded adaptive method** should yield **fast convergence** to $\mathcal{X}^*$ **without knowledge of problem dependent parameters**, such as the gradient Lipshitz constant or the strong convexity constant. As a result, the ideal adaptive method should require very little tuning by the user, while matching or surpassing the performance of a fine-tuned $\gamma_k$. While commonly used adaptive methods such as Adam and AdaGrad often **require less tuning** than vanilla SGD in practice, the associated convergence rates **do not quantitatively showcase this advantage** and often rely on strong assumptions — e.g. that the iterates live on a **bounded domain**, or that gradients are uniformly bounded in norm (Duchi et al., 2011; Ward et al., 2019; Vaswani et al., 2020). While the above assumptions are valid in the constrained setting, they are problematic for problems defined in the whole $\mathbb{R}^d$ — where SGD enjoys the strongest guarantees under limited assumptions.

A NEW PROMISE: THE STOCHASTIC POLYAK STEPSIZES. A promising new direction in the adaptive stepsizes literature is based on the idea of **Polyak stepsizes**, introduced by Polyak, 1987 in the context of deterministic convex optimization. Recently, Loizou et al., 2021 successfully adapted Polyak stepsizes to the stochastic setting. The algorithm by Loizou et al., 2021 is named the Stochastic Polyak Stepsize — **SPS**:

$$\gamma_k = \min\left\{\frac{f_{\mathcal{S}_k}(x_k) - f^*_{\mathcal{S}_k}}{c\|\nabla f_{\mathcal{S}_k}(x_k)\|^2}, \gamma_b\right\}, \qquad \text{(SPS}_{\max}\text{)}$$

where $\gamma_b, c > 0$ are problem-independent constants, $f_{\mathcal{S}_k} := \frac{1}{|\mathcal{S}_k|}\sum_{i\in\mathcal{S}_k} f_i$, and $f^*_{\mathcal{S}_k} = \min_{x\in\mathbb{R}^d} f_{\mathcal{S}_k}(x)$. Loizou et al., 2021 provided convergence rates matching fine-tuned SGD — yet crucially the algorithm does **not require knowledge** of the unknown quantities such as the gradient **Lipschitz constant**. The results especially shine in the overparameterized strongly convex setting, where linear convergence to $\mathcal{X}^*$ is shown. This result is especially important since, under the same assumption, no such rate exists for AdaGrad (see e.g. (Vaswani et al., 2020) for the latest results) or other adaptive stepsizes. Moreover, SPS was shown to work **well on deep learning** problems (Loizou et al., 2021).

ISSUES WITH SPS. While the stochastic Polyak stepsize (SPS) (Loizou et al., 2021) is a clear step forward in the literature, it has two main drawbacks when it is used in **non-over-parameterized** regimes:

1. It requires *a priori* knowledge of the optimal mini-batch losses $f^*_{\mathcal{S}_k} := \min_{x\in\mathbb{R}^d} f_{\mathcal{S}_k}(x)$, which are not easily available for big batch sizes or regularized objectives.

2. It guarantees convergence only to a neighborhood of the solution, and this cannot be improved by decreasing/increasing hyperparameters. In other words, **SPS cannot** be used to **reach an arbitrarily small neighborhood of the solution**.

In this chapter, inspired by the shortcomings of SPS, we study in Section 4.1 its dynamics and found concerning issues with its convergence as the maximum stepsize bound $\gamma_b$ decreases. While in Section 4.2 we precisely **solve the two issues** above, we propose in Section 4.3 an **improved algorithm, NGN**, which stems from different principles (Gauss-Newton approximation) but is related to the Polyak stepsize yet possesses stronger theoretical guarantees and performance.

Since this chapter is heavy on convex smooth optimization, we recall some basic definitions used throughout the chapter. A similar discussion can also be found in Appendix A.

**Definition 4.0.1** (Strong Convexity / Convexity). *The function $f : \mathbb{R}^d \to \mathbb{R}$, is called $\mu$-strongly convex, if there exists $\mu > 0$ such that $\forall x, y \in \mathbb{R}^d$:*

$$f(x) \geq f(y) + \langle \nabla f(y), x - y\rangle + \frac{\mu}{2}\|x - y\|^2 \tag{89}$$

*for all $x \in \mathbb{R}^d$. If inequality (89) holds with $\mu = 0$ the function $f$ is convex.*

**Lemma 4.0.1** (Gradient lower bound). *Let $f : \mathbb{R}^d \to \mathbb{R}$ be $\mu$-strongly convex with minimizer at $f^*$. Then, for all $x \in \mathbb{R}^d$ we have*

$$2\mu(f(x) - f(x^*)) \leq \|\nabla f(x)\|^2. \tag{90}$$

*Proof.* From the definition of strong convexity,

$$f^* = \inf_{x \in \mathbb{R}^d} f(x) \geq f(y) + \inf_{x \in \mathbb{R}^d} \left[ \langle \nabla f(y), x - y \rangle + \frac{\mu}{2} \|x - y\|^2 \right]. \tag{91}$$

The inf on the right-hand-side is computable, and it is achieved at $x = y - \frac{1}{\mu} \nabla f(y)$. By direct substitution, we get

$$f^* \leq f(y) - \frac{1}{\mu} \|\nabla f(y)\|^2 + \frac{\mu}{2} \frac{1}{L^2} \|\nabla f(y)\|^2 = \frac{1}{2\mu} \|\nabla f(y)\|^2. \tag{92}$$

This proves the result. □

**Definition 4.0.2** ($L$-smooth). *The function $f : \mathbb{R}^d \to \mathbb{R}$, is $L$-smooth (with $L \geq 0$), if $\forall x, y \in \mathbb{R}^d$:*

$$\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\|, \tag{93}$$

*or equivalently (since $\mathbb{R}^d$ is a convex set):*

$$f(x) \leq f(y) + \langle \nabla f(y), x - y \rangle + \frac{L}{2} \|x - y\|^2. \tag{94}$$

**Lemma 4.0.2** (Gradient upper bound). *Let $f : \mathbb{R}^d \to \mathbb{R}$ be $L$-smooth with minimizer at $f^*$. Then, for all $x \in \mathbb{R}^d$ we have*

$$2L(f(x) - f(x^*)) \geq \|\nabla f(x)\|^2. \tag{95}$$

*Proof.* The proof is very similar to that Lemma 4.0.1. Thanks to Eq. (94)

$$f^* = \inf_{x \in \mathbb{R}^d} f(x) \leq f(y) + \inf_{x \in \mathbb{R}^d} \left[ \langle \nabla f(y), x - y \rangle + \frac{L}{2} \|x - y\|^2 \right]. \tag{96}$$

The inf on the right-hand-side is computable, and it is achieved at $x = y - \frac{1}{L} \nabla f(y)$. Plugging in this value, the result follows. □

## 4.1 THE STOCHASTIC POLYAK STEPSIZE AND ITS ISSUES

In this section, we provide a concise overview of the results in Loizou et al., 2021, and highlight the main assumptions and open questions.
To start, we remind the reader a definition

**Definition 4.1.1** (Interpolation)**.** *The problem in Eq. (88) is said to be **interpolated** if there exists a problem solution $x^* \in \mathcal{X}^*$ such that $\inf_{x\in\mathbb{R}^d} f_i(x) = f_i(x^*)$ for all $i \in [n]$.*

The **degree of interpolation** at batch size $B$ can be quantified by the following quantity, introduced by Loizou et al., 2021 and studied also in Vaswani et al., 2020; D'Orazio et al., 2021: fix a batch size $B$, and let $\mathcal{S} \subseteq [n]$ with $|\mathcal{S}| = B$.

$$\sigma_B^2 := \mathbb{E}_{\mathcal{S}}[f_{\mathcal{S}}(x^*) - f_{\mathcal{S}}^*] = f(x^*) - \mathbb{E}_{\mathcal{S}}[f_{\mathcal{S}}^*]. \tag{97}$$

It is easy to realize that as soon as the problem in Eq. (88) is interpolated, then $\sigma_B^2 = 0$ for each $B \leq n$. In addition, note that $\sigma_B^2$ is non-increasing as a function of $B$.
The stepsize proposed by Loizou et al., 2021 is

$$\gamma_k = \min\left\{\frac{f_{\mathcal{S}_k}(x_k) - f_{\mathcal{S}_k}^*}{c\|\nabla f_{\mathcal{S}_k}(x_k)\|^2}, \gamma_b\right\}, \tag{SPS$_{\max}$}$$

where $\gamma_b, c > 0$ are problem-independent constants, $f_{\mathcal{S}_k} := \frac{1}{|\mathcal{S}_k|}\sum_{i\in\mathcal{S}_k} f_i$, $f_{\mathcal{S}_k}^* = \min_{x\in\mathbb{R}^d} f_{\mathcal{S}_k}(x)$.
The following lemma is the fundamental starting point for the analysis in Loizou et al., 2021. It can be deduced by combining Lemma 4.0.2 with Lemma 4.0.1.

**Lemma 4.1.1.** *Let $f(x) = \frac{1}{n}\sum_{i=1}^n f_i(x)$ where the functions $f_i$ are $\mu_i$-strongly convex and $L_i$-smooth, then*

$$\frac{1}{2L} \leq \frac{1}{2L_i} \leq \frac{f_i(x^k) - f_i^*}{\|\nabla f_i(x^k)\|^2} \leq \frac{1}{2\mu_i} \leq \frac{1}{2\mu}, \tag{98}$$

*where $f_i^* := \inf_x f_i(x)$, $L = \max\{L_i\}_{i=1}^n$ and $\mu = \min\{\mu_i\}_{i=1}^n$. When each $f_i$ is smooth but potentially nonconvex, the lower bound still holds true.*

**Problem 1: in the non-interpolated setting, $f^*_{\mathcal{S}_k}$ is not computable.** Crucially the algorithm requires knowledge of $f^*_{\mathcal{S}_k}$ for every realization of the mini-batch $\mathcal{S}_k$. In the non-regularized overparametrized setting (e.g. neural networks), $f_{\mathcal{S}_k}$ is often zero for every subset $\mathcal{S}$ (Zhang et al., 2021a). However, this is not the only setting where $f^*_{\mathcal{S}}$ is computable: e.g., in the regularized logistic loss with batch size 1, it is possible to recover a cheap closed form expression for each $f_i^*$ (Loizou et al., 2021). Unfortunately, if the batch-size is bigger than 1 or the loss becomes more demanding (e.g. cross-entropy), then *no such closed-form computation is possible*.
We now comment on the main result of Loizou et al., 2021.

**Theorem 4.1.1** (Main result of Loizou et al., 2021)**.** *Let each $f_i$ be $L_i$-smooth convex functions. Then SGD with $\text{SPS}_{\max}$, mini-batch size $B$, and $c = 1$, converges as:*

$$\mathbb{E}\left[f(\bar{x}^K) - f(x^*)\right] \leq \frac{\|x^0 - x^*\|^2}{\alpha K} + \frac{2\gamma_b \sigma_B^2}{\alpha}, \quad \alpha = \min\left\{\frac{1}{2cL}, \gamma_b\right\},$$

*where $\bar{x}^K = \frac{1}{K}\sum_{k=0}^{K-1} x_k$ and $L = \max\{L_i\}_{i=1}^n$ is the maximum smoothness constant. If in addition $f$ is $\mu$-strongly convex, then, for any $c \geq 1/2$, SGD with $\text{SPS}_{\max}$ converges as:*

$$\mathbb{E}\|x_k - x^*\|^2 \leq (1 - \mu\alpha)^k \|x^0 - x^*\|^2 + \frac{2\gamma_b \sigma_B^2}{\mu\alpha}.$$

In the overparametrized setting, the result guarantees convergence to the minimizer, **without knowledge of the gradient Lipschitz constant** (as vanilla SGD would instead require) and **without assuming bounded iterates** — in contrast to Vaswani et al., 2020.
As soon as (1) a *regularizer* is applied to the loss (e.g. $L2$ penalty), or (2) the number of datapoints gets comparable to the dimension, then the problem is not interpolated and $\text{SPS}_{\max}$ only converges to a neighborhood and it gets impractical to compute $f^*_{\mathcal{S}}$. There is also a more sneaky issue hidden in the rate, recently pointed out by Wang et al., 2023:

**Problem 2: The interpolation error is non-vanishing.** In Theorem 4.1.1, suboptimality reached both in the convex and strongly-convex setting is proportional to $\gamma_b/\alpha$, where $\alpha = \min\left\{\frac{1}{2cL}, \gamma_b\right\}$. It follows that $\gamma_b/\alpha = \gamma_b \max\{2cL, 1/\gamma_b\} \geq 1$. Therefore, for instance in the convex setting, the

error in the non-interpolated setting cannot be made, at least according to the provided rate, smaller than $2\sigma_B^2$. This for instance means that simply multiplying $\gamma_k$ by a decay factor $1/\sqrt{k}$ cannot directly yield convergence to $f^*$ for $\sigma_B^2 > 0$.
Our first contribution in Loizou et al., 2021 is to show that this problem is not an artifact of the analysis — **SPS has a bias**!

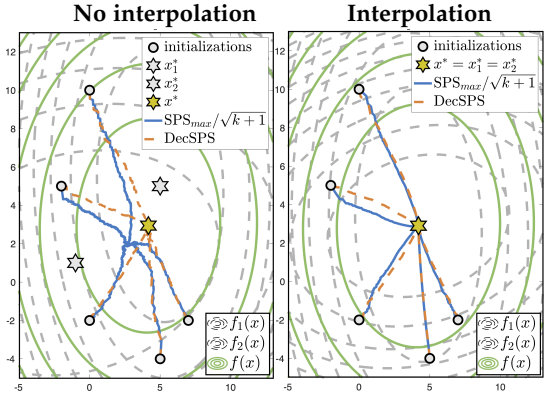


**Figure 32:** *Dynamics of $\text{SPS}_{\max}$ with decreasing multiplicative constant (SGD style) compared with DecSPS. We compared both in the **interpolated setting (right)** and in the **non-interpolated setting (left)**. In the non-interpolated setting, a simple multiplicative factor introduces a bias in the final solution, as discussed in this section. We consider two dimensional $f_i = \frac{1}{2}(x - x_i^*)^\top H_i (x - x_i^*)$, for $i = 1, 2$ and plot the contour lines of the corresponding landscapes, as well as the average landscape $(f_1 + f_2)/2$ we seek to minimize. Solution is denoted with a gold star.*

*Counterexample for $\text{SPS}_{\max}$ convergence under vanishing decay factor.* Consider the following finite-sum setting: $f(x) = \frac{1}{2}f_1(x) + \frac{1}{2}f_2(x)$ with $f_1(x) = \frac{a_1}{2}(x-1)^2, f_2(x) = \frac{a_2}{2}(x+1)^2$. To make the problem interesting, we choose $a_1 = 2$ and $a_2 = 1$: this introduces asymmetry in the average landscape with respect to the origin. During optimization, we sample $f_1$ and $f_2$ independently and seek convergence to the unique minimizer $x^* = \frac{a_1 - a_2}{a_1 + a_2} = 1/3$. The first thing we notice is that $x^*$ is not a stationary point for the dynamics under SPS. Indeed note that since $f_i^* = 0$ for $i = 1, 2$ we have (assuming $\gamma_b$ large enough): $\gamma_k \nabla f_{i_k}(x) = \frac{x-1}{2c_k}$, if $i_k = 1$, and

$\gamma_k \nabla f_{i_k}(x) = \frac{x+1}{2c_k}$ if $i_k = 2$. Crucially, note that this update is *curvature-independent*. The expected update is $\mathbb{E}_{i_k}[\gamma_k \nabla f_{i_k}(x)] = \frac{x-1}{4c_k} + \frac{x+1}{4c_k} = \frac{1}{2c_k} x$. Hence, the iterates can only converge to $x = 0$ — because this is the only fixed point for the update rule. The proof naturally extends to the multi-dimensional setting, an illustration can be found in Fig. 32, where $\text{SPS}_{\max}$ is also compared to DecSPS, the variation of SPS we propose in Orvieto et al., 2022c to solve this issue.

## 4.2 DECSPS: CONVERGENCE TO THE EXACT SOLUTION

*By concentrating on precision, one arrives at technique, but by concentrating on technique one does not arrive at precision.*

– Bruno Walter.

**Question:** SPS (Loizou et al., 2021) is a promising algorithm — has strong theory and compelling practical performance. However, two issues that limit its application in practice. Are the issues related to artifacts in the analysis? Is it possible to improve the algorithm and its convergence guarantees?

**Answer (Orvieto et al., 2022c):** The problems of SPS are rooted in the algorithm design (as we showed in Section 4.1) — rates are tight. We solve both issues by modifying the algorithm — introducing DecSPS. *Stay tuned for Section 4.3, where we design something even better*!

The algorithm we study in this section is **Decreasing SPS (DecSPS)**

$$\gamma_k := \frac{1}{c_k} \min \left\{ \frac{f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}}{\|\nabla f_{\mathcal{S}_k}(x^k)\|^2}, \; c_{k-1}\gamma_{k-1} \right\}, \tag{DecSPS}$$

for $k \in \mathbb{N}$, where $c_k \neq 0$ for every $k \in \mathbb{N}$. We set $c_{-1} = c_0$ and $\gamma_{-1} = \gamma_b > 0$ (stepsize bound, similar to Loizou et al., 2021), to get $\gamma_0 := \frac{1}{c_0} \cdot \min \left\{ \frac{f_{\mathcal{S}_0}(x^0) - \ell^*_{\mathcal{S}_0}}{\|\nabla f_{\mathcal{S}_0}(x^k)\|^2}, \quad c_0\gamma_b \right\}$.

Note that (solving the first problem of SPS) in the update rule we use a **lower bound** $\ell^*_{\mathcal{S}} \leq f^*_{\mathcal{S}}$. This bound can be set to zero for most problems **without reducing the method adaptivity** as we discuss in the next subsection. Due to this modification, we need to slightly **modify our suboptimality measure**, used for deriving our guarantees:

$$\hat{\sigma}^2_B := \mathbb{E}_{\mathcal{S}_k}[f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}] = f(x^*) - \mathbb{E}_{\mathcal{S}_k}[\ell^*_{\mathcal{S}_k}]. \tag{99}$$

It is easy to see by induction that DecSPS enjoys the following bounds.

**Lemma 4.2.1** ($\gamma_k$ is decreasing). *Let each $f_i$ be $L_i$ smooth and let $(c_k)_{k=0}^{\infty}$ be any non-decreasing positive sequence of real numbers. For DecSPS, we have*

$$\min\left\{\frac{1}{2c_kL}, \frac{c_0\gamma_b}{c_k}\right\} \leq \gamma_k \leq \frac{c_0\gamma_b}{c_k}, \quad \gamma_{k-1} \leq \gamma_k. \tag{100}$$

The proof can be found in Appendix D.

### 4.2.1 *Convergence Under Bounded Iterates*

The following result shows convergence for SGD with DecSPS.

**Theorem 4.2.1.** *Consider SGD with DecSPS and let $(c_k)_{k=0}^{\infty}$ be any non-decreasing sequence such that $c_k \geq 1, \forall k \in \mathbb{N}$. Assume that each $f_i$ is convex and $L_i$ smooth. We have:*

$$\mathbb{E}[f(\bar{x}^K) - f(x^*)] \leq \frac{2c_{K-1}\tilde{L}D^2}{K} + \frac{1}{K}\sum_{k=0}^{K-1}\frac{\hat{\sigma}_B^2}{c_k}, \tag{101}$$

*where $D^2 := \max_{k\in[K-1]} \|x^k - x^*\|^2$, $\tilde{L} := \max\left\{\max_i\{L_i\}, \frac{1}{2c_0\gamma_b}\right\}$ and $\bar{x}^K = \frac{1}{K}\sum_{k=0}^{K-1} x^k$.*

If $\hat{\sigma}_B^2 = 0$, then $c_k = 1$ for all $k \in \mathbb{N}$ leads to a rate $\mathcal{O}(\frac{1}{K})$, well known from Loizou et al., 2021. If $\hat{\sigma}_B^2 > 0$, as for the standard SGD analysis under decreasing stepsizes, the choice $c_k = \mathcal{O}(\sqrt{k})$ leads to an **optimal asymptotic trade-off** between the deterministic and the stochastic terms, hence to the asymptotic rate $\mathcal{O}(1/\sqrt{k})$ since $\sum_{k=0}^{K-1}\frac{1}{\sqrt{k+1}} \leq 2\sqrt{K}$. Moreover, picking $c_0 = 1$ minimizes the convergence speed for the deterministic factor. Under the assumption that $\hat{\sigma}_B^2 \ll \tilde{L}D^2$ (e.g. reasonable distance initialization-solution and $L > 1/\gamma_b$), this factor is dominant compared to the factor involving $\hat{\sigma}_B^2$. For this setting, the rate simplifies as follows.

**Corollary 4.2.1.** *Under the setting of Thm. 4.2.1, for $c_k = \sqrt{k+1}$ ($c_{-1} = c_0$) we have*

$$\mathbb{E}[f(\bar{x}^K) - f(x^*)] \leq \frac{2\tilde{L}D^2 + 2\hat{\sigma}_B^2}{\sqrt{K}}. \tag{102}$$

**Remark 4.2.1** (Beyond bounded iterates)**.** *The result above crucially relies on the **bounded iterates** assumption: $D^2 < \infty$. To the best of our knowledge, if no further regularity is assumed, modern convergence results for adaptive*

*methods (e.g. variants of AdaGrad) in convex stochastic programming require*[1] *this assumption, or else require gradients to be globally bounded. To mention a few: Duchi et al., 2011; Reddi et al., 2018; Ward et al., 2019; Défossez et al., 2022; Vaswani et al., 2020. A simple algorithmic fix to this problem is adding a cheap projection step onto a large bounded domain (Levy et al., 2018). We can of course include this projection step in DecSPS, and the theorem above will hold with no further modification. Yet we found this to be not necessary: the strong guarantees of SPS in the strongly convex setting (Loizou et al., 2021) let us go one step beyond: in Sec. 4.2.2 we show that, if each $f_i$ is strongly convex (e.g. regularizer is added), then one can bound the iterates globally with probability one, without knowledge of the gradient Lipschitz constant. To the best of our knowledge, no such result exist for AdaGrad — except Traoré and Pauwels, 2021, for the deterministic case.*

**Remark 4.2.2** (Dependency on the problem dimension)**.** *In standard results for AdaGrad, a dependency on the problem dimension often appears (e.g. Thm. 1 in Vaswani et al., 2020). This dependency follows from a bound on the AdaGrad preconditioner that can be found e.g. in Thm. 4 in Levy et al., 2018. In the SPS case no such dependency appears — specifically because the stepsize is lower bounded by* $1/(2c_kL)$.

### 4.2.2 *Removing the Bounded Iterates Assumption*

We prove that under **DecSPS the iterates live in a set of diameter $D_{\max}$ almost surely**. This can be done assuming **strong convexity** of each $f_i$. The result uses this alternative definition of *neighborhood*:

$$\hat{\sigma}^2_{B,\max} := \max_{\mathcal{S}\subseteq[n],|\mathcal{S}|=B} [f_{\mathcal{S}}(x^*) - \ell^*_{\mathcal{S}}]. \tag{103}$$

Note that trivially $\hat{\sigma}^2_{B,\max} < \infty$ under the assumption that all $f_i$ are lower bounded and $n < \infty$.

**Proposition 4.2.1.** *Let each $f_i$ be $\mu_i$-strongly convex and $L_i$-smooth. The iterates of SGD with DecSPS with $c_k = \sqrt{k+1}$ (and $c_{-1} = c_0$) are such that $\|x^k - x^*\|^2 \leq D^2_{\max}$ almost surely $\forall k \in \mathbb{N}$, where*

$$D^2_{\max} := \max\left\{\|x^0 - x^*\|^2, \frac{2c_0\gamma_b\hat{\sigma}^2_{B,\max}}{\min\left\{\frac{\mu}{2L}, \mu\gamma_b\right\}}\right\},$$

1 Perhaps the only exception is the result of Xie et al., 2020a, where the authors work on a different setting: i.e. they introduce the RUIG inequality.

*with* $\mu = \min_{i\in[n]} \mu_i$ *and* $L = \max_{i\in[n]} L_i$.

The proof relies on the variations of constants formula and an induction argument — it is provided in the appendix. We are now ready to state the main theorem for the unconstrained setting, which follows from Prop. 4.2.1 and Thm. 4.2.1.

**Theorem 4.2.2.** *Consider SGD with the DecSPS stepsize for* $k \geq 1$ *and* $\gamma_0$ *defined at the beginning of this section. Let each* $f_i$ *be* $\mu_i$*-strongly convex and* $L_i$*-smooth:*

$$\mathbb{E}[f(\bar{x}^K) - f(x^*)] \leq \frac{2\tilde{L}D_{\max}^2 + 2\hat{\sigma}_B^2}{\sqrt{K}}. \tag{104}$$

To the best of our knowledge, at the time of its publication, this was the **first rate for adaptive methods that showcases convergence** to the precise solution **without knowledge of problem-dependent parameters and without bounded gradients/domain assumptions**.

**Remark 4.2.3** (Strong Convexity)**.** *The careful reader might notice that, while we assumed strong convexity, our rate is slower than the optimal* $\mathcal{O}(1/K)$*. This is due to the adaptive nature of DecSPS. It is indeed notoriously hard to achieve a convergence rate of* $\mathcal{O}(1/K)$ *for adaptive methods in the strongly convex regime. While further investigations will shed light on this interesting problem, we note that the result we provide is somewhat unique in the literature: we are not aware of any adaptive method that enjoys a similar convergence rate without either (a) assuming bounded iterates/gradients or (b) assuming knowledge of the gradient Lipschitz constant or the strong convexity constant.*

**Remark 4.2.4** (Comparison with Vanilla SGD)**.** *On a convex problem, the non-asymptotic performance of SGD with a decreasing stepsize* $\gamma_k = \eta/\sqrt{k}$ *strongly depends on the choice of* $\eta$*. The optimizer might diverge if* $\eta$ *is too big for the problem at hand. Indeed, most bounds for SGD, under no access to the gradient Lipschitz constant, display a dependency on the size of the domain and rely on projections after each step. If one applies the method in the unconstrained setting, such convergence rates technically do not hold, and tuning is sometimes necessary to retrieve stability and good performance. Instead, for DecSPS, simply by adding a small regularizer, the method is guaranteed to converge at the non-asymptotic rate we derived even in the unconstrained setting.*

### 4.2.3 *Experiments in Convex Optimization*

In this subsection, we present a few experiments comparing DecSPS with SGD and adaptive methods. As we motivate in the main paper (Orvieto et al., 2022c) leveraging a stability analysis, choosing DecSPS hyperparameters $c_0 = 1, \gamma_b = 10$ yields consistent result – we fix these in our experiments. For additional experiments and comparisons, please check Orvieto et al., 2022c.

First, we compare the performance of DecSPS against the classical decreasing SGD stepsize $\eta/\sqrt{k+1}$, which guarantees convergence to the exact solution at the same asymptotic rate as DecSPS. We show that, while the asymptotics are the same, DecSPS with hyperparameters $c_0 = 1, \gamma_b = 10$ performs competitively to a fine-tuned $\eta$ — where crucially the optimal value of $\eta$ depends on the problem. This behavior is shown on all the considered datasets (three in the main paper, see Orvieto et al., 2022c), and is reported in Figure 33 for the LIBSVM A1A dataset (Chang, 2011). We inspect the value of $\gamma_k$ returned by DecSPS. Compared to the vanilla SGD stepsize $\eta/\sqrt{k+1}$, a crucial difference appears: $\gamma_k$ decreases faster than $\mathcal{O}(1/\sqrt{k})$. This showcases that, while the factor $\sqrt{k+1}$ can be found in the formula of DecSPS[2], the algorithm structure provides additional **adaptation to curvature**. Next, in Figure 34 we compare DecSPS with another adaptive coordinate-independent stepsize with strong theoretical guarantees: the norm version of AdaGrad (a.k.a. AdaGrad-Norm, AdaNorm), which guarantees the exact solution at the same asymptotic rate as DecSPS (Ward et al., 2019). AdaGrad-norm at each iteration updates the scalar $b_{k+1}^2 = b_k^2 + \|\nabla f_{\mathcal{S}_k}(x_k)\|^2$ and then selects the next step as $x_{k+1} = x_k - \frac{\eta}{b_{k+1}}\nabla f_i(x_k)$. Hence, it has tuning parameters $b_0$ and $\eta$. In Fig. 34 we show that, on the Breast Cancer dataset, after fixing $b_0 = 0.1$ as recommended in Ward et al., 2019 (see their Figure 3), tuning $\eta$ cannot quite match the performance of DecSPS.

Finally, in Figures 35 and 36 we compare DecSPS with tuned versions of Adam (Kingma and Ba, 2014) and AMSgrad (the convergent version of Adam) (Reddi et al., 2018). We show an advantage over both.

2 We pick $c_k = c_0\sqrt{k+1}$, as suggested by Cor. 4.2.1

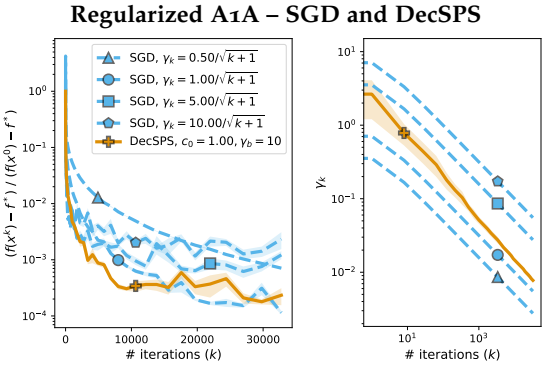


**Figure 33:** *Performance of SGD compared with DecSPS, on the A1A Dataset (regularization $\lambda = 0.01$).*

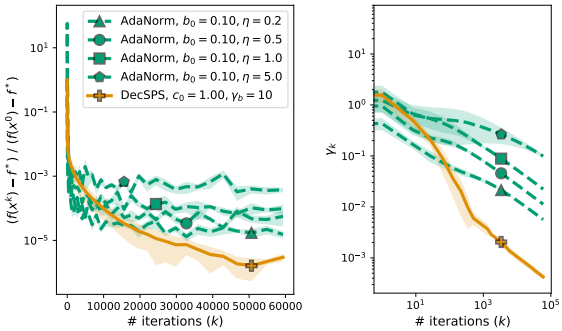


**Figure 34:** *Performance of AdaNorm compared with DecSPS on the Breast Cancer Dataset (regularization $\lambda = 1e-1$).*

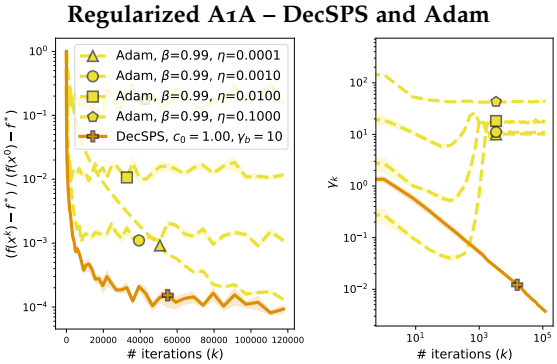


**Figure 35:** *Performance of Adam (with fixed stepsize and no momentum) compared to DecSPS on the A1A dataset (regularization $\lambda = 0.01$). Plotted is also the average stepsize (each parameter evolves with a different stepsize).*

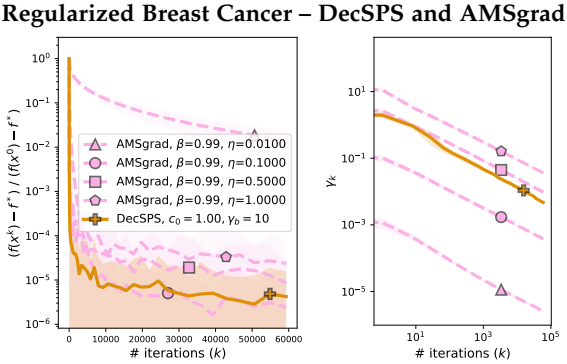


**Figure 36:** *Performance of AMSgrad (with sqrt decreasing stepsize and no momentum) compared to DecSPS on the Breast Cancer dataset (regularization $\lambda = 0.1$), respectively. Plotted is also the average stepsize (each parameter evolves with a different stepsize).*

## 4.3 NON-NEGATIVE GAUSS-NEWTON: A NEW APPROACH

*No great discovery was ever made without a bold guess.*

– Isaac Newton.

If you are an optimization researcher, you will love the analysis of the new algorithm presented here. If you are a practitioner, you will be impressed by the empirical results.

**Question:** In Section 4.2 we modified the SPS update rule to solve its bias problem and delivered a solid adaptive algorithm with desirable theoretical guarantees. The resulting method however is arguably complicated, has two hyperparameters, and a bounded domain issue (which can be solved assuming strong convexity). Can we develop something simpler that is interpretable and has stronger guarantees?

**Answer (Orvieto and Xiao, 2023):** Starting from the assumption that the loss over datapoints is non-negative, we can construct an adaptive method leveraging a novel Gauss-Newton-like approximation. The resulting algorithm is cheap to implement (same complexity as SGD) and is related to SPS — and has much stronger guarantees and performance, also in the Deep Learning setting. *We are extremely excited about this method, and our current efforts are focused on testing performance on state-of-the-art architectures.*

Consider again, as usual in this chapter, the problem of minimizing the average of a large number of loss functions: $\min_{x\in\mathbb{R}^d} f(x) := \frac{1}{N}\sum_{i=1}^N f_i(x)$, where each $f_i : \mathbb{R}^d \to \mathbb{R}$ is smooth but potentially non-convex. In machine learning applications each $f_i$ is the loss function associated with a training example, and is usually **non-negative**.

In this section, we derive Non-negative Gauss-Newton (NGN), a novel adaptive method that shows superior performance compared to SGD both in theory and in practice. Specifically, by leveraging a **Gauss-Newton inspired approximation of non-negative losses**, we propose the follow-

ing adaptive stepsize on top of the stochastic gradient update $x^{k+1} = x^k - \gamma_k \nabla f_{i_k}(x^k)$:

$$\gamma_k = \frac{\sigma}{1 + \frac{\sigma}{2f_{i_k}(x^k)}\|\nabla f_{i_k}(x^k)\|^2}, \qquad \text{(NGN-stochastic)}$$

where $i_k \in \{0, 1, \ldots, N\}$ is the datapoint sampled at iteration $k$ and $\sigma$ is a regularization hyperparameter. This adaptive rule acts as an **automatic warmup-decay scheduler**, which often has to be hand-picked in deep learning practice (Loshchilov and Hutter, 2016).

### 4.3.1 *Algorithm Derivation*

Let each $f_i$ be differentiable and non-negative. We define

$$r_i(x) := \sqrt{f_i(x)}, \tag{105}$$

and rewrite the loss as a **sum of squares**:

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} r_i^2(x). \tag{106}$$

Consequently, we have

$$\nabla f_i(x) = 2r_i(x)\nabla r_i(x), \quad \text{and} \quad \nabla r_i(x) = \frac{1}{2\sqrt{f_i(x)}}\nabla f_i(x). \tag{107}$$

The Gauss-Newton update constructs the descent direction $p$ as follows. First, we approximate the loss function with the first-order Taylor expansions of the $r_i$'s around $x$:

$$f(x+p) = \frac{1}{n}\sum_{i=1}^{n} r_i^2(x+p) \approx \frac{1}{n}\sum_{i=1}^{n}\left(r_i(x) + \nabla r_i(x)^\top p\right)^2. \tag{108}$$

Then $p$ is set as the minimizer of the approximation together with a regularization term

$$\tilde{f}_\sigma(x+p) := \frac{1}{n}\sum_{i=1}^{n}\left(r_i(x) + \nabla r_i(x)^\top p\right)^2 + \frac{1}{2\sigma}|p\|^2. \tag{109}$$

Since this is a quadratic function in $p$, the minimizer can be found by setting $\nabla_p \tilde{f}_\sigma(x+p) = 0$, i.e.,

$$\begin{aligned} 0 = \nabla_p \tilde{f}_\sigma(x+p) &= \frac{1}{n}\sum_{i=1}^{n} 2\left(r_i(x) + \nabla r_i(x)^\top p\right)\nabla r_i(x) + \frac{1}{\sigma}p \\ &= \frac{2}{n}\sum_{i=1}^{n} r_i(x)\nabla r_i(x) + \left[\frac{2}{n}\sum_{i=1}^{n}\nabla r_i(x)\nabla r_i(x)^\top + \frac{1}{\sigma}I\right]p. \end{aligned} \tag{110}$$

Therefore $p \in \mathbb{R}^d$ needs to satisfy the normal equation

$$\left[\frac{1}{n}\sum_{i=1}^{n}\nabla r_i(x)\nabla r_i(x)^\top + \frac{1}{2\sigma}I\right]p = -\frac{1}{n}\sum_{i=1}^{n} r_i(x)\nabla r_i(x), \tag{111}$$

which leads to

$$p = -\left[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{2f_i(x)}\nabla f_i(x)\nabla f_i(x)^\top + \frac{1}{\sigma}I\right]^{-1}\nabla f(x). \tag{112}$$

To summarize, the regularized Gauss-Newton, or Levenberg-Marquardt, method, is given by

$$\begin{aligned} x^{k+1} &= x^k - G_\sigma(x^k)^{-1}\nabla f(x^k), \\ G_\sigma(x) &= \frac{1}{\sigma}I + \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2f_i(x)}\nabla f_i(x)\nabla f_i(x)^\top. \end{aligned} \tag{113}$$

GENERAL NON-NEGATIVE FUNCTIONS. Notice that we can apply the same trick with any non-negative function $f$, regardless if it has a finite-sum structure. In the general case, we have

$$x^{k+1} - x^k = -\left(\frac{1}{\sigma}I + \frac{\nabla f(x^k)\nabla f(x^k)^\top}{2f(x^k)}\right)^{-1}\nabla f(x^k) \tag{114}$$

$$= -\frac{\sigma}{1 + \frac{\sigma}{2f(x^k)}\|\nabla f(x^k)\|^2}\nabla f(x^k). \tag{115}$$

We call this method **NGN: Non-negative Gauss-Newton**, for optimization without mini-batching. For short, we will refer to this as NGN-det.

STOCHASTIC OPTIMIZATION. We can apply the reasoning above to the function sampled at each iteration, i.e. the one-sample estimation of Eq. (113). The resulting algorithm is NGN-stochastic, presented in the introduction.

### 4.3.2 *Basic Properties of NGN*

We give below an overview of the properties of the NGN stepsizes, here discussed in the deterministic setting but with direct application in the stochastic case.

**Non-negativity implies a stepsize range.** For any $f : \mathbb{R}^d \to \mathbb{R}$ that is $L$-smooth with minimum value $f^*$, for all $x \in \mathbb{R}^d$ we have

$$2L(f(x) - f^*) \geq \|\nabla f(x)\|^2. \tag{116}$$

Note that, under our assumption $f^* > 0$, so — crucially — we also have

$$2Lf(x) \geq \|\nabla f(x)\|^2. \tag{117}$$

Or, more clearly, $0 \leq \frac{\|\nabla f(x)\|^2}{2f(x)} \leq L$. This directly implies a range for $\gamma_k$:

$$\gamma_k \in \left[\frac{\sigma}{1+\sigma L}, \sigma\right] = \left[\frac{1}{L+\sigma^{-1}}, \sigma\right]. \tag{118}$$

This property illustrates the behavior of NGN, as can also be observed in our experiments: $\sigma$ bounds the maximum achievable stepsize, but the algorithm can adaptively choose to decrease the stepsize until $1/(L+\sigma^{-1})$ if the landscape gets more challenging. On the theoretical side, the stepsize bounds above imply that NGN, in the worst case, cannot be asymptotically worse than gradient descent in deterministic optimization. As we will see, however in the next section, the algorithm actually has a very peculiar convergence guarantee in the convex setting: even for large values of $\sigma$, we are **guaranteed to never diverge**, see Figures 37, 38, and 40.

**Curvature estimation.** As it can already be concluded from the last section, the term $\frac{\|\nabla f(x)\|^2}{2f(x)}$, which modulates the stepsize, is an approximate (conservative) curvature estimator. Indeed, simply taking $f(x) =$

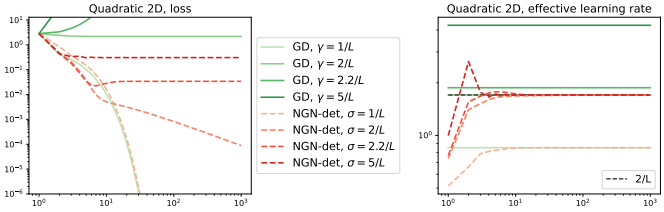


**Figure 37:** *Two-dimensional positive semidefinite quadratic with minimum at $f^* = 0$, performance and dynamics for different hyperparameters for the deterministic variant of NGN (regularization $\sigma$) and GD (stepsize $\gamma$). As the hyperparameter increases, GD diverges, while NGN does not due to its feedback mechanism: the effective stepsize converges to $2/L$. Combined with decreasing $\sigma$, NGN is found convergent (see Thm. 4.3.1) in the stochastic setting for any hyperparameter values and no bounded gradient assumptions.*

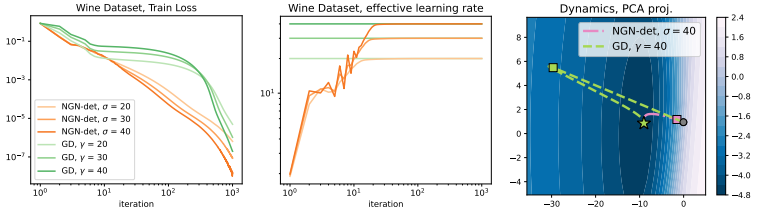


**Figure 38:** *Loss (normalized to reach zero), effective stepsize, and PCA projection of iterations (dynamics for biggest $\gamma$ and $\sigma$ are plotted) for the regularized Wine dataset (Chang, 2011) (more datasets shown in the appendix). In the dynamics plot, the circle denotes the starting point, the star the solution found at the last iteration, and the square indicates the solution after one iteration.*

$\lambda(x - x^*)^2/2 + c$, we have $\frac{\|\nabla f(x)\|^2}{2f(x)} \geq \lambda$. As a result, the method is allowed the maximum stepsize $\sigma$ on flat regions and decreases the stepsize on sharp regions, as can also be observed in Figure 39.

**Relation to the Polyak stepsize (Polyak, 1987) as SGD.** The reader can istantaneously spot the resemblance between Eq. (115) and the Polyak stepsize $\gamma_k = \frac{f(x^k) - f^*}{\|\nabla f(x^k)\|}$, where $f^* = \min_x f(x)$. Indeed, recall the following limit: any $a \neq 0$ we have $\lim_{\sigma \to \infty} \frac{\sigma}{1 + a\sigma} = \frac{1}{a}$. Therefore, as $\sigma \to \infty$, NGN reduces to $\gamma_k = \frac{2f(x)}{\|\nabla f(x)\|^2}$. The difference is that $f^*$ is (conveniently)

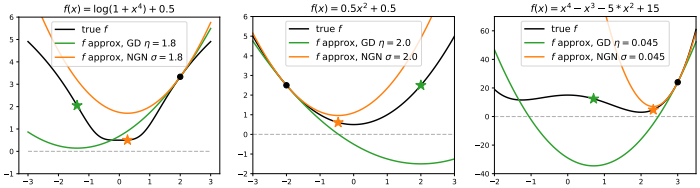


**Figure 39:** *NGN update and corresponding function approximation on a few toy examples. The black dot denotes the algorithm's initial position and the star the position after one step. Compared to gradient descent with stepsize $\eta = \sigma$, NGN is more conservative then GD if the landscape is sharp. Note that the function approximation provided by NGN is always non-negative, as clear from the algorithm derivation and the motivation.*

not in the update rule. We show in Section 4.3.3 that this difference does not hurt convergence in the interpolation setting. Moreover, as $\sigma \to 0$ we get back to standard gradient descent with stepsize $\sigma$. As such, NGN effectively interpolates between SGD and an $f^*$-agnostic Polyak stepsize.

EMPIRICAL VERIFICATION OF BASIC PROPERTIES. In Figure 39 we show the NGN function approximation leading to a stepsize $\gamma_k$ aware of the loss sharpness: the **step is more conservative if the landscape is sharp**. In Figure 37, we test one prediction of our main theorem, Thm 4.3.1. This theorem states that, in the case where $f^* > 0$, if $\sigma$ (the maximum allowed stepsize, see stepsize bounds) is big (bigger than $\sigma > 1/(2L)$ in the general convex case), then NGN does not lead to divergence (as would SGD with stepsize $\sigma$) but instead we get convergence to a quantity proportional to $f^*$.

### 4.3.3 *Convergence Results (you won't believe these!)*

We provide here convergence guarantees for stochastic NGN. We start by defining $x^* := \arg\min f(x)$ and $f_i^* := \min_x f_i(x)$, and then define the following two error quantities

$$\Delta_{\text{int}} := \mathbf{E}[f_i(x^*) - f_i^*], \qquad \Delta_{\text{pos}} := \mathbf{E}[f_i^*]. \tag{119}$$

Here $\Delta_{\text{pos}}$ measures how close is, on average, the individual datapoint loss to the value zero, while $\Delta_{\text{int}}$ measures interpolation. While for over-parametrized models in deep learning one has $\Delta_{\text{pos}} = \Delta_{\text{int}} = 0$ (Vaswani

et al., 2020), both the cases $\Delta_{\text{pos}} = 0, \Delta_{\text{int}} > 0$ and $\Delta_{\text{pos}} > 0, \Delta_{\text{int}} = 0$ are feasible in theory. We provide proofs for all the results below in the appendix.

**Theorem 4.3.1** (NGN, convex). *Let $f = \frac{1}{N}\sum_{i=1}^{N} f_i$, where each $f_i : \mathbb{R}^d \to \mathbb{R}$ is non-negative, $L_i$-smooth and convex. Let $L = \max_{i\in[N]} L_i$. Consider stochastic gradient descent with batch size 1 and let $i_k$ be the data index sampled at iteration $k$. For any value of $\sigma > 0$,NGN-stochastic leads to*

$$\mathbf{E}\left[f(\bar{x}^k) - f(x^*)\right] \leq \frac{\mathbf{E}\|x^0 - x^*\|^2}{\eta_\sigma K} + 3\sigma L \cdot (1+\sigma L)\Delta_{int} + \sigma L \cdot \max\{0, 2\sigma L - 1\}\,\Delta_{pos},$$

*where $\bar{x}^K = \frac{1}{K}\sum_{k=0}^{K-1} x^k$, $\eta_\sigma := \frac{2\sigma}{(1+2\sigma L)^2}$. Decreasing $\sigma$, we get a rate $\mathcal{O}\left(\frac{\ln(K)}{\sqrt{K}}\right)$.*

COMMENT ON THE RESULT. If $\Delta_{\text{int}} = \Delta_{\text{pos}} = 0$, then the result guarantees convergence to the solution for any value of $\sigma$. The best achievable constant in the rate is achieved when knowing the Lipschitz constant, indeed $\min_\sigma \eta_\sigma$, achieved at $\sigma = 1/(2L)$. If $\Delta_{\text{pos}} > 0$, then for $\sigma > 1/(2L)$ we get error term — which does not come from gradient stochasticity but is instead an effect of **correcting divergence** of the gradient update for large stepsizes. We note that in this case our method behaves better than SGD, which is divergent for large stepsizes. Finally, the interpolation error term $2\sigma\Delta_{\text{int}}$ results from gradient stochasticity in the non-overparametrized setting.

COMPARISON WITH RELATED WORKS. The error in Thm. 4.3.1 is $\mathcal{O}(\sigma)$, meaning that as $\sigma \to 0$ the **error vanishes**. This is not the case for adaptations of the Polyak scheme to the stochastic setting (Loizou et al., 2021; Wang et al., 2023), where the error is $\mathcal{O}(1)$. To the best of our knowledge, this is the **first rate in the literature to a ball of arbitrary size around the solution without knowledge of the smoothness constant**.
As we will see in Section 4.3.4, stochastic NGN can also be applied successfully in the nonconvex setting. In this setting, knowledge of the Lipschitz constant is required.

**Theorem 4.3.2.** *Let $f = \frac{1}{N}\sum_{i=1}^{N} f_i$, where each $f_i : \mathbb{R}^d \to \mathbb{R}$ is non-negative, $L_i$-smooth and potentially non-convex. Let $L = \max_{i\in[N]} L_i$. Consider stochas-*

*tic gradient descent with batch size* 1 *and let* $i_k$ *be the data index sampled at iteration* $k$*. For* $\sigma \leq \frac{1}{2L}$*, NGN-stochastic leads to*

$$\mathbf{E}\left[\frac{1}{K}\sum_{k=0}^{K-1}\|\nabla f(x^k)\|^2\right] \leq \frac{12\cdot \mathbf{E}[f(x^0)-f^*]}{\sigma K} + 18\sigma L\Delta_{noise}^2,$$

*where* $\Delta_{noise}^2 = \sup_{x\in\mathbb{R}^d}\mathbf{E}[\|\nabla f(x)-\nabla f_i(x)\|^2]$*. Decreasing* $\sigma$*, we get a rate* $\mathcal{O}\left(\frac{\ln(K)}{\sqrt{K}}\right)$.

### 4.3.4 *Experiments in Deep Learning*

First, in Figure 40, we showcase the convergence of NGN on a convex problem with decreasing $\sigma$. The method converges **faster than tuned SGD**. In particular, NGN is **robust to initial overshooting**.

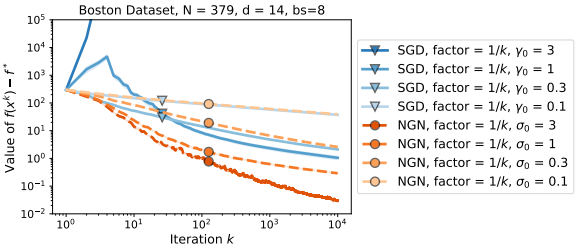


**Figure 40:** *Performance of NGN and SGD with a decreasing stepsize (average over 5 iterations) for linear regression on the Boston Dataset (Chang, 2011):* 14 *features and* 379 *datapoints. Since the problem is strongly convex, we choose a stepsize discount of* $1/k$ *(corresponding rate not presented in this thesis since currently under investigation). The batch size is* 8*.*

Next, we test convergence in the deep learning setting for a non-vanishing $\sigma$. As the reader can observe in Figures 41 (CIFAR10 on ResNet18) and Figure 42 (ImageNet on ResNet50), stochastic NGN, which has the same runtime as SGD, has faster convergence — even after tuning or adding momentum on top of SGD. In Figure 41 we also show that the effective stepsize $\gamma_k$ has an **interesting warm-up/decay behavior**, similar to those based on heuristics and hand-tuning in deep learning practice. Finally, in Figure 42 we also show that NGN is competitive with AdamW (Loshchilov

and Hutter, 2017), the de-facto optimizer used by practitioners e.g. when training large language models (Brown et al., 2020). We note that, in contrast to AdamW, our optimizer enjoys stronger theoretical guarantees such as convergence to a ball around the solution for every hyperparameter value (cf. Défossez et al., 2022).

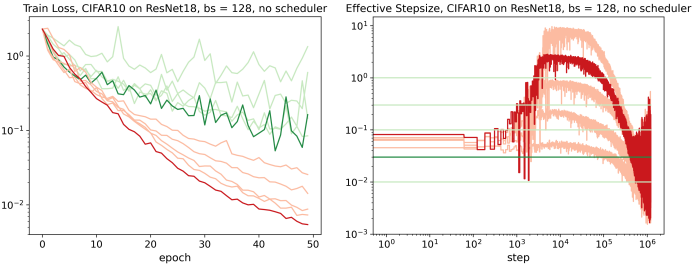


**Figure 41:** *Training CIFAR10 on a ResNet18 (11M parameters). Shown is the training loss for SGD (without momentum, in green) with stepsize* $\gamma = [0.01, 0.03, 0.1, 0.3, 1]$ *and NGN-stoch with* $\sigma = [0.1, 0.3, 1, 3, 10]$ *(in red). No scheduling is used for* $\gamma$ *and* $\sigma$*, and the batch size for each method is 128 (NGN-stoch is the mini-batch version of NGN-stochastic). Highlighed are the best hyperparameter options,* $\gamma = 0.03, \sigma = 3$.

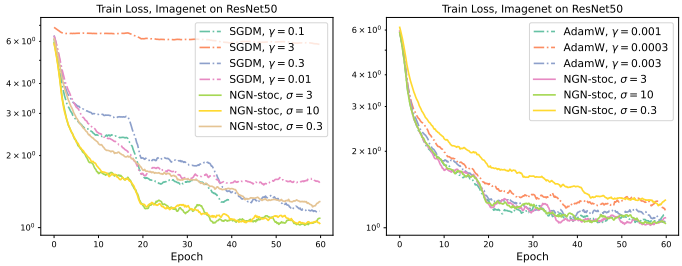


**Figure 42:** *Training Imagenet on a ResNet50 (23M parameters). Shown is the training loss for SGD (with momentum), AdamW (Loshchilov and Hutter, 2017) and NGN-stoch. Piecewise constant scheduling is used for* $\eta$ *and* $\sigma$*, the regularizer strength is* $1e-4$ *and the batch size is 128.*

# 5

# SOLVING A CHALLENGE: DESIGNING TRAINABLE DEEP RNNS

*And I enter the fields and roomy chambers of memory, where are the treasures of countless images, imported into it from all manner of things by the senses. There is treasured up whatsoever likewise we think, either by enlarging or diminishing, or by varying in any way whatever those things which the sense has arrived at; yea, and whatever else has been entrusted to it and stored up, which oblivion has not yet engulfed and buried.*

– St. Augustine

Throughout this thesis, we have demonstrated the effectiveness of adaptive momentum methods in training complex loss landscapes. In Chapter 3, we presented how the Adam optimizer handles both flat saddle points at initialization of deep unnormalized MLPs and challenging non-isotropic curvatures within the transformer architecture. In Chapter 4, we introduced NGN, a novel stepsize which enables automatic learning rate warm-up, facilitating training in scenarios where the gradient Lipschitz constant undergoes changes along the optimizer trajectory.

Based on our discussion, it appears that the key factor for successful training lies in the design of the optimizer. In this chapter, we want to showcase that **sometimes a good optimizer is not enough** — careful **parametrization and normalization of the model is crucial** — in challenging settings such as training deep Recurrent Neural Networks (RNNs) to state-of-the-art performance in long-range reasoning tasks.

The contribution presented in this chapter is, as opposed to the previous chapters, not a new optimizer or a novel analysis, but instead a new interpretable architecture for sequence modeling (Orvieto et al., 2023c, Oral at ICML2023). Our motivation for **revisiting the old optimization problem of training recurrent models** (Bengio et al., 1994) — which leads to **vanishing/expoding gradients** (Pascanu et al., 2013) — is a well-known **issue with transformers: inference and memory complexity scales quadratically with the sequence length**.

The structure of this chapter is as follows: in Section 5.1 we give additional motivation and summarize of the takeaways of our work; in Section 5.2 we provide a review of RNNs as well as of recently introduced sequence-to-sequence models like S4 (Gu et al., 2021). In Section 5.3, following Orvieto et al., 2023c we show how to design performant deep RNNs, and expand on interesting optimization issues in Section 5.4. Last, in Section 5.5 we present our latest work (Orvieto et al., 2023a) on universal approximation properties of the architecture studied in this Chapter.

**Question:** The attention mechanism, pervasive in modern deep learning due also to its high scalability, has provided incredible advances. For tasks that require reasoning along thousands or millions of tokens (e.g. music generation, protein structure prediction, biological signal processing), a direct implementation of attention becomes intractable (complexity quadratic in the sequence length). On the other hand, RNNs are not easy to scale and suffer from fundamental training issues such as vanishing/exploding gradients. However, inference and memory complexities for RNNs are lower than attention: proportional only to the sequence length.
Can a modern take on the fundamental issues of RNNs lead to a new architecture capable of dealing with challenging sequential data?

**Answer (Orvieto et al., 2023c):** It is possible to successfully train deep RNNs. Yet, in combination with an adaptive optimizer, fundamental architectural modifications (which do not lead to a change in model expressivity) are needed. We start from a vanilla stack of *tanh* RNNs and provide a sequence of theory-inspired steps towards the LRU, a modern recurrent mechanism that avoids vanishing/exploding gradients and reaches state-of-the-art on the long-range arena (Tay et al., 2020).

## 5.1 MOTIVATION AND MAIN STEPS

Recurrent neural networks (RNNs) have played a central role since the early days of deep learning, and are a natural choice when modelling sequential data (McCulloch and Pitts, 1943; Hopfield, 1982; Rumelhart et al., 1985; Elman, 1990). However, while these networks have strong theoretical properties, such as Turing completeness (Kilian and Siegelmann, 1996; Chung and Siegelmann, 2021), they can be hard to train in practice. In particular, RNNs suffer from the vanishing and exploding gradient problem (Hochreiter, 1991; Bengio et al., 1994; Pascanu et al., 2013), which makes it difficult to learn the long-range dependencies in the data. Several techniques were developed that attempt to mitigate this issue, including orthogonal/unitary RNNs (Arjovsky et al., 2016; Helfrich et al., 2018), and gating mechanisms such as long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997b) and gated recurrent units (GRUs) (Cho et al., 2014). Nonetheless, these models are still slow to optimize due to the inherently sequential nature of their computation (Kalchbrenner et al., 2016), and are therefore hard to scale.

In recent years, Transformers (Vaswani et al., 2017) have gained increasing prominence for sequence modelling tasks, achieving remarkable success in a wide range of applications (Brown et al., 2020; Dosovitskiy et al., 2020; Jumper et al., 2021). **Compared to RNNs, attention layers are easier to scale** and parallelize during training, and crucially they do not suffer from the vanishing gradient problem, since the interaction between any two tokens in the sequence is modeled by direct edges in the network. A key issue with **attention layers** however is that their **computation and memory costs scale quadratically as $O(L^2)$** for sequence length $L$. Transformers can therefore be expensive to deploy on long sequences. **RNNs**, which **scale linearly with the sequence length**, are faster than transformers at inference even for modest sequence lengths (Liu et al., 2019a).

Motivated by these problems, Gu et al., 2021 recently introduced the **S4 model**, a carefully designed **deep state-space model (SSM)** that achieves remarkable performance on tasks from the Long Range Arena (LRA) (Tay et al., 2020), a benchmark explicitly designed to require very long-ranged reasoning. The S4 layer and its variants (DSS, S4D, Liquid S4, S5, etc) (Gupta et al., 2022a; Gu et al., 2022a; Hasani et al., 2022; Smith et al., 2022) overcome the $O(L^2)$ bottleneck of attention layers by model-

ing interactions between tokens sequentially using a hidden state. These models are therefore very efficient at inference time as we can simply unroll the recurrent layer like an RNN. Futhermore, since SSMs are linear in the temporal dimension, they are easily parallelizable during training, in contrast to the slow sequential nature of training nonlinear RNNs. This makes them very computationally efficient on long sequences.

While the **S4 model is equivalent to an RNN during inference**, it has a number of **unique characteristics during training**. For example, S4 is parameterized as a discretization of a latent **continuous-time system** of differential equations, and it uses specific initializations of the state matrices motivated from the theory of **polynomial projections** (Gu et al., 2020). While these characteristics might seem to motivate the impressive performance of these models, later works (Gu et al., 2022a; Smith et al., 2022; Gupta et al., 2022a; Gupta et al., 2022b) have suggested that the specific initialization used by S4 is often not crucial for performance, and that the **discretization** rules which achieve best performance may not be the most accurate in theory (Smith et al., 2022). It is therefore unclear what these unique characteristics of deep SSMs are doing mechanistically, and how they **can be simplified**.

Motivated by the striking similarities between RNNs and deep SSMs, and in an attempt to better understand the underlying mechanism driving the performance of these models, we study the power and limitations of RNNs when used as core components of deep architectures for long-range reasoning. Our main goal is to answer the question:

*"Can we match the performance and efficiency of deep continuous-time SSMs using deep RNNs?"*

We give a *positive answer* to this question. We show that the performance boost provided by deep SSMs like S4 can also be achieved via a series of **small changes to a vanilla deep RNN**. With these changes, we can recover the performance and efficiency of deep SSMs on the **Long Range Arena (LRA)** benchmark (Tay et al., 2020). We call this new RNN model the Linear Recurrent Unit (or LRU for short).

MAIN STEPS. We outline here the main steps needed to design performant and efficient RNN models. While some of these observations appear in prior works (see the related work section in the appendix of Orvi-

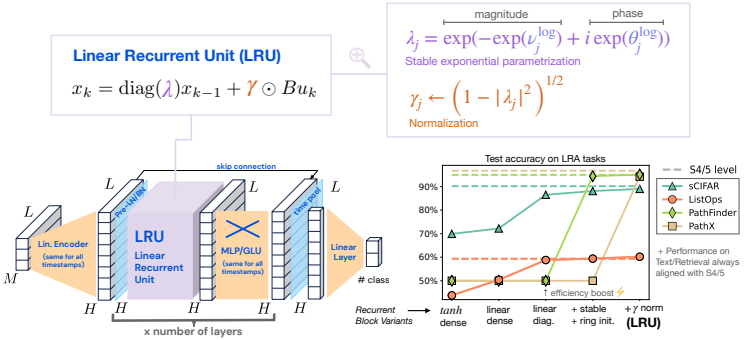


**Figure 43:** ***Left****: Our final architecture is a stack of Linear Recurrent Units (LRU), with nonlinear projections in between, and also uses skip connections and normalization layers like batch/layer normalization. We expand on model details in the Appendix of Orvieto et al., 2023c. We use the same architecture structure (Norm-Recurrence-GLU-Skip) for every variant of recurrent module in our study (tanh dense, linear dense, etc..).* ***Right****: Summary of performance for each of the main steps outlined in the introduction to design LRUs starting from tanh RNNs. We show the average performance on the Long Range Arena (LRA) across 3 seeds at each step, and also provide the average performance of deep SSMs. For all LRA tasks, the LRU matches the performance of deep SSMs like S4/S4D/S5. Detailed results in Sec. 5.3.*

eto et al., 2023c), we provide novel perspectives and careful ablations leading to new insights. **Each step presented in this chapter unveils a specific property of recurrent networks, and showcases the challenges and best practices in training deep RNNs**.

- **Linear Recurrences.** When replacing SSM layers in a deep architecture with vanilla RNN layers using tanh or ReLU activations, the performance on Long Range Arena (LRA) drops significantly. Surprisingly, in Sec. 5.3.1 we find that simply **removing the nonlinearities in the recurrence of the RNN** (i.e., using linear recurrences) gives a substantial boost in test accuracy. We motivate this effect in Sec. 5.5 by showing that stacking linear RNN layers and nonlinear MLP blocks (Fig.43) can model complex nonlinear sequence-to-sequence maps without requiring nonlinearities in the recurrence. While dropping the nonlinearity does not harm expressivity, it leads to several advantages, from the ability to directly control how quickly the gradients might vanish or explode, to allowing us to parallelize training. Our findings also mo-

tivate the success of deep SSMs, where the recurrence is also linear.

- **Complex Diagonal Recurrent Matrices.** Dense linear RNN layers can be reparameterized to a complex diagonal form without affecting the expressivity of the network or the features at initialization (Sec. 5.3.1 and Sec. 5.5). Diagonal linear RNN layers additionally allow for a **highly parallelizable unrolling** of the recurrence using parallel scans to substantially improve training speeds (Martin and Cundy, 2017). We validate that these observations, which have been leveraged by prior SSMs (Gupta et al., 2022a; Smith et al., 2022), also provide important efficiency improvements for linear RNN layers.

- **Stable Exponential Parameterization.** In Sec. 5.3.3 we show that using an exponential parameterization for the diagonal recurrent matrix has important benefits. Crucially, this enables us to easily enforce stability during training, which in turn allows us to modify the initialization distribution to facilitate long-range reasoning and improve performance. Our results indicate that rather than the specific deterministic initializations used by several recent SSMs, it is the eigenvalue distribution of the recurrent layer at initialization that determines whether or not the model can capture long-range reasoning.

- **Normalization.** In Sec. 5.3.4 we show that normalizing the hidden activations on the forward pass is important when learning tasks with very long-range dependencies. With this final modification, our RNNs can match the performance of deep SSMs on all tasks in the LRA benchmark. Connecting back to state-space models, we show in Sec. 5.6 how our normalization scheme can be linked to the discretization structure in S4.

We summarize the deep **Linear Recurrent Unit (LRU)** architecture used in this chapter, and the effect of each of the above steps on performance in Fig.43. We emphasize that the main purpose of our work is not to surpass the performance of S4-based models, but rather to demonstrate that simple RNNs can also achieve strong performance on long range reasoning tasks when properly initialized and parameterized. We believe the insights derived in this chapter can be useful to design future architectures, and to simplify existing ones.

## 5.2 A PRIMER ON RNNS AND THE S4 BLOCK

In this section, we compare the key architectural components (RNNs and SSMs) studied in this work, and also describe our methodology and experimental setup. For a more thorough discussion including related architectures, we refer the reader to our related work section in Orvieto et al., 2023c. We give an overview of the main architectural components considered in this chapter, focusing on the major differences between Vanilla RNNs and recent S4-like deep SSMs (Gu et al., 2021; Gu et al., 2022a; Gupta et al., 2022a; Smith et al., 2022).

VANILLA RNN RECAP. Let $(u_1, u_2, \dots, u_L)$ be an $H_{\text{in}}$-dimensional input tokens sequence, which can be thought of as either the result of intermediate layer computations (which keep the sequential structure) or as the initial input. An RNN layer with $N$-dimensional hidden state computes a sequence of $H_{\text{out}}$-dimensional outputs $(y_1, y_2, \dots, y_L)$ through a recurrent computation[1] using learnable parameters $A \in \mathbb{R}^{N\times N}, B \in \mathbb{R}^{N\times H_{\text{in}}}, C \in \mathbb{R}^{H_{\text{out}}\times N}, D \in \mathbb{R}^{H_{\text{out}}\times H_{\text{in}}}$:

$$x_k = \sigma(Ax_{k-1} + Bu_k), \quad y_k = Cx_k + Du_k, \tag{120}$$

starting from $x_0 = 0 \in \mathbb{R}^N$. $\sigma$ here denotes a nonlinearity, often chosen to be a tanh or sigmoid activation. If $\sigma$ is the identity function, then we say the RNN layer is *linear*.

S4-LIKE RECURRENT LAYER. We present a simplified[2] version of the S4 recurrence introduced in Gu et al., 2021. The input $(u_0, u_1, \dots, u_{L-1})$ is now seen as the result of sampling a latent continuous-time signal $u_{\text{ct}} : \mathbb{R}_{\geq 0} \to \mathbb{R}^{H_{\text{in}}}$ at multiples of a stepsize $\Delta > 0$: i.e. $u_{\text{ct}}(\Delta k) := u_k$ for all $k \in 0, \dots, L-1$. The output sequence $(y_0, y_1, \dots, y_{L-1})$ is then sampled,

1 We do not use bias parameters as they can be incorporated into the MLP blocks preceding and following the RNN block. Classical RNNs also included a nonlinearity on the output $y_k = \sigma_{\text{out}}(Cx_k + b)$ with $D = 0$. Having $D \neq 0$ basically introduces a skip connection, and the $\sigma_{\text{out}}$ can be thought of as part of the MLP following the RNN.

2 This version is most similar to S5 (Smith et al., 2022), but is here presented for ease of reasoning for a single discretization parameter $\Delta$, shared across input dimensions. See our related works in Orvieto et al., 2023c for details.

again with stepsize $\Delta$, from the signal $y_{\text{ct}} : \mathbb{R}_{\geq 0} \to \mathbb{R}^{H_{\text{out}}}$ computed by the following continuous-time state-space model, initialized at $x_{\text{ct}}(0) = 0$:

$$\begin{aligned} \frac{d}{dt} x_{\text{ct}}(t) &= \tilde{A} x_{\text{ct}}(t) + \tilde{B} u_{\text{ct}}(t), \\ y_{\text{ct}}(t) &= \mathfrak{Re}\left[\tilde{C} x_{\text{ct}}(t)\right] + \tilde{D} u_{\text{ct}}(t), \end{aligned} \tag{121}$$

where $\mathfrak{Re}(p)$ denotes the real part of a complex-valued vector $p$, $\tilde{A} = \text{diag}(\tilde{a})$ with $\tilde{a} \in \mathbb{C}^N$, $\tilde{B} \in \mathbb{C}^{N \times H_{\text{in}}}$, $\tilde{C} \in \mathbb{C}^{H_{\text{out}} \times N}$ and $\tilde{D} \in \mathbb{R}^{H_{\text{out}} \times H_{\text{in}}}$. Aside from the use of continuous time, the most striking differences compared to Eq.(120) are (a) the computation on the right-hand-side is *linear* in the hidden state and the input, and (b) most parameters are *complex* valued, with $\tilde{A}$ diagonal. While $\tilde{B}, \tilde{C}, \tilde{D}$ follow complex random or uniform initialization, the transition matrix $\tilde{A}$ is *structured*, i.e., initialized *deterministically* through HiPPO theory (Gu et al., 2020) in diagonal form. Common choices (Gu et al., 2022a) are $\tilde{a}_n = -\frac{1}{2} + i\pi n$ (S4D-Lin) and $\tilde{a}_n = -\frac{1}{2} + i\frac{N}{\pi}\left(\frac{N}{n+1} - 1\right)$ (S4D-Inv), for $n = 1, 2, \ldots, N$.

For training and inference, the continuous-time system in Eq.(121) is discretized at stepsize $\Delta$ using the Zero-Order-Hold (ZOH) or Bilinear method. The ZOH method gives

$$x_k = A x_{k-1} + B u_k, \quad y_k = C x_k + D u_k, \tag{122}$$

where $x_{-1} = 0$, $A = \exp(\Delta \tilde{A})$, $B = (A - I)\tilde{A}^{-1}\tilde{B}$, $C = \tilde{C}$ and $D = \tilde{D}$, and exp denotes the matrix exponential. Under the assumption that $u_{\text{ct}}$ is constant in between timestamps (which can be thought of as a modeling assumption), this numerical integration is *exact* (Jacquot, 2019). Moreover, note that all these discretization operations can be quickly performed element-wise since $\tilde{A}$ is diagonal.

KEY DIFFERENCES. We now highlight some of the most important differences between RNNs and SSMs:

- Since Eq.(122) is **linear**, it can be **efficiently parallelized** until $k = L - 1$ using parallel scans (Martin and Cundy, 2017; Smith et al., 2022), unlike a **nonlinear RNN** where the **computation has to be performed sequentially**.
- While Eq.(122) is similar to the linear RNN computation, we note that (a) $A$ and $B$ are **parameterized in a peculiar way** prescribed by discretization, and (b) these matrices share parameters (e.g. $\Delta$ affects

both $A$ and $B$). These differences are critical as in SSMs learning is performed on the continuous-time parameters $\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D}, \Delta$; hence parameterization choices directly affect optimization.

- Unlike RNNs, most SSMs use **complex-valued diagonal** recurrent matrices that are initialized deterministically using HiPPO theory. The literature attributes much of the success of SSMs to the specific initialized used (Gu et al., 2021; Gupta et al., 2022a; Gu et al., 2022b).

The points above motivate our investigation. We consider the same architecture as Gu et al., 2021; Gu et al., 2022a and Smith et al., 2022, but replace the SSM layer in the recurrent core by an RNN. We then study which steps need to be taken to gradually retrieve S4-like performance on LRA (Tay et al., 2020) tasks. The effectiveness of each of our steps is supported by empirical evidence and theoretical considerations, and leads to the architecture shown in Fig.43.

EXPERIMENTAL SETUP. We consider the Long Range Arena benchmark (Tay et al., 2020), a set of tasks designed to test the ability of models to do long-range sequence modelling (we use coloured images instead of grayscale images for the sequential CIFAR-10 classification task). Transformers fail to perform well on most of these tasks, while deep SSMs have shown remarkable performance on these tasks (Gu et al., 2021; Dao et al., 2022). This makes it an appropriate benchmark to explore the long-range modelling capabilities of deep RNNs.

For all experiments, we use a network of 6 layers with residual connections and layer/batch normalization (Ioffe and Szegedy, 2015; Ba et al., 2016) similar to Gu et al., 2021 (see Fig.43). We replace SSM layers with RNN layers with roughly the same number of parameters, building up to our LRU in a sequence of steps (see Sec. 5.3). All experiments are repeated three times, and we report the mean and standard error. We train using the AdamW optimizer (Loshchilov and Hutter, 2017), using a smaller learning rate and no weight decay on the recurrent parameters, as suggested by Steil, 2004; Gu et al., 2021. We tune hyperparameters such as learning rates for all models on a logarithmic grid for best accuracy. See the appendix of this thesis for more details.

| Recurrence | sCIFAR | ListOps | Text | Retrieval |
|---|---|---|---|---|
| RNN-ReLU | 69.7 (0.2) | 37.6 (8.0) | 88.0 (0.1) | 88.5 (0.1) |
| RNN-Tanh | 69.9 (0.3) | 43.9 (0.1) | 87.2 (0.1) | 88.9 (0.2) |
| RNN-Lin | **72.2** (0.2) | **50.4** (0.2) | **89.1** (0.1) | **89.1** (0.1) |

**Table 3:** *The effect of removing the nonlinearity from the recurrent unit on **test accuracy** (Sec. 5.3.1). We show results only for the sCIFAR, ListOps, Text and Retrieval, as these models did not exceed random guessing on PathFinder/PathX (further improvements in Tb.4&5). Performance of deep SSMs shown in Tb.5.*

## 5.3 HOW TO DESIGN TRAINABLE DEEP RNNS

In this section, we discuss the fundamental steps needed for designing RNNs to reach the impressive performance of deep SSMs on the LRA benchmark. We present these steps, already outlined in the introduction, in **logical order**, and support each claim with experimental evidence and theoretical considerations (discussed further in section Sec. 5.4).
We consider the architecture of Fig.43, where the recurrent computation is gradually modified starting from a vanilla RNN. We start by showcasing the advantage of using linear recurrences in Sec. 5.3.1; then, in the same section, we show how to speed-up training and inference without affecting expressivity or the initialization distribution. In Sec. 5.3.3, we discuss how (and why) changing the parameterization and initialization distribution enables us to make the RNN stable and improve long-range modeling. In Sec. 5.3.4, we finalize the LRU architecture by proposing a normalization strategy for the hidden activations that results in a close match in performance with SSMs.

### 5.3.1 *Linear RNN Layers are Performant*

One of the main findings of our work is that **linear RNN layers can be surprisingly expressive when coupled with nonlinear MLPs** or GLUs (Dauphin et al., 2017), outperforming tuned nonlinear RNN variants.
In Tb.3, we show that simply removing[3] the nonlinearity, and therefore computing the next state as $x_k = Ax_{k-1} + Bu_k$, is able to improve test

3 All other settings in the recurrent block match the Vanilla RNN Haiku module (Hennigan et al., 2020). All matrices have Glorot initialization (Glorot and Bengio, 2010). The overall architecture is as in Fig.43, with the LRU block replaced by an RNN.

accuracy on most LRA tasks. While the boost provided by vanilla linear RNN blocks leads to performance which is still far behind S4 on some tasks (sCIFAR, PathFinder and PathX), this first finding motivates us to drop nonlinearities in the recurrence for the rest of this chapter. In later sections, we leverage the linearity of the recurrence to significantly speed up training as well as derive principled initialization and normalization principles to learn long-range dependencies. We note that, on the Text and Retrieval tasks, performance using vanilla RNNs already matches performance of deep SSMs (see Tb.5 for the performance of S4/S4D/S5).

The empirical result in Tb.3 is **surprising**, since recurrent nonlinearities are believed to be a key component for the success of RNNs — both in theory and in practice (Siegelmann, 2012; Pascanu et al., 2013; Erichson et al., 2021). Indeed, a strong property of single-layer sigmoidal and tanh RNNs is **Turing completeness**, which cannot be achieved by the linear variant (Stogin et al., 2020; Chung and Siegelmann, 2021). However, the architecture we use (Fig.43) is deeper than a standard RNN and includes nonlinearies, placed position-wise *after* each RNN block. In Sec. 5.5, we investigate how the expressivity and trainability of deep models is affected by recurrent nonlinearities. **We prove** that linear RNNs interleaved with position-wise multi-layer perceptrons (MLPs) can approximate arbitrarily well any sufficiently regular nonlinear sequence-to-sequence map. The main idea behind our result is to see recurrent layers as compression algorithms that can faithfully store information about the input sequence into an inner state, before it is processed by the highly expressive MLP.

COMPLEX DIAGONALIZATION. We now show that we can significantly **speed up** training and inference for deep linear RNNs without losing performance by using **complex-valued diagonal** recurrent matrices. While the idea of diagonalizing linear systems for computational efficiency is a dominating feature of all deep SSMs, following the introduction of DSS by Gupta et al., 2022a, in this section we construct our diagonalized version to exactly match the initialization spectrum of the Glorot-initialized deep linear RNN in Tb.3. Our main purpose with this approach is to *disentangle the effects of initialization and diagonalization on performance*.

We start by recalling some useful linear algebra elements, and then proceed in Sec. 5.3.2 with a discussion on how to diagonalize the recurrence while preserving the eigenvalue spectrum at initialization.

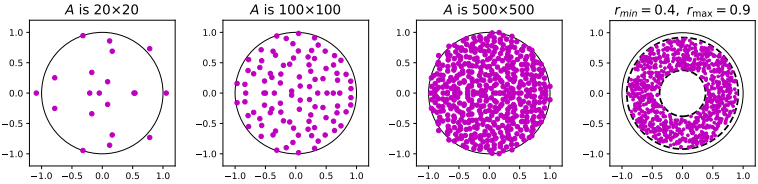


**Figure 44:** ***(First, Second, Third plots)*** *Eigenvalues of $A \in \mathbb{R}^{N\times N}$ following Glorot initialization: each entry of $A$ is sampled independently from a Gaussian (mean 0, variance $1/N$). The eigenvalues are complex ($A$ is not symmetric) and are represented on the complex plane. The black circle is the unit disk. The limit behavior (uniform initialization) is predicted by Thm. 5.3.1.* ***(Fourth plot)*** *Eigenvalues of a diagonal matrix $A$ with entries sampled using Lemma 5.3.1. For $r_{\min} = 0$, $r_{\max} = 1$, the distribution coincides with Glorot init. in the limit.*

The recurrence $x_k = Ax_{k-1} + Bu_k$ can be unrolled easily using the assumption that $x_{-1} = 0 \in \mathbb{R}^N$:

$$x_k = \sum_{j=0}^{k-1} A^j B u_{k-j}. \tag{123}$$

EFFECTS OF DIAGONALIZATION. Exponentiations of the matrix $A$ in the equation above are the source of the well-known **vanishing/exploding gradient** issue in RNNs (Bengio et al., 1994; Pascanu et al., 2013). While in nonlinear RNNs the state $x_k$ is forced to live on the compact image of the activation function, the hidden-state of our linear variant can potentially explode or vanish exponentially as $k$ increases. This phenomenon can be better understood by leveraging an eigenvalue (a.k.a. spectral) analysis: up to an arbitrarily small perturbation of the entries, every matrix $A \in \mathbb{R}^{N\times N}$ is diagonalizable[4] (Axler, 1997), i.e. one can write $A = P\Lambda P^{-1}$, where $P \in \mathbb{C}^{N\times N}$ is an invertible matrix and

$$\Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_N) \in \mathbb{C}^{N\times N}. \tag{124}$$

Unlike the symmetric setting where eigenvalues and eigenvectors are real, in the non-symmetric case[5] one has to allow for *complex* entries to

4 In other words, the set of non-diagonalizable matrices has measure zero, see e.g. Zhinan, 2002 for a proof idea.

5 Take e.g. $A = ((0,1)(-1,0))$. The solution to the standard eigenvalue equation gives $\lambda = \pm i$, where $i$ is the imaginary unit.

achieve full equivalence. Plugging the decomposition $A = P\Lambda P^{-1}$ into Eq.(123) and multiplying both sides by $P^{-1}$, we get

$$\bar{x}_k = \sum_{j=0}^{k-1} \Lambda^j \bar{B} u_{k-j}, \tag{125}$$

where $\bar{x}_k := P^{-1}x_k$, $\bar{B} := P^{-1}B$. The output can then be computed as $y_k = \mathfrak{Re}[\bar{C}\bar{x}_k] + Du_k \in \mathbb{R}^H$, where $\bar{C} = CP^{-1}$, and we take the real part of $\bar{C}\bar{x}_k$. Therefore, instead of learning $(A, B, C, D)$, one can equivalently learn $(\Lambda, \bar{B}, \bar{C}, D)$, where $\Lambda, \bar{B}, \bar{C}$ are complex valued, and $\Lambda$ is diagonal. For additional insights on the effectiveness of complex numbers in this setting, the reader is invited to check Section 5.5.

STABILITY. Since $\bar{x}_k = \sum_{j=0}^{k-1} \Lambda^j \bar{B} u_{k-j}$, the norm of component $j$ of $\bar{x}$ at timestamp $k$ evolves such that $|x_{k,j}| = O(|\bar{x}_{k,j}|) = O(|\lambda_j|^k)$. Therefore, a sufficient condition to ensure stability (i.e. $x_k$ does not explode) is therefore to require $|\lambda_j| < 1$ for all $j$ (Gu et al., 2021).

### 5.3.2 *Efficient Learning the Diagonalized Space*

Learning recurrent linear systems in diagonal form provides substantial computational speedups both for training and inference. For example, in our implementation of sCIFAR, we found diagonal linear RNNs to be $\sim$**8 times faster** to train than a dense RNN with ReLUs, matching the speed of our implementations of S4D and S5. The main reasons for this computational benefit are that (a) **taking powers** of diagonal matrices is **trivial** (speeding up both training (see also Li et al., 2018) and inference), while exponentiating dense matrices is computationally expensive, and (b) while nonlinear recurrences must be computed sequentially, unrolling a linear recurrence can be parallelized using **associative scans**, resulting in faster training (Smith et al., 2022) (App. E.1).

EQUIVALENT INITIALIZATION. To disentangle the benefits of diagonal linear systems from the role of **initialization**, we seek an initialization for the diagonal system which keeps the eigenvalue spectrum of the recurrence unchanged when comparing our diagonal system with the dense linear RNN in Sec. 5.3.1, where $A$ followed **Glorot initialization**. We use a classical result from random matrix theory (Ginibre, 1965):

**Theorem 5.3.1** (Strong circular law). *Let $\mu_N$ be the empirical spectral measure of $A_N \in \mathbb{R}^{N\times N}$, with i.i.d. Gaussian entries, each with zero mean and variance $1/N$. Then, $\mu_N$ converges weakly almost surely as $N \to \infty$ to the uniform probability measure on $\{|z| \leq 1\} \subseteq \mathbb{C}$.*

The theorem above (used also by Rajan and Abbott, 2006), illustrated in Fig.44, shows that under Glorot initialization the spectrum of a dense matrix $A$ is *de-facto* sampled from the unit disk in $\mathbb{C}$. This result motivates the strong performance of linear RNNs in Sec. 5.3.1, since it implies Glorot initialization provides an approximately stable initialization. Theorem 5.3.1 allows us to identify an **equivalent spectral initialization** for the diagonal system, which matches exactly for the large width limit: $\Lambda$ should be diagonal with entries sampled uniformly on the unit disk. Using the definition of exponential of a complex number: $\exp(-\nu + i\theta) := e^{-\nu}(\cos(\theta) + i\sin(\theta))$, we adopt a simple scheme for sampling on a ring in between circles with radii $r_{\min}$ and $r_{\max}$ in $\mathbb{C}$:

**Lemma 5.3.1.** *Let $u_1, u_2$ be independent uniform random variables on the interval $[0, 1]$. Let $0 \leq r_{\min} \leq r_{\max} \leq 1$. Compute*

$$\nu = -\frac{1}{2}\log\left(u_1(r_{\max}^2 - r_{\min}^2) + r_{\min}^2\right), \quad \text{and} \quad \theta = 2\pi u_2,$$

*then, $\exp(-\nu + i\theta)$ is uniformly distributed on the ring in $\mathbb{C}$ between circles of radii $r_{\min}$ and $r_{\max}$.*

We recover the spectrum of Glorot-initialization (in the limit of infinite width) by setting $r_{min} = 0$ and $r_{max} = 1$ (we will explore tuning these hyper-parameters in Sec. 5.3.3). Tb.4 (first two rows) shows the results of learning deep linear RNNs in complex diagonal form,[6] where each diagonal entry of $\Lambda$ is initialized uniformly on the unit disk in $\mathbb{C}$ using Lemma 5.3.1 with $[r_{\min}, r_{\max}] = [0, 1]$. In our experiments, $\bar{B}, \bar{C}$ (which we rename for convenience back to $B$ and $C$) follow Glorot initialization for both real and imaginary parts (parameterized separately), with halved variance in each component to preserve lengths on the input-output projections (Glorot and Bengio, 2010). Finally, after the SSM computation, only the real part is kept (as in Gupta et al., 2022a; Gu et al., 2022a).

6 To avoid issues with backpropagation on complex variables, each complex parameter in the network is stored and learned as a pair of floats encoding the real and imaginary parts.

Surprisingly, our results in Tb.4 show that diagonalizing the recurrence **improves accuracy** on tasks like ListOps and sCIFAR. More importantly, it **drastically reduces training and inference time** on all LRA tasks (see Tb.8 in Sec. E.1 for training speed comparisons), making the RNN just as fast to train as deep SSMs like S4D and S5.

### 5.3.3 *Benefits of Stable Exponential Parameterization*

In Sec. 5.3.1 we showed that moving to complex diagonal recurrences improved performance and is computationally efficient. However we also found that learning the diagonal model can be more unstable than learning the dense model in some experiments. To learn long-range dependencies and avoid vanishing gradients, eigenvalues in the recurrence need to have **magnitude close to 1** (Gu et al., 2022b; Gupta et al., 2022a); however, these large eigenvalues may also make the system **unstable** during training. In this section, we show the benefits of a **stable parameterization** of the RNN, and of tuning $r_{\min}$ and $r_{\max}$ (see Lemma 5.3.1).

EASIER OPTIMIZATION WITH EXPONENTIAL PARAMETERIZATION. Lemma 5.3.1 suggests a natural parameterization of the diagonalized RNN as $\Lambda = \text{diag}(\exp(-\nu + i\theta))$ with $\nu \in \mathbb{R}^N$ and $\theta \in \mathbb{R}^N$ as the learnable parameters (instead of the real and imaginary parts of $\Lambda$). As we explain in Sec. 5.4, this choice **decouples eigenvalue magnitude and oscillation frequency**, making **optimization with Adam easier**. The positive effects of this exponential parametrization, which resembles some features of ZOH discretization (see Sec. 5.2 and Sec. 5.6) can be observed in the third row of Tb.4. Notably, the performance on PathFinder rises above random chance for the first time.

ENFORCING STABILITY. An important benefit of the exponential parameterization is that it makes it **simple to enforce stability** on the eigenvalues. To see this, note that at initialization, $|\lambda_j| = |\exp(-\nu_j)| \leq 1$ since $\nu_j > 0$. Therefore, to preserve stability during training, we can use an exponential or another positive nonlinearity:

$$\lambda_j := \exp(-\exp(\nu_j^{\log}) + i\theta_j), \tag{126}$$

where $\nu^{\log} \in \mathbb{R}^N$ is the parameter we optimize, and we set $\nu_j^{\log} := \log(\nu)$ at initialization. Note that a similar idea is used in deep SSMs (Gu et al., 2021) in the context of discretization. We choose an exponential nonlinearity over a simple ReLU nonlinearity to increase granularity around $|\lambda| = 1$, achieved at $\nu^{\log} = -\infty$ (while $|\lambda| = 0$ is achieved at $\nu^{\log} = \infty$). Stable parameterization helps on most LRA tasks. In the fourth row of Tb.4, we show its effects on sCIFAR, ListOps and Pathfinder. We observe the most **drastic improvement** on Pathfinder, one of the harder tasks in LRA, where performance now reaches above 93%.

The benefits of the stable parameterization becomes more apparent when we explore the idea of **initializing** the eigenvalues of $\Lambda$ on a **ring closer to the unit disk** (increasing $r_{\min}$ closer to 1 in Lemma 5.3.1) to bias the network towards long range interactions and avoid vanishing gradients. As discussed in detail in Gu et al., 2022b; Gupta et al., 2022a, for tasks requiring consideration of interactions between distant tokens, eigenvalues in the recurrence need to have magnitude close to 1. Otherwise, as clear from the diagonal version of Eq.(123), when taking powers of eigenvalues close to the origin, the signal from past tokens quickly dies out. However, as we show in Orvieto et al., 2023c, without enforcing stability performance starts to degrade as we increase $r_{\max}$ past 0.9 in the sCIFAR task. With stability enforced, we can increase $r_{\max}$ up to 0.99 and improve performance. We see similar benefits on the other tasks where we sweep different values of $r_{\min}$ and $r_{\max}$.

Finally, note that while we explore changing the magnitude of the eigenvalues of $\Lambda$ in this section, in Sec. 5.3.4 we also show the benefits of initializing the eigenvalues to have a small phase to learn more global patterns, which is useful on very long-range reasoning tasks (e.g. PathX).

### 5.3.4 *Additional Considerations for Long-range Tasks*

Up to this point, our model did not succeed on PathX, the hardest dataset in the LRA benchmark, with a sequence length of $16k$ tokens. In this section, we discuss two final modifications which improve our model's ability to learn very long-range dependencies, and finalize our LRU model.

NORMALIZATION. In Sec. 5.3.3, we initialized the eigenvalues of $\Lambda$ close to the unit disk for better performance on long-range tasks. However, we observed that **as we moved $r_{\min}$ and $r_{\max}$ closer to 1, the train-**

| | sCIFAR | ListOps | Pathfinder |
|---|---|---|---|
| Dense $A$ | 72.2 (0.2) | 50.4 (0.2) | ✗ |
| $\Lambda$ Real + Im | 86.5 (0.1) | 58.8 (0.3) | ✗ |
| $\Lambda$ Exp | 85.4 (0.7) | **60.5** (0.3) | 65.4 (9.0) |
| $\Lambda$ Stable Exp | 87.2 (0.4) | 59.4 (0.3) | 93.5 (0.5) |
| + Ring Init | **88.1** (0.0) | 59.4 (0.3) | **94.4** (0.3) |

**Table 4:** *Test accuracy of linear diagonal complex RNNs under different parametrizations of the transition matrix (see Sec. 5.3.1). These results improve on the results in Tb.3, and showcase the advantage of an exponential (polar) representation for $\Lambda$. Ring Init denotes a changed initialization where $r_{\min}$ and $r_{\max}$ are tuned. Performance on the Text and Retrieval tasks is not shown as linear RNNs already match the performance of S4 (c.f. Tb.3 with Tb.5).*

**ing loss also started to blow up at initialization** (see Fig.45). In this section, we first present a result explaining this phenomenon, before deriving a practical normalization scheme for the hidden activations to tackle this problem and further improve performance.

**Theorem 5.3.2** (Forward-pass blow-up)**.** *Let $\Lambda$ be diagonal with eigenvalues sampled uniformly on the ring in $\mathbb{C}$ between circles of radii $r_{\min} < r_{\max} < 1$. Under constant or white-noise input and Glorot input projection, the squared norm of the state $x_k$ converges as $k \to \infty$ to:*

$$\mathbb{E}[\|x_\infty\|_2^2] = \frac{1}{r_{\max}^2 - r_{\min}^2} \log\left(\frac{1 - r_{\min}^2}{1 - r_{\max}^2}\right) \mathbb{E}[\|Bu\|_2^2].$$

This result (related to similar propositions for Echo-State Networks Couillet et al., 2016) has the following intuitive form if $r_{\min} = r_{\max} = r$: if we initialize $\rho$-close to the unit disk, the forward pass blows up by a factor $1/\rho$ (since the contributions from previous states will take longer to decay). Let $\epsilon = r_{\max}^2 - r_{\min}^2$ and $\rho = 1 - r_{\max}^2$, then:

$$\lim_{\epsilon \to 0} \frac{\mathbb{E}[\|x_\infty\|_2^2]}{\mathbb{E}[\|Bu\|_2^2]} = \lim_{\epsilon \to 0} \left[\frac{1}{\epsilon} \log\left(1 + \frac{\epsilon}{\rho}\right)\right] = \frac{1}{\rho} = \frac{1}{1 - r^2}. \tag{127}$$

Towards the derivation of an **effective normalization** scheme for the forward pass, we present a simplified derivation of the $1/\rho$ gain formula

| | sCIFAR | ListOps | Text | Retrieval | Pathfinder | PathX |
|---|---|---|---|---|---|---|
| **LRU** | 89.0 | 60.2 | 89.4 | 89.9 | 95.1 | 94.2 |
| S4 | *91.1* | *59.6* | *86.8* | *90.9* | *94.2* | *96.4* |
| S4D | 89.9 | 60.5 | 86.2 | 89.5 | 93.1 | 91.9 |
| S5 | *90.1* | *62.2* | *89.3* | *91.4* | 95.3 | *98.6* |

**Table 5:** *Performance of the final LRU architecture after adding $\gamma$ normalization to the diagonal RNN with stable exponential parameterization and initialization on the ring (see Sec. 5.3.4). For PathX, we additionally use a smaller eigenvalue phase at initialization. We sweep $r_{\min}$, $r_{\max}$ and learning rate. We report results from the S4/S4D and S5 papers for comparison. The LRU reaches similar performance to deep SSMs on all LRA tasks..*

for the one-dimensional setting under white-noise input[7]: let $\Lambda = \lambda \in \mathbb{C}$, and $B = 1$. Let $p^*$ denote the conjugate of $p \in \mathbb{C}$, we have that $|p|^2 = p^* p$ and in expectation over the input, using Eq.(123) and the fact that $\mathbb{E}[u_{k-i} u_{k-j}] = 0$ for $i \neq j$:

$$\mathbb{E}|x_k|^2 = \mathbb{E}\left[\left(\sum_{i=0}^{k-1} \lambda^i u_{k-i}\right)\left(\sum_{j=0}^{k-1} \lambda^j u_{k-j}\right)^*\right] \tag{128}$$
$$= \sum_{i,j=0}^{k-1} \lambda^i (\lambda^j)^* \mathbb{E}[u_{k-i} u_{k-j}] = \sum_{i=0}^{k-1} |\lambda|^{2i} \overset{\infty}{\to} \frac{1}{1-|\lambda|^2}.$$

Since the formula above holds for every Euclidean direction in our recurrence ($\Lambda$ is diagonal), we add a normalization parameter that is initialized element-wise. Note as $\lambda$ approaches 1, $(1-|\lambda|^2)$ approaches 0, making further adaptations of this parameter via SGD hard. We therefore use normalization parameter $\gamma^{\log} \in \mathbb{R}^N$, initialized element-wise as $\gamma_i^{\log} \leftarrow \log(\sqrt{1-|\lambda_i|^2})$,[8] and modify the recurrence as:

$$x_k = \Lambda x_{k-1} + \exp(\gamma^{\log}) \odot (B u_k), \tag{129}$$

where $\odot$ denotes the element-wise product. The $\gamma$ parameter enables the

7 We use the random input assumption for our normalization scheme as we found it to work well in practice.

8 We also tried setting $\gamma_i$ to $\sqrt{1-|\lambda_i|^2}$ in each training iteration, and found it to work similarly to a trainable $\gamma$.

RNN to adaptively scale the input for each eigendirection. This consistently improves performance on tasks that benefit from initializing close to the unit disk, such as sCIFAR and Pathfinder, as shown in Tb.5.

REDUCING EIGENVALUE PHASE AT INITIALIZATION. We have $\Lambda = \text{diag}(\exp(-\exp(\nu^{\log}) + \theta))$, where $\nu^{\log} \in \mathbb{R}^N$ is the vector of log eigenvalue magnitudes and $\theta \in \mathbb{R}^N$ the vector of eigenvalue *phases*. While $\nu^{\log}$ encodes the distance to the origin, $\theta$ is the angle from the vector $1 + 0i$. **For long sequences**, initializing $\theta \sim [0, 2\pi]$ uniformly implies that most state entries will exhibit a large number of oscillations at initialization (see upper panel in Fig. 46). Equivalently, in this setting, most state dimensions are the result of **convolutions**[9] capturing an average of **local oscillation patterns**. This behavior is independent from the ability of capturing long-range dependencies (controlled by $\nu^{\log}$), but pertains to the nature of the information stored by the RNN. Therefore, initializing $\Lambda$ with uniform phase on long sequence data inherently biases the network towards learning spurious features in the input sequence. The model cannot recover from this suboptimal initialization. Indeed we observe that, for our best model so far on PathX, the training loss after a few iterations converges to a suboptimal minimizer with random chance test performance (see Fig.45). To fix this issue, we found it sufficient to restrict the range of $\theta$ to a thin slice around 0, biasing the model towards learning **global features**. Since the optimal values of $\theta$ are small, we parameterize the phase logarithmically: $\theta = \exp(\theta^{\log})$, where $\theta^{\log}$ is optimized.

Restricting the range of the phase at initialization to be $[0, \pi/10]$, our model achieved **94.2% on PathX**, aligning with state-of-the-art deep SSMs. We did not explore using a smaller phase at initialization for the other LRA tasks, although we believe this might further improve performance here as well. Note that both $\gamma$ normalization and restricting the eigenvalue phase at initialization were crucial to solving PathX. We were unable to learn when using restricted phase at initialization without also introducing $\gamma$ normalization.

With all the components of Sec. 5.3 taken together, we name this new model the **Linear Recurrent Unit** (or **LRU** for short). It provides a flexible, interpretable, and theoretically principled framework for initializing

9 See Gu et al., 2022a for a discussion of kernel perspectives.

and learning deep RNNs efficiently, and it matches performance and efficiency of deep SSMs across all the LRA tasks, as shown in Tb.5.

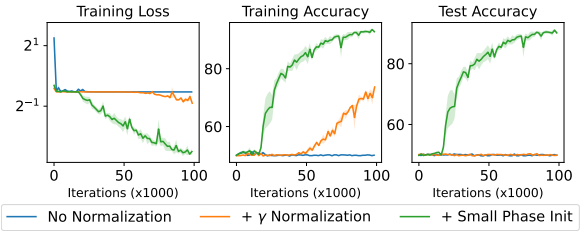


**Figure 45:** *Effect of normalization and using a small phase at initialization on the PathX task. Without normalization, the model presents higher loss values at initialization and quickly converges to a suboptimal value, where train and test accuracy are both at random chance. Adding normalization helps: the train loss is lower at initialization, and the optimizer is able to escape the suboptimal region and train accuracy also increases. Interestingly, this model still fails to generalize. Finally, reducing initialization phase (tuning the range of θ) dramatically improves convergence on the training set, while also generalizing to the test set.*

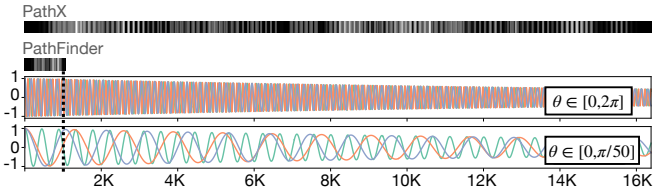


**Figure 46:** *Evolution of $x \in \mathbb{R}^3$ under input $u = (1,0,0,\dots,0) \in \mathbb{R}^{16k}$. Plotted in different colors are the 3 components of $x$. $\Lambda$ has parameters $\nu_j = 5e^{-5}$ and $\theta_j$ sampled uniformly in $[0,2\pi]$ or $[0,\pi/50]$. For short sequences, such as $L = 1024$ (PathFinder), $[0,2\pi]$ produces kernels with an acceptable number of oscillations: information about $u_0$ is recalled only a few times in the state history. For large L (PathX), the range of the imaginary part at initialization has to be smaller to obtain a similar effect.*

## 5.4 OPTIMIZATION OF DEEP RNNS

In this subsection we back-up some of our claims about optimization of linear RNNs in Section 5.3 with experimental findings on toy examples. Our purpose is to confirm validity of our intuition in the absence of architecture-dependent confounders: i.e on vanilla RNNs with one layer.

**Recurrent nonlinearities slow gradient descent.** In Section 5.3 and Section 5.5 we showed how linear RNNs can be used as elementary recurrent blocks for the purpose of modeling nonlinear dynamics when stacked in deep architectures. Similarly, the results in Li et al., 2022a indicate that, to achieve S4 performance, one can equivalently replace the recurrent core with a collection of convolutions parametrized by filters.
While a **single-layer** level, a (dense) RNNs (Eq.120) with tanh or sigmoid activation can express convolutions with filters (Wang et al., 2022b), the results in Tb. 3 (and Fig. 1(a) in Wang et al., 2022b) indicate an advantage on test accuracy from dropping such nonlinearities in the recurrence — i.e. of making the RNN linear. Motivated by this, in Fig. 47 we consider the problem of **learning a single one-dimensional convolution** kernel with a single layer RNN, and compare performance of linear and tanh activations. The sequence length in this problem was 100, and our data consists in 32 input-output one-dimensional trajectories, where the output is the result of a convolution with the kernel of elements $h_k := \frac{1}{10}\exp(-0.015 \cdot k)\cos(0.04 \cdot k)^2$, which induces moderate-length dependencies in the data (see bump in the kernel in Figure 47 at $k = 70$). The 32 input sequences are generated sampling random $a, c$ parameters on a range and have form $\sin(0.05 \cdot a \cdot k)\cos(0.05 \cdot c \cdot k)^2$. Outputs are generated by convolving each input by $h$. Learning is performed using Adam (Kingma and Ba, 2014) with standard $\beta_1, \beta_2$ parameters.
Already on this simple task, **linear RNNs outperform the** tanh **variant** even after careful tuning of the stepsize. While the input-output map the system had to approximate is linear (i.e. a convolution), this result still indicates that on deep architectures, where the MLPs interleaving RNNs can quickly perform position-wise nonlinearities lifting the function approximation class (see Section 5.5), linear RNNs are preferrable.

**Exponential Parameterization makes the landscape axis-aligned.** Our experimental results in Section 5.3.3 indicate that linear RNN cores can

be more effectively learned under exponential parameterization of the eiganvalues: $\lambda = \exp(-\nu + i\theta)$. To understand the reason behind this phenomenon, we go back at the classical (hard) problem of **learning powers** (Bengio et al., 1994), crucially linked with linear RNN models (see Eq. (123)). For a specific planted solution

$$\lambda^* = \lambda_r^* + i\lambda_i^* = \exp(-\nu^* + i\theta^*), \tag{130}$$

we consider the problem of minimizing the loss $L(\hat{\lambda}) = \frac{1}{2}|\hat{\lambda}^k - (\lambda^*)^k|^2$, where $k = 100$ and $\hat{\lambda}$ is generated from two real parameters following standard ( real + imaginary) or exponential parameterization. Note that in this paragraph $\lambda^* \in \mathbb{C}$ denotes the solution, not the complex conjugate of $\lambda$. In Fig. 48, we show that as the target phase $\theta^*$ approaches $\pi/2$ (i.e. $\lambda^*$ gets close to the imaginary axis), **standard parameterization slows down learning**, as the corresponding landscape gets non-axis-aligned — a feature that does not match well the **inner workings of the Adam optimizer**[10], which is a diagonal preconditioner (Kingma and Ba, 2014). Instead, under exponential parameterization, the effects of phase and magnitude parameters on the powers of $\lambda$ are more efficiently decouped: for example, while the real part of $\lambda^k$ is simply $\exp(-k\nu)$ using exponential parameterization, if standard parameterization is used, $\Re\left[\lambda^k\right]$ is a function of both $\lambda_r$ and $\lambda_i$. We noticed that the performance difference gets most pronounced when the system has to learn how to "turn": i.e. the initialization magnitude is correct, but the position on the complex plane is not (this is the precise setting for Figure 48): while for standard parameterization changing the phase $\theta^*$ requires a careful balance between real and imaginary components, for exponential parameterization gradients are fully aligned with the phase parameter. This makes the learning more flexible, a feature which we observed necessary in our experiments on the Long Range Arena, see Section 5.3.3 and Tb.4.

10 For this problem, vanilla gradient descent cannot be effectively used as the landscape is highly non-convex, with challenging curvature vanishing as $|\lambda|$ approaces 0.

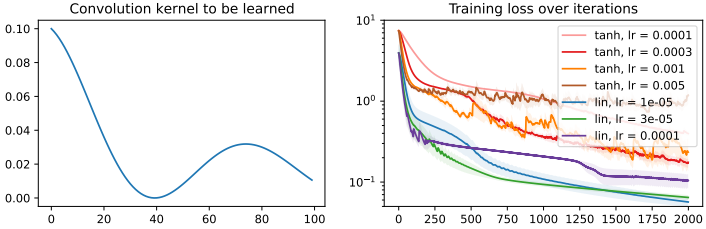


**Figure 47:** *Learning with Adam a one-dimensional convolution with a length-*100 *kernel using a single-layer RNNs with linear or* tanh *recurrent activations and* 100*-dimensional hidden state. Initialization is performed using Glorot on all quantities for both options. For all learning rates in our grid, the linear variant is faster to converge.*

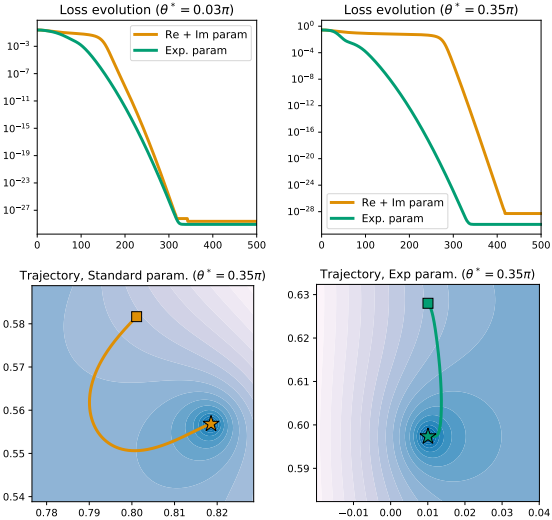


**Figure 48:** *Exponential parametrization helps when learning a single complex eigenvalue* $\lambda^* = \exp(-\nu^* + i\theta^*)$*, exponentiated* 100 *times. As* $\lambda^*$ *gets close to the purely imaginary setting* $\theta^* = \pi/2$*, the geometry of the loss landscape under standard real+imaginary parametrization becomes suboptimal for the Adam optimizer, which works best in the axis-aligned setting (exponential parametrization). In the plot, the square denotes initialization , while the star denotes the solution after* 500 *iterations.*

## 5.5 UNIVERSALITY OF LINEAR RNNS + NONLINEAR PROJECTIONS

> *For millennia under the name of logic, logic itself and the doctrine of knowledge have been confused: representing is confused with representing (with the represented as such), thinking with thinking (with meaningful sense and proposition), knowing with knowledge (with evident and grounded truth).*
>
> – Edmund Husserl.

In this section, based on our recent paper (Orvieto et al., 2023a), we elaborate on a surprising finding of Section 5.3: recurrent nonlinearities are not necessary for performance — linear recurrences are enough. We show here that a family of sequence models based on **recurrent *linear* layers** (including S4, S5 and the LRU) **interleaved with position-wise multilayer perceptrons** (MLPs) **can approximate arbitrarily well any sufficiently regular nonlinear sequence-to-sequence map**. The main idea behind our result is to see **recurrent layers as compression algorithms** that can faithfully store information about the input sequence into an inner state, before it is processed by the highly expressive MLP.

### 5.5.1 *Background and Notation*

Consider length-$L$ sequences of real $M$-dimensional inputs: $(u_i)_{i=1}^L \in \mathbb{R}^{M\times L}$. A nonlinear causal sequence-to-sequence map produces a sequence of real $M_{\text{out}}$-dimensional outputs $(y_i)_{i=1}^L$ as

$$y_k = T_k[(u_i)_{i=1}^k], \tag{131}$$

where $T_k : \mathbb{R}^{M\times k} \to \mathbb{R}^{M_{\text{out}}}$ is a nonlinear map. Here, we consider an approximation of this map using deep networks with linear diagonal RNNs interleaved with position-wise MLPs (see Fig. 49).

RELATED WORKS. The approximation properties of deep neural networks with ReLU activations are well studied. While recent advances concern the effect of depth (Lu et al., 2017), the study by Pinkus, 1999, as well as previous works (Funahashi, 1989; Hornik et al., 1989; Hornik, 1991), already established the power of neural networks with a single hidden layer, which can approximate arbitrary continuous nonlinear maps

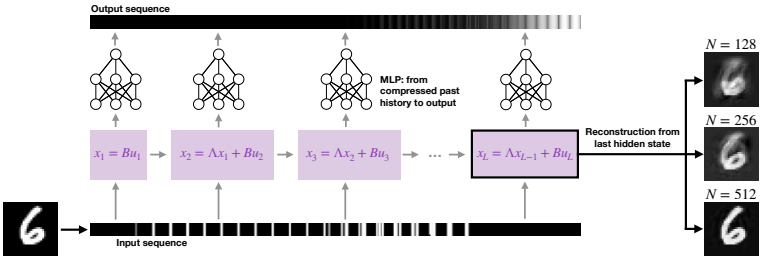


**Figure 49:** *Linear RNN + position-wise MLP on flattened MNIST (LeCun, 1998) digits. The role of the linear RNN is to compress (if possible) and store the input sequence into the hidden state: from $x_L \in \mathbb{C}^N$ one can recover past tokens using a linear transformation. As the state size $N$ increases, the reconstructions becomes more and more faithful. The MLP (same for all tokens) takes this representation as input, and is able to produce any output sequence of the form of eq.(131).*

on compacts as the size of the hidden layer grows to infinity. This result can be also used in the context of nonlinear RNN approximation of dynamical systems (e.g. in neuroscience), where the state can be seen as part of an MLP (see e.g. Hanson and Raginsky, 2020): compared to Eq. (125) we have $x_k = \sigma(Ax_{k-1} + Bu_k)$, where $\sigma$ is is a nonlinearity. Meanwhile linear RNNs, where $x_k = Ax_{k-1} + Bu_k$, have often been considered of minor interest, equivalent in approximation power to convolutions (Li et al., 2022b). However, we focus on a restricted scenario of dealing with finite length sequences, and show that, when sufficiently wide, the linear RNN *does not* form a bottleneck, and the architecture maintains universality through the application of the pointwise MLP, as done in recent SSMs achieving state-of-the-art results (Gu et al., 2021; Smith et al., 2022; Orvieto et al., 2023c). We give a detailed overview in the appendix of Orvieto et al., 2023a.

OUR PROOF STRATEGY. Let us start again from Eq. (123):

$$x_k = \sum_{j=0}^{k-1} \Lambda^j B u_{k-j}. \tag{132}$$

One linear RNN+MLP layer therefore computes the sequence $(y_k)_{k=1}^L$ as $y_k = \phi\left(\sum_{j=0}^{k-1} \Lambda^j B u_{k-j}\right)$, where $\phi : \mathbb{C}^N \to \mathbb{R}^M$ is a nonlinear map

parameterized by an MLP. Under some assumptions, we show in this section that $\sum_{j=0}^{k-1} \Lambda^j B u_{k-j}$ captures all information about the input up to step $k$. As a result, $\phi$ can effectively parametrize any **regular enough** nonlinear sequence-to-sequence map from past inputs to outputs:

$$\phi\left(\sum_{j=0}^{k-1} \Lambda^j B u_{k-j}\right) \stackrel{!}{=} T_k((u_i)_{i=1}^k). \tag{133}$$

### 5.5.2 *RNNs Perform Compression*

We start with a simple one-dimensional example. Let $M = 1$ and $B = (1, 1, \dots, 1)^\top \in \mathbb{R}^{N \times M}$. Then, Eq. (123) can be written as a matrix multiplication, recalling $\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_N)$, with $\lambda_i \in \mathbb{C}$:

$$x_k = \begin{pmatrix} \lambda_1^{k-1} & \lambda_1^{k-2} & \cdots & \lambda_1 & 1 \\ \lambda_2^{k-1} & \lambda_2^{k-2} & \cdots & \lambda_2 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \lambda_N^{k-1} & \lambda_N^{k-2} & \cdots & \lambda_N & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_k \end{pmatrix} = V_k u_{1:k}. \tag{134}$$

VANDERMONDE PSEUDOINVERSION. As long as $N \geq k$, we can hope to recover $u_{1:k}$ by pseudoinversion of the Vandermonde matrix $V_k$:

$$u_{1:k} = V_k^+ x_k. \tag{135}$$

Indeed, note that at the boundary setting $N = k$ (number of equations = number of unknowns), the matrix $V_N$ is **invertible** since $\det(V_N) = \prod_{1 \leq i < j \leq N} (\lambda_i - \lambda_j) \neq 0$ with probability one under e.g. uniform on the complex unit disk initialization (proposed in Orvieto et al., 2023c, see Lemma 5.3.1).

RECONSTRUCTION OF THE TIMESTAMP. Given a generic $x \in \mathbb{C}^N$, how do we know the length of the vector $u_{1:k}$ to reconstruct? Fortunately, architectures such as S4, S5 and LRU include an **encoding layer** before the linear RNN. Let us consider an embedding such that $u_k$ is mapped

to $(1, u_k)$ for all $k$. Let $B = ((1,0,0,\dots,0),(0,1,1,\dots,1))^\top$, we get $x_k = \left(\sum_{i=0}^{k-1} \lambda_0^i,\ \tilde{x}_k\right)$, where $\tilde{x}_k$ follows Eq. (134). To reconstruct $k$ from $\tilde{x}_k$,

$$x_{k,0} = \sum_{i=0}^{k-1} \lambda_0^i = \frac{\lambda_0^k - 1}{\lambda_0 - 1} \quad \Longrightarrow \quad k = \log_{\lambda_0}(x_{k,0}(\lambda_0 - 1) + 1). \tag{136}$$

TAMING ILL-CONDITIONING WITH COMPLEX NUMBERS. From the Vandermonde determinant formula, it is clear that $V_k$ is likely ill-conditioned under random initialization, which makes it hard to implement the inversion in practice. However, while if $\Lambda$ is real the condition number of $V_k$ grows exponentially with $N$ (Gautschi and Inglese, 1987), in the complex case it is possible to make the condition number of $V_k$ exactly 1 by e.g. choosing the $N$th-roots of unity as eigenvalues of $\Lambda$ (Gautschi, 1975; Córdova et al., 1990). This fact provides a **precise justification for the use of complex numbers in the recurrent computation**. We explore this topic experimentally in the appendix of Orvieto et al., 2023a, where we also show performance increase in reconstruction of MNIST digits (LeCun, 1998) when we move initialization close to the boundary of the unit disk (see Fig. 50), as observed in practice in all recent works on state-space models (Gu et al., 2021; Gupta et al., 2022b; Smith et al., 2022; Orvieto et al., 2023c). We note that the discussion here opens up interesting directions for future investigations, such as the derivation of an optimal $\Lambda$ for reconstruction using the Vandermonde inverse. For the time being, from our experiments (see e.g. Fig. 50), we conclude that a setting which safely leads to perfect reconstruction without ill-conditioning is, empirically, $N \gg L$ with initialization of $\Lambda$ close to the unit disk.

ROLE OF SPARSENESS. While the set $(V_k^+)_{k=1}^L$ is **numerically stable** only for $N$ considerably larger than $L$, we note that it is not surprising that, in order to memorize a sequence of $L$ (one-dimensional) inputs one needs a hidden state which is at least as big as $L$. From an information theory perspective, one can only have $N < L$ if the set of inputs is a structured subset of $\mathbb{R}^L$ – i.e. only if the **input can be compressed**. Is it useful to note that, in the setting of the last subsection, the role of the RNN has mainly been ensuring no information is lost: the final hidden state provides a direct copy of the inputs, with no further reasoning.

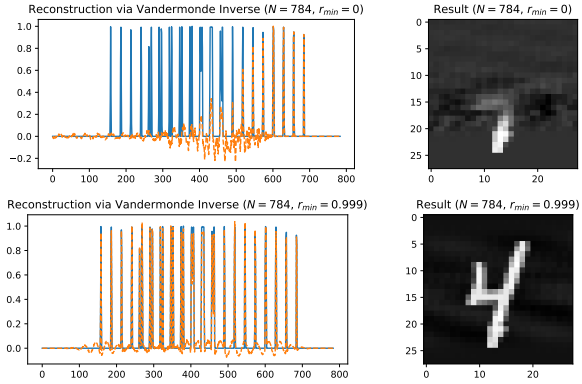


**Figure 50:** *Reconstruction of MNIST digits from the final linear RNN hidden state. For* $r_{\min} = 0$ *(left hand side), the Vandermonde is ill-conditioned (see appendix in Orvieto et al., 2023a) and hence only the recent past (ie lower half of the image) can be reconstructed. For* $r_{min} = 0.99$ *we can reconstruct the whole image.*

Towards fixing this, leveraging ideas from sparse coding, compressed sensing (Candes et al., 2006), and echo state networks (Jaeger and Haas, 2004), we introduce the following assumption.

**Assumption 5.5.1.** *Consider a proper collection of ordered basis matrices* $(\psi^i)_{i=1}^P$*, with* $\psi^i \in \mathbb{R}^{M\times L}$ *for all* $i = 1, 2, \cdots, P$*. Let* $u = (u_{:,1}, u_{:,2}, \cdots, u_{:,L}) \in \mathbb{R}^{M\times L}$*, be a sample sequence from the input sequence distribution. For any* $k \in \{1, 2, \ldots, L\}$ *one can always write* $u_{:,1:k} = \sum_{i=1}^P \alpha_i^{u,k} \psi^i_{:,1:k}$*, where* $\psi^i_{:,1:k}$ *are the first* $k$ *columns of* $\psi^i$*, and* $(\alpha_i^{u,k})_{i=1}^P$ *is a sequence of real numbers. We do not assume such decomposition is unique, nor we assume we have access to the basis functions.*

The idea behind leveraging this is representation is that one can assume $P \ll L$, and reduce the task of estimating $u$ from $x$, to the task of estimating $\alpha$ from $x$. In formulas, since for every $k \in \mathbb{N}$, $u_{1:k} = \Psi_k \cdot \alpha^{u_{1:k}}$, with $\Psi_k \in \mathbb{R}^{k\times P}$ and $\alpha^{u_{1:k}} \in \mathbb{R}^P$, we have

$$\tilde{x}_k = V_k u_{1:k} = \underbrace{V_k \cdot \Psi_k}_{\Omega_k} \cdot \alpha^{u_{1:k}}, \tag{137}$$

where $\Omega_k \in \mathbb{C}^{(N-1)\times P}$. Under the assumption that $\Omega_k$ has column rank greater than row rank, i.e., need $N > P+1$, we are able to solve the system for $\alpha^{u_{1:k}}$. As before, since $x_{k,0}$, the first coordinate of $x_k$, can be used to recover $k$, the procedure above gives a **unique way to recover an arbitrary-length signal** $u$ sparse in the basis $\Psi$, through a simple operation on the linear RNN output.

$\Psi$ CAN ALSO BE LEARNED. One does not need to assume a specific $\Psi$ a priori: as long as we have **input sparsity**, one can guarantee approximation with a linear operator $\Omega_k^+$ on the state $x_k$, which can be learned and may be **task-dependent**. Note that even if the input is strictly-speaking not sparse, not all information might be relevant for prediction. In a way, sparsity mathematically quantifies the belief that the information content relevant in a certain task has fewer bits than the original signal.

SPARSENESS IMPROVES CONDITIONING. In the last section, we discussed ill-conditioning of $V_k$. We show with a simulation in the appendix of Orvieto et al., 2023a that if $P \ll L$ then for the matrix $\Omega_k$ we have no computational issues when calculating its pseudo-inverse (reconstruction map is "easy").

MULTIDIMENSIONAL SETTING. We provide a discussion for the multidimensional setting in the appendix of Orvieto et al., 2023a.

### 5.5.3 *Role of the MLP*

In the last paragraphs, we showed how linear recurrences can be used as compression algorithms, with guarantees of perfect reconstruction under suitable assumptions on the RNN size and information content in the input. Mathematically, in the settings described in the last subsection, there exists a function $\gamma$ such that $x_k \xrightarrow{\gamma} ((u_i)_{i=1}^k, k)$ for all $k = 1, 2, \dots, L$. Then, intuitively, since we can perfectly reconstruct the input sequence from $x_k$, we can let the **MLP parametrize any nonlinear function on this subsequence**, indexed by $k$. In formulas:

$$y_k = T_k[(u_i)_{i=1}^k] = \bar{T}((u_i)_{i=1}^k, k) = \bar{T}(\gamma(x_k)) = \phi(x_k). \tag{138}$$

We can then let $\phi$ be parametrized by an MLP. Indeed, as it is well known, as the MLP size grows, it can approximate arbitrary nonlinear map on compact sets (in our case, $\bar{T} \circ \gamma$). To make the argument behind Eq. (138) precise, we need the following assumption.

**Assumption 5.5.2.** *The input sequences $(u_i)_{i=1}^L$ live on a compact set $U \subset \mathbb{R}^{M \times L}$. The maps $T_k : \mathbb{R}^{M \times k} \to \mathbb{R}^M$, producing the output sequence $(y_i)_{i=1}^L$ as $y_k = T_k[(u_i)_{i=1}^k]$, are continuous on $U$.*

We can then summarize the findings of this chapter in an informal way as follows. A proof can be found in Orvieto et al., 2023a.

**Theorem 5.5.1** (Informal)**.** *Assume finite-length sequence data is generated from a sequence-to-sequence model such that Assumption (5.5.2) holds. Consider a randomly initialized linear recurrent network of size $N$, with $N$ large enough depending on the sparseness of the input set (see Assumption (5.5.1) and discussion on initialization). Then, there exists a wide enough MLP which, if applied pointwise to the RNN hidden states, leads to perfect reconstruction of the system's output.*

We provide a numerical validation of our claim on a nonlinear controlled Lotka-Volterra system in Figure 51. As clear from the results, the MLP was able to translate the input tokens representations into the correct values of the output sequence.

**Current research.** Our current research objective is to provide a non-asymptotic version of the theorem above (potentially in the deep setting), as well as more thorough experimental evaluations, addressing more precisely potential limitations and issues such as bad conditioning.

## 5.6 INSIGHTS ON S4 AND VARIANTS

We believe our ablations in Sec. 5.3 **explain the underlying mechanisms driving the success of deep SSMs**. To conclude the chapter, we inspect in detail the main similarities and differences between our LRU model and diagonal SSMs, and elaborate a few insights. As in Sec. 5.2, to avoid technicalities, we provide a simplified discussion capturing the main features of models stemming from the original S4 paper.
As detailed in Sec. 5.2, diagonal SSMs (DSS, S4D, S5) are instantiated and parameterized through the **discretization** of a latent continuous-

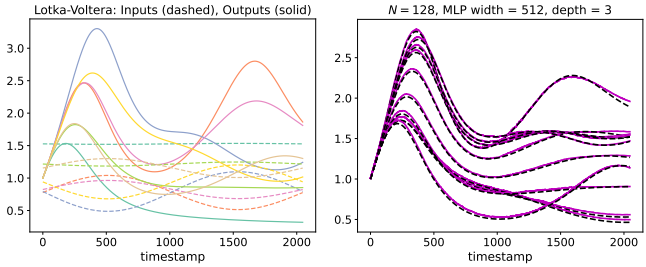


**Figure 51: (Left)** example input-output trajectories (dashed are inputs, solid are outputs) as well as **(right)** validation performance in predicting outputs from unseen input sequences (dashed is ground truth).

time model $\dot{x}_{\text{ct}}(t) = \tilde{A}x_{\text{ct}}(t) + \tilde{B}u_{\text{ct}}(t)$, where $A = \text{diag}(\tilde{a})$ is initialized with complex entries, often guided by HiPPO theory (Gu et al., 2020). Zero-Order-Hold (ZOH) discretization with stepsize $\Delta$ leads to the recurrence $x_k = \exp(\Delta\tilde{A})x_{k-1} + (\exp(\Delta\tilde{A}) - I)\tilde{A}^{-1}\tilde{B}u_k$. This formula, while **arguably complex** compared to our Eq.(129), relates to it as outlined in the following paragraphs.

**Matrix exponentials make training easier.** The exponential in the ZOH formula arises from the integration of $\dot{x}_{\text{ct}}(t) = \tilde{A}x_{\text{ct}}(t)$, obtaining $x_{\text{ct}}(\Delta k) = \exp(\Delta\tilde{A})x_{\text{ct}}(\Delta(k-1))$. To **enforce stability**, in models inspired by S4 the real part of $A$ is often fed into a positive nonlinearity, as we also do in Sec. 5.3.3. From our results in Sec. 5.3.3 we claim that the **power of exponential parameterization is not necessarily attributable to accurate integration** (which is not present in our system), but is more fundamentally rooted in a **magnitude-phase decoupling** on the recurrence (which **makes training with Adam easier**, see Fig.48), as well as in the overall advantage of learning in diagonalized space (see Tb.4). We also note that stabilizing the recurrence by adding a nonlinearity in parameter generation was beneficial also in our experiments, although this is not prescribed by the theory underlying S4.

**Structured initialization is not necessary.** While Gu et al., 2022a; Gupta et al., 2022b; Smith et al., 2022 also discuss initializations for $A$ deviating from the **HiPPO structure** (see Sec. 5.2), to the best of our knowledge we are the first to show that simple uniform initialization on a slice of the

unit disk, combined with proper normalization, is able to also solve the hardest task in LRA: PathX.[11] We also show (Tb.4) that **uniform initialization** on the disk, which is simply the diagonalized version of Glorot initialization (Thm. 5.3.1), is **sufficient to achieve performance close to more complex deep state-space models** on the remaining LRA tasks. Our results ultimately suggest that HiPPO theory, while fundamental for the development of this field, should not be thought of as the main source of S4's success.

**Discretization changes the initialization spectrum.** For simplicity, we restrict our attention to S4D-Lin, for which $A = \text{diag}(\tilde{a})$ with $\tilde{a}_n = -\frac{1}{2} + i\pi n$, yielding a diagonal transition matrix with elements (i.e. eigenvalues) initialized at $\exp(-\Delta/2 + i\pi\Delta n)$. Under typical choices, such as $\Delta = 1e{-}3$ and $N = 128$, the SSM eigenvalues have magnitude $\exp(-\Delta/2) \approx 0.9995$, and phase $\theta = \pi\Delta n \overset{\sim}{\in} [0, \pi/8]$; i.e. initialization is performed on a ring[12] close to the unit circle in $\mathbb{C}$, with restricted phase connected to the eigenvalues magnitude. As is clear from the results in Sec. 5.3.3 and Sec. 5.3.4, **linking the eigenvalues phase and magnitude is not necessary to achieve good performance**: indeed, as it can be seen in Tb.5, test accuracy on the Long Range Arena (except PathX) can be recovered by using a more natural magnitude-independent initialization on the complete ring. As we discussed in Sec. 5.3.4, changing the initialization phase to a small range around 0 can be motivated by first principles, yet is only needed for extremely long sequences: this modification is already hard-coded in S4, where choosing a small $\Delta$ also shrinks the phase.[13] However, our results clearly show that connecting real and imaginary parts during training through the $\Delta$ parameter is not necessary to achieve good performance, even on PathX.

**Discretization performs normalization.** The most striking visual difference between our model and ZOH-discretized S4 recurrence is in the

11 Among the models in Gu et al., 2022a, only S4D-inv and S4D-LegS (options heavily inspired by the HiPPO theory) perform beyond random guessing on PathX. In S5, the skew-symmetric component of the HiPPO matrix is used for initialization.

12 For all diagonal SSMs, $\Delta$ is actually a vector initialized in the range $[\Delta_{\text{min}}, \Delta_{\text{max}}]$. This interval can be directly mapped through the exponential map to a ring in complex space (see Lemma 5.3.1).

13 This is a useful effect of having a latent continuous-time model: choosing eigenvalues close to the unit circle (i.e. small $\Delta$) changes the oscillation frequencies in the discretized system.

matrix multiplier for $u_k$: $(\exp(\Delta\tilde{A}) - I)\tilde{A}^{-1}\tilde{B}$. After conducting experiments on S4D, we found that simply replacing this multiplier with its first-order expansion in $\Delta$, i.e. $\Delta\tilde{B}$, achieves very similar performance, despite being much simpler and not being a valid second order integration scheme. For input dimension $H = 1$ and unit $B \in \mathbb{R}^{N\times 1}$ (to keep reasoning simple), the corresponding recurrence is

$$x_k = \exp(\Delta\tilde{a}) + \Delta 1_N u_k. \tag{139}$$

Elementwise unrolling of this recurrence without a $\Delta$ factor in front of $u$ yields

$$|x_{k,i}| \leq \sum_{j=0}^{k-1} |\exp(\Delta\tilde{a}_i)|^j u_{k-j,i} \quad \overset{\infty}{\to} \quad O(\Delta^{-1}), \tag{140}$$

We conclude that including a $\Delta$ multiplier on $B$ **implicitly scales the recurrence to normalize the activations**, similar to our $\gamma$ normalization.

**Parameter sharing is not necessary.** As a result of discretization, the $\Delta$ parameter multiplying both $\tilde{A}$ and $\tilde{B}$ couples the recurrence formula with the input projection during training. In our S4 ablations, we found that **decoupling these as two separate parameters**, while keeping the same initialization to normalize the hidden activations (see last paragraph), **does not decrease performance**, suggesting that the ODE discretization viewpoint (which induces parameter sharing) is not necessary to achieve S4 performance.

From this discussion, we conclude that the success of (diagonal) state-space models is attributable to the use of linear recurrences and complex diagonal exponential matrices, combined with the normalization and initialization induced by discretization. On the other hand, other artifacts of discretization such as parameter sharing or the continuous-time interpretation do not necessarily contribute to performance.

# 6

# IMPROVING GENERALIZATION WITH NOISE INJECTION

*Art arises when from a multiplicity of empirical notions a single universal judgment is produced that embraces all things similar to one another. For experience is limited to judging that a certain medicine is suitable for Callia stricken with a certain disease and for many others taken individually, but to judge that a certain medicine is suitable for all those considered a single species is reserved for art.*
*Art is knowledge of the universal.*

– Aristotle.

So far in this thesis, we have focused on algorithmic and architectural insights regarding optimization of complex training loss landscapes. However, the ultimate goal of an optimizer is not solely to find local minimizers that yield vanishing training loss but rather to identify model parameters that result in good **test-set (generalization)** performance (Vapnik, 1999) — beyond memorization (Arpit et al., 2017). While characterizing *good minimizers* mathematically poses a significant challenge, recent literature has presented compelling evidence suggesting that **flat minima** (e.g., with small Hessian eigenvalues) often exhibit superior test performance compared to other minimizers (Hochreiter and Schmidhuber, 1997a; Keskar et al., 2016; Smith and Le, 2017; Jiang et al., 2019; Xie et al., 2020b). Intuitively (perhaps näively), flat minima should generalize better since they are more **robust** to loss landscape perturbations.

A prevailing belief in the literature is that the **data-dependent noise** inherent in stochastic gradient descent (SGD) can drive its trajectory towards flat minima. This mechanism can be further amplified by introducing additional noise in the labels (Hochreiter, 1991). In this chapter, our aim is not to study the link between flatness and generalization (still a subject of **debate among researchers**, see e.g, discussion in Andriushchenko et al., 2023 or the counterexamples in Dinh et al.,

2017) — we take this as an assumption[1]; instead, the question we aim to address is whether it is possible to devise a **data-independent noise injection** scheme (i.e., could also be used in deterministic optimization) capable of guiding the optimization dynamics toward flat minima. Our question is then

*Which injected noise structure can help*
*gradient-based methods find flat minima?*

The structure of this chapter unfolds as follows: after our background section (Section 6.1), in Section 6.2, we explore injection of fractional noise, inspired by the concept of **fractional Brownian motion** (Mandelbrot and Van Ness, 1968). We demonstrate how this approach offers **improved landscape exploration** compared to the conventional Gaussian noise injection on the gradients. Drawing on the insights from the properties of fractional noise, in Section 6.3, we present a novel **anticorrelated noise injection** scheme, Anti-PGD, designed to provably drive the convergence towards flat minima. Leveraging the lessons learned from fractional noise, this scheme hold promise in enhancing the generalization performance of deep learning models. In the same section, we give evidence that simply injecting **uncorrelated** Gaussian noise is generally **not a good strategy**. Lastly, in Section 6.4, we tackle the challenges associated with successful noise injection in the context of deep learning. By addressing these obstacles, we pave the way for more effective noise injection strategies and a deeper understanding of the interplay between flat minima and the success of deep learning models.

Even though the central theme of this thesis revolves around the exploration of adaptive methods, in this last chapter, we will **solely focus on plain SGD**. Our forthcoming research endeavors will concentrate on integrating noise injection with adaptive methods. In particular, we aim to investigate the unique and not fully comprehended generalization characteristics exhibited by optimizers like Adam (refer to Figure 52 for intriguing observations).

1 and find good evidence supporting this claim in our experiments in the next sections.

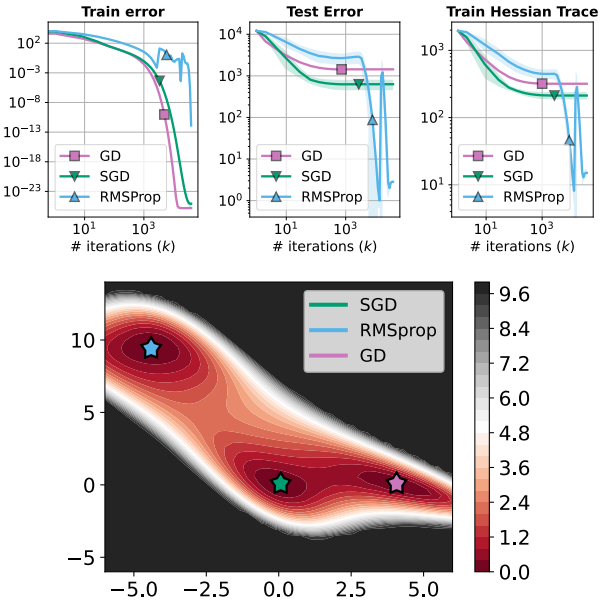


**Figure 52:** *Gradient Descent, Stochastic Gradient Descent, and RMSprop (i.e. Adam without momentum) on a matrix sensing problem. All methods have the same initialization and use the largest stable learning rate with warmup-cosine annealing scheduler. Plotted are the dynamics of the mean and standard deviation of train error, test error, and Hessian trace (5 runs). Plotted is also the landscape slice passing through the minimizes of the 3 methods. SGD behaves better than GD at test time, and converges to a* ***flatter minimum****. Even more interestingly here* ***RMSprop generalizes better*** *that both GD and SGD. However, this effect completely depends on the* ***choice of scheduler****. For a constant stepsize, RMSprop is always suboptimal. The behavior of RMSprop on this problem is* ***not capture by current theory*** *and will be subject of further investigations.*

## 6.1 BACKGROUND ON FLATNESS AND GENERALIZATION

The generalization performance of deep models heavily depends on the choice of optimizer (Wilson et al., 2017; Zhang et al., 2019a; Chen et al., 2020a), and on hyperparameters such as batch size (Shallue et al., 2019; Smith et al., 2020), learning rate schedule (Jastrzebski et al., 2017; Gotmare et al., 2018; Loshchilov and Hutter, 2016), gradient clipping (Zhang

et al., 2019c; Zhang et al., 2020a) and weight decay (Zhang et al., 2018; Loshchilov and Hutter, 2017). To this day, a thorough theoretical characterization of the effect of most of these choices on generalization is missing even for plain SGD. A core fundamental question in this direction concerns the so-called **implicit bias** of vanilla SGD. In brief, it was observed that, while there exist global minimizers which generalize poorly (Zhang et al., 2021a; Liu et al., 2020b), SGD often converges to "good minima" (Soudry et al., 2018; Gunasekar et al., 2017; Neyshabur et al., 2017b; Liu et al., 2021). This feature is often linked to the observed flatness of the loss landscape around the SGD solution (Hochreiter and Schmidhuber, 1997a; Keskar et al., 2016; Chaudhari et al., 2019; Jiang et al., 2019). On the theory side, recent works (Xie et al., 2020b; Simsekli et al., 2019) motivate how the noise structure in SGD can indeed bias the dynamics towards such flat minimizers[2]. Specifically, a few months ago, Wu et al., 2022 showed formally that the Frobenius norm of the loss Hessian at the SGD solution is upper-bounded by a model-independent quantity determined by the batch size and the learning rate, and that if SGD gets close to a sharper region, then it escapes exponentially fast. This phenomenon is linked to the **edge-of-stability** observation in full-batch gradient descent (Cohen et al., 2020): the sharpness (maximum Hessian eigenvalue) of the solution found by full-batch gradient descent is empirically inversely proportional to the used learning rate.

SHARPNESS-AWARE MINIMIZATION. Arguably the most successful "trick" that can be incorporated in SGD to bias minima selection is the one provided by the **Sharpness-aware-minimization** (SAM) algorithm (Foret et al., 2020). Mathematically, SAM proposes to update SGD with gradients from the loss

$$\tilde{f}(x) = \max_{\|\epsilon\|_p \leq \rho} f(x+\epsilon). \tag{141}$$

For $p = 2$, this can be linked to the ascent-descent update $x \leftarrow x - \eta \nabla f(x + \rho \nabla L(x)/\|\nabla f(x)\|)$. While careful tuning of SAM can boost the test performance of SGD by a few percentage points, the connection between the actual update rule and the regularized loss $\tilde{L}$ is only **approximate** (Wen et al., 2022), and suggests further research is needed.

2 Strong correlation between flatness and generalization can be seen e.g. in Figure 52 and in Orvieto et al., 2022a; Orvieto et al., 2023b.

A RESULT LINKING FLATNESS AND GENERALIZATION. While an in-depth look at the theory linking flatness with generalization is out of the scope of this thesis (the reader can check (Dinh et al., 2017) for an in-depth discussion), we like to provide at least one theoretical result on this relation, belonging to the PAC-Bayesian literature. **PAC-Bayes** bounds can be interpreted as bounds on the average loss over a posterior distribution $Q$. These bounds connect to the curvature of the loss through the concept of **expected sharpness** — related to the **trace of the Hessian**.

**Theorem 6.1.1** (Neyshabur et al., 2017a; Tsuzuku et al., 2020)**.** *Let $Q(x|x^*)$ be any distribution over the parameters, centered at the solution $x^*$ found by a gradient-based method. For any non-negative real number $\lambda$, with probability at least $1-\delta$ one has*

$$f_{true}(Q(x|x^*)) \leq f(x^*) + \frac{\lambda}{2M} + \frac{1}{\lambda}\ln\left(\frac{1}{\delta}\right) + \underbrace{f(Q(x|x^*)) - f(x^*)}_{\text{expected sharpness}} + \frac{1}{\lambda}KL[Q(x|x^*)||P(x)],$$

*where $f_{true}$ is the generalization loss; $f(Q) := \mathbb{E}_{x\sim Q}f(x)$ and $f_{true}(Q) := \mathbb{E}_{x\sim Q}f_{true}(x)$; $P$ is a distribution over parameters; and $KL$ denotes the Kullback-Leibler divergence.*

For a proof, see Tsuzuku et al., 2020. In the setting of this theorem, by picking $Q$ to be Gaussian with variance $s^2$, one obtains the following approximation of the expected sharpness

$$f(Q(x|x^*), x^*) - f(x^*) \approx \frac{s^2}{2}Tr(\nabla^2 f(x^*)). \tag{142}$$

This simple connection motivates us to design methods biased to minimizers where the trace of the Hessian is small.

## 6.2 FRACTIONAL PERTURBED GRADIENT DESCENT

*Now, as Mandelbrot points out, Nature has played a joke on the mathematicians. The 19th-century mathematicians may not have been lacking in imagination, but Nature was not. The same pathological structures that the mathematicians invented to break loose from 19th-century naturalism turn out to be inherent in familiar objects all around us.*

– Freeman Dyson

**Question:** The problem of finding flat minimizers is inherently linked to the issue of landscape exploration. Which noise structure enables SGD to explore a greater portion of the landscape?

**Answer (Lucchi et al., 2022):** Fractional Brownian noise has provably better space-filling properties than regular Gaussian noise. When injected on the gradient update, this property translates to enhanced exploration. In the next section, we show how to use these insights to design an efficient algorithm.

Fractional Brownian motion (fBM) is a generalization of Brownian motion where the increments of the stochastic process need not be independent. This family of Gaussian processes was developed for some hydrological modelling by Kolmogorov, 1940. Hurst, 1956 later studied the long term water flow characteristics of the Nile River that was affected by long periods of drought. In such a scenario, the self similarity of the process is related to a long term dependence in the water levels. Since then, fBM has also found applications in financial modeling (Øksendal, 2003). We now give a formal definition of fBM, and discuss its main properties.

**Definition 6.2.1.** *Fractional Brownian motion (fBM) is a centered Gaussian process* $(B^H(t))_{t\in[0,T]}$ *for* $T > 0$ *with covariance function*

$$\mathbb{E}[B^H(t)B^H(s)] = \tfrac{1}{2}(|t|^{2H} + |s|^{2H} - |t-s|^{2H}),$$

*where* $H \in (0,1)$ *is called the* ***Hurst parameter****.*

Hence, fBM is a generalization of Brownian motion, which is recovered for $H = \frac{1}{2}$. If $H > \frac{1}{2}$, then the increments are positively correlated, while

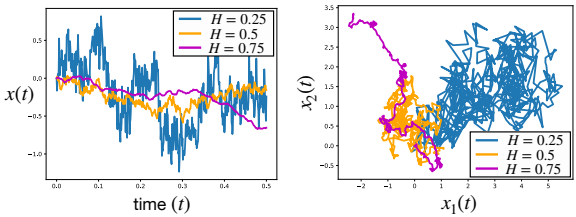


**Figure 53:** *Discretized path of fBM in one dimension (left) and two dimensions (right) for different Hurst parameters. We observe that low values of H yield stochastic processes that explore the space in a denser way that is less "biased" by the direction of the first few perturbations. This can in fact be formalized through the concept of Hausdorff dimension which is given by $d = 2 - H$ for all $H \in (0,1)$.*

they are negatively correlated if $H < \frac{1}{2}$. For $H \neq \frac{1}{2}$, the increments of fBM are not independent, although they are stationary, with variance

$$\mathbb{E}(|B^H(t) - B^H(s)|^2) = |t-s|^{2H}. \tag{143}$$

A crucial property of negatively correlated increments is their ability to fill in the space, as illustrated in Fig. 53.

For injection of standard (multidimensional) Brownian motion $B_t$ (i.e. for Hurst parameter $H = 1/2$), the discretization of the perturbed gradient flow SDE $dX_t = -\nabla f(X)dt + \sigma dB_t$ leads to

$$x_{k+1} = x_k - \eta \nabla f(x_k) + \sigma[B_{\eta(k+1)} - B_{\eta(k)})], \tag{144}$$

with step size $\eta > 0$. This method is called *perturbed gradient descent* (PGD) (Jin et al., 2021). Here, by instead discretizing the fractional SDE $dX_t = -\nabla f(X)dt + \sigma dB_t^H$ for any $H \in (0,1)$, we obtain a generalized version of PGD, which we call **fractional PGD** (fPGD):

$$x_{k+1} = x_k - \eta \nabla f(x_k) + \sigma[B^H_{\eta(k+1)} - B^H_{\eta(k)})], \tag{145}$$

which is equal to PGD if $H = 1/2$. The parameter $H$ determines the correlation of noise in fPGD: anticorrelated noise for $H \in (0, 1/2)$, positively correlated noise for $H \in (1/2, 1)$, and uncorrelated (independent) noise for $H = 1/2$. Importantly – since $B_t^H$ converges to a white-noise process (in distribution) as $H \to 0$; see Lemma 4.1 by Borovkov et al., 2017.

We test the ability of fPGD to escape local minima and explore the landscape. To this end, we consider the Styblinski–Tang function in two dimensions. The result in Figure 54 confirms our intuition.

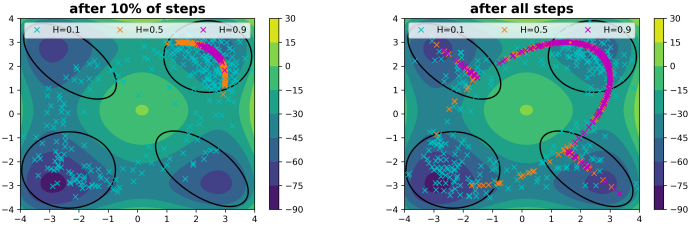


**Figure 54:** *Exploration performance of fPGD on the Styblinski-Tang function* $f(x,y) = \frac{1}{2}\left(x^4 - 16x^2 + 5x + y^4 - 16y^2 + 5y\right)$. ***Right****: empirical distribution of fPGD for different* $H$ *after* $10^4$ *steps.* ***Left****: empirical distribution after all* $10^5$ *steps. We can see that a small* $H$ *deals better with this challenging escape task.*

COMMENT ON THE RESULT. Our intuition is also supported by many mathematical results confirming that fractional Brownian motion with low Hurst parameter $H$ **fills the space more densely**. This behavior comes from the fact that the graph of the fBM has a **Hausdorff dimension** and box-counting dimension given by $d = 2 - H$. The latter implies that $N(1/n) \approx Cn^d$ for large $n$, where $N(\varepsilon)$ denotes the number of boxes with side length $\varepsilon$, which is needed to cover the graph of the fBM. Therefore, the graph of a (1-dimensional) fBM behaves like a $2-$dimensional smooth manifold for small $H$. See Schroeder, 1991. The above claim about the Hausdorff dimension and the box-filling dimension of fBM is proved by Wu and Xiao, 2007a. Further related articles are Ayache and Xiao, 2005 and Wu and Xiao, 2007b.

## 6.3 FINDING FLAT MINIMA WITH ANTI-PGD

> *The validity of all the Inductive Methods depends on the assumption that every event, or the beginning of every phenomenon, must have some cause; some antecedent, upon the existence of which it is invariably and unconditionally consequent.*
>
> – John Stuart Mill

While fGN leads to desirable properties, unfortunately, its simulation in high dimensions is **computationally expensive** (Dieker, 2004). In this section, we address this issue and design a new cheap anticorrelated method capable of promoting flatness and empirically boosting generalization.

**Question:** Fractional Gaussian noise with a small Hurst parameter leads to greater landscape exploration than simple uncorrelated Gaussian noise injection. Can we use this lesson to design a cheap noise injection scheme? What would be the effect on flatness?

**Answer (Orvieto et al., 2022a):** Maximally anticorrelated noise injection ($H \to 0$) can be implemented in a cheap way, and provably drives gradient descent to flat minima enhancing generalization. In the next section, we further modify the algorithm presented here and improve its performance on wide neural networks.

Gradient descent (GD) iteratively optimizes the loss $f(x)$ by computing a sequence of solution approximation $\{x_k\}_{k=0}^{K-1}$ where $x_{k+1} = x_k - \eta \nabla f(x_k)$ with step size (a.k.a. learning rate) $\eta > 0$. Perturbed gradient descent (PGD) simply adds an i.i.d. perturbation to each step, i.e.

$$x_{k+1} = x_k - \eta \nabla f(x_k) + \xi_{k+1}, \tag{146}$$

where $\{\xi_k\}_{k=0}^{K-1}$ is a set of centered i.i.d. random variables with variance $\sigma^2 I$. Similarly, we define *anticorrelated perturbed gradient descent* (**Anti-PGD**) as

$$x_{k+1} = x_k - \eta \nabla f(x_k) + (\xi_{k+1} - \xi_k). \tag{147}$$

In other words, Anti-PGD replaces the i.i.d. perturbations $\{\xi_k\}_{k=0}^{K-1}$ in PGD with their increments $\{\xi_{k+1} - \xi_k\}_{k=0}^{K-1}$. The name Anti-PGD comes from the fact that consecutive **perturbations are anticorrelated**:

$$\frac{\mathbb{E}\left[(\xi_{k+1} - \xi_k)(\xi_k - \xi_{k-1})^\top\right]}{2\sigma^2} \overset{\text{(iid)}}{=} -\frac{Cov\,(\xi_0)}{2\sigma^2} = -\frac{1}{2}I. \tag{148}$$

### 6.3.1 *Regularization in Anti-PGD*

While Anti-PGD is defined as a modification of PGD, it can alternatively be viewed as a regularization (smoothing) of the loss landscape $L$. To see this, note that, after a change of variables $w_k := x_k - \xi_k$, the Anti-PGD step becomes

$$w_{k+1} = w_k - \eta \nabla f(w_k + \xi_k). \tag{149}$$

The corresponding loss $f(\cdot + \xi_k)$ can, in expectation, be regarded as a **smoothed** version of the original $f$. To see in which direction the gradients of this loss (and thus Anti-PGD) are biased, we perform a Taylor expansion of $\partial_i f(\cdot)$ around $w_k$:

$$\begin{aligned} w_{k+1,i} = w_{k,i} - \eta \partial_i f(w_k) - \eta \sum_j \partial^2_{ij} f(w_k) \xi_{k,j} \\ \underbrace{- \frac{\eta}{2} \sum_{j,r} \partial^3_{ijr} f(w_k) \xi_{k,j} \xi_{k,r}}_{= \frac{\eta}{2} \partial_i \sum_{j,r} \partial^2_{jr} f(w_k) \xi_{k,j} \xi_{k,r}} + O(\eta \|\xi_k\|^3), \end{aligned} \tag{150}$$

where the term under the brace is due to Clairaut's theorem (assuming that $f$ has continuous fourth-order partial derivatives). By exploiting that $\xi_k$ has mean zero and covariance $\sigma^2 I$, we can express the conditional expectation of each step as

$$\mathbb{E}\left[w_{k+1} | w_k\right] = w_k - \eta \nabla \tilde{f}(w_k) + O\left(\eta \mathbb{E}[\|\xi_k\|^3]\right), \tag{151}$$

where the *modified loss* $\tilde{L}$ is given by

$$\tilde{f}(x) := f(x) + \frac{\sigma^2}{2} Tr(\nabla^2 f(x)), \tag{152}$$

where $Tr(A)$ denotes the trace of a square matrix $A$. The conditional mean, Eq. (151), highlights the *motivation* for Anti-PGD: When expressed in terms of the variable $w_k$, Anti-PGD in expectation (modulo the impact of the third moment of the noise) takes steps in the direction of a loss which is **regularized** by adding the **trace of the Hessian**. The higher the noise variance $\sigma^2$, the stronger is the influence of the (trace of the) Hessian on Anti-PGD. This is related to how stochastic gradient noise smoothes the loss in standard SGD (Kleinberg et al., 2018), with the difference that, here, we inject artificial noise that *explicitly* regularizes the trace of the Hessian.
In the next theorem, we analyze the case where the noise $\xi_k$ follows a symmetric Bernoulli distribution. We find that, indeed, Anti-PGD (on average) minimizes the regularized loss $\tilde{f}$ – in the sense that the regularized gradient converges.

**Theorem 6.3.1** (Convergence of the regularized gradients)**.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be lower bounded with continuous fourth-order partial derivatives and $\beta$-Lipschitz continuous third-order partial derivatives, for some constant $\beta > 0$. Consider the iterates $\{w_k\}_{n=0}^{K-1}$ computed by Anti-PGD as in Eq. (149), where for each $n$ the noise coordinate $\xi_{k,i}$ follows a symmetric centered Bernoulli distribution with variance $\sigma^2$ (i.e., $\sigma$ and $-\sigma$ have probability $1/2$). If we set $\epsilon > 0$ small enough such that $\eta = \Theta(\epsilon/\sigma^2) < \frac{1}{\beta}$ and let $N = \Omega(\sigma^2\epsilon^{-2})$, then it holds true that*

$$\mathbb{E}\left[\frac{1}{K}\sum_{n=0}^{K-1}\|\nabla\tilde{f}(w_k)\|^2\right] \leq O(\epsilon) + O(\sigma^3). \tag{153}$$

By minimizing the trace of the Hessian, Anti-PGD is expected also to reduce the PAC-Bayes bound from Thm. 6.1.1. In fact, the reasoning behind the bound in Thm. 6.1.1 has motivated researchers to find an explicit link between stochastic gradient noise and the trace of the Hessian at the solution found by SGD. Empirically, these quantities have a high correlation in many settings (Yao et al., 2020; Smith et al., 2021): usually, the lower the trace (i.e., the flatter the minima), the higher is the test accuracy. Similar bounds involving the trace of the Hessian are also discussed by Dziugaite and Roy, 2018; Wang et al., 2018.

COMPARISON WITH LABEL NOISE. Instead of perturbing $x$ as in PGD, label-noise methods perturb labels before computing the gradient. If we

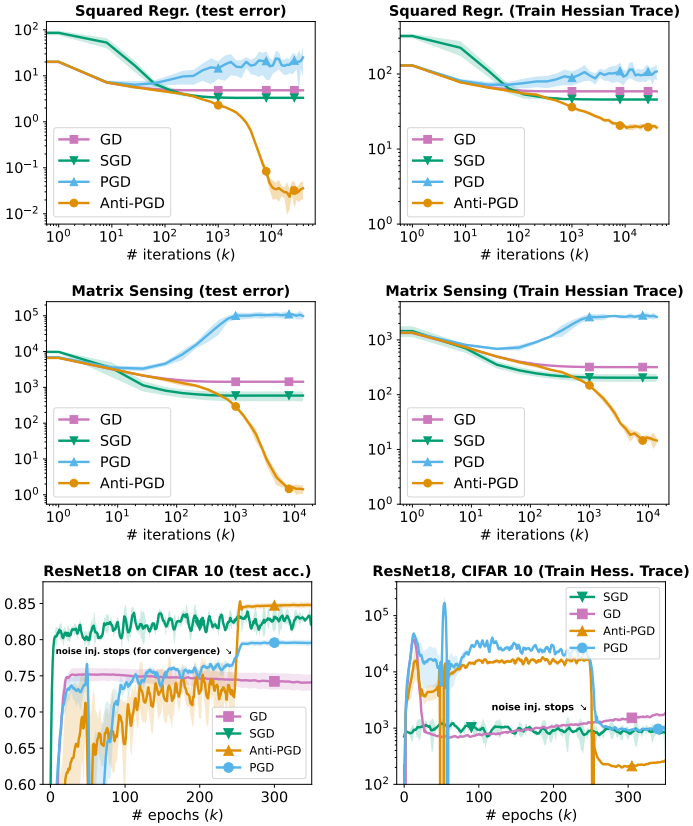


**Figure 55:** *Effect of uncorrelated (PGD) and anticorrelated (Anti-PGD) noise injection on learning with gradient descent. Experiments are conducted on three non-convex machine learning problems with increasing complexity. These experiments are inspired by recent literature on label noise Blanc et al., 2020; HaoChen et al., 2021. Shown is the mean and standard deviation over several runs (5 for the first two problems, 3 for the last). Findings are robust to hyperparameter tuning. In the ResNet18 experiments, the high dimensionality makes it hard to evaluate metrics under noise injection – since we converge to a neighborhood with big (dimension dependent) radius. Hence, we evaluate the accuracy and the Hessian trace after stopping noise injection, to allow exact convergence to the nearest minimizer. Note that, while SGD can temporarily be better than Anti-PGD in test performance for a small number of iterations, Anti-PGD ultimately outperforms SGD in test performance for all experiments.*

denote $\Phi_x(z^{(i)})$ as the output of our model for input $z^{(i)}$, the label-noise update in the full-batch setting with squared loss is $x_{k+1} = x_k - \eta \nabla \bar{f}(x_k)$, with

$$\bar{f}(w) = \frac{1}{2} \sum_{i=1}^{M} \left[ \Phi_x(z^{(i)}) - y^{(i)} + \xi_{k+1} \right]^2, \tag{154}$$

for a set of random perturbations $\{\xi_k\}_{n=0}^{K-1}$. It is instructive to compare the label-noise loss $\bar{f}$ with the Anti-PGD loss $f(\cdot + \xi_k)$ from Eq. (149). The above formula gives the gradient as

$$\nabla \bar{f}(x) = \nabla f(x) + \sum_{i=1}^{M} \nabla \Phi_x(z^{(i)}) \xi_{k+1}. \tag{155}$$

Hence, while **label noise** was observed to yield an improvement in terms of generalization (Blanc et al., 2020; HaoChen et al., 2021; Damian et al., 2021), its effect (in general) is **highly dependent on the model and on the data**. Instead, the noise injection we propose is both data and model independent, as can be seen from the regularization in Eq. (151).

CONNECTION TO LANDSCAPE SMOOTHING. Eq. (152) shows that Anti-PGD amounts to optimizing a regularized loss, which we can also interpret as a smoothing of the original objective function. Smoothing is of course not a new concept in the field of optimization as it is often used to regularize non-differentiable functions in order to compute approximate derivatives (Nesterov and Spokoiny, 2017), or to obtain faster rates of convergence (Lin et al., 2018). In the context of deep learning, noise injection (or even stochastic gradient noise) is often linked to smoothing (Kleinberg et al., 2018; Stich and Harshvardhan, 2021; Bisla et al., 2022). As we saw in Eq. (149), anticorrelated noise injection is equivalent to smoothing after a change of variables – this property was crucial in deriving the trace regularizer. We are not aware of any similar explicit regularization result in the smoothing literature (most work focuses on the resulting landscape properties and convergence guarantees). Even though Anti-PGD is linked to smoothing, it is much more convenient to analyze: $\nabla f(x + \xi)$ follows a data-dependent distribution that is complex to characterize. Instead, in Anti-PGD, the smoothing effect comes from adding — this is a *linear* operation — anticorrelated random variables. This is very convenient and will be leveraged in the proof for the next result, Thm. 6.3.2.

### 6.3.2 *Behavior of Anti-PGD in Widening Valleys*

We have seen above that Anti-PGD acts as a regularizer on the trace of the Hessian. In this section, we will analyze the dynamics of Anti-PGD in more detail on the "widening valley" – the simplest possible loss landscape with a changing trace of the Hessian. We demonstrate with experiments that Anti-PGD successfully finds flat minima in this model, prove this behaviour theoretically.
The *widening valley* is defined as the loss function

$$f(u,v) = \tfrac{1}{2}v^2\|u\|^2, \quad x := (u,v), \tag{156}$$

where $\|\cdot\|$ is the Euclidean norm, $v \in \mathbb{R}$, and $u \in \mathbb{R}^d$ ($x \in \mathbb{R}^{d+1}$); see Fig. 56. The gradient and Hessian of $L$ are given by

$$\nabla f(u,v) = \begin{bmatrix} v^2 \cdot u \\ \|u\|^2 v \end{bmatrix}, \quad \nabla^2 f(u,v) = \begin{bmatrix} v^2 I_d & 2vu \\ 2vu^\top & \|u\| \end{bmatrix}. \tag{157}$$

The trace of the Hessian is thus

$$Tr(\nabla^2 f(u,v)) = dv^2 + \|u\|^2. \tag{158}$$

We consider $L$ as a suitable problem to analyze the dynamics of GD and Anti-PGD as it has a relatively simple structure consisting of a valley of minima with monotonously changing flatness (as measured by the trace of the Hessian): All $(u,v)$ with $v = 0$ are minima, but we also require $\|u\|$ to be minimized as well to get a small trace of the Hessian.
The **widening valley** can also be seen as a simplified local model of the landscape close to a minimizer. Indeed, Draxler et al., 2018 showed that minimizers in neural networks are often connected by a path where the loss is exactly zero: no jumping is required for an optimizer to gradually increase the solution flatness. While these valleys are not straight in general and the flatness might not change monotonously, our straight valley serves as a first simplified model.

EMPIRICAL DEMONSTRATION. When optimizing the widening valley (Eq. (156)), GD will get stuck in any of the global minima $(u, v = 0)$, regardless of their flatness. In particular, if the dimension $d \gg 1$, the path of GD will be biased towards making $v$ small and not optimizing

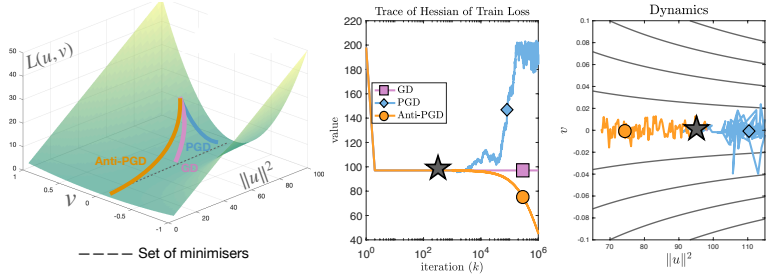


**Figure 56:** ***(Left)*** *Illustration of the* widening valley *loss L, Eq. (156). A valley of minima with loss $f(u,v) = 0$ for all $(u,v)$ with $v = 0$; the smaller $\|u\|$, the flatter the minimum. GD gets stuck where it first touches this valley. PGD diverges to sharp regions (with high $\|u\|$). Anti-PGD converges to a flat minimum (with small $\|u\|$).* ***(Right)*** *Simulation of the considered algorithms on the widening valley. After convergence of GD (black star), we start injecting uncorrelated and anticorrelated noise. We choose $\eta = 0.01$, and $\sigma = 0.005$ – yet the findings generalize to all sets of stable parameters. The observed behavior is supported by Thm. 6.3.2. The plot looks similar for both Gaussian and Bernoulli noise injection.*

$u$ (since the direction along $v$ is the most curved). As a result, the final Hessian trace will be $\|u_0\|^2$. Improving this by injecting noise is challenging: when adding stochastic perturbations, one has to balance perturbing $v$ away from zero – to get a gradient to reduce $\|u\|$ – while preventing $\|u\|$ from growing too much.

We find empirically that Anti-PGD succeeds to do this and **moves to flat parts of the valley**, while PGD does not; see Fig. 56. This means that Anti-PGD converges to flat parts of the valley, while PGD diverges to sharper regions; see Fig. 57.

THEORETICAL ANALYSIS. The following theorem proves what we empirically demonstrated in the last subsection.

**Theorem 6.3.2** (Widening Valley)**.** *Let $f : \mathbb{R}^{d+1} \to \mathbb{R}$ be the* widening valley *loss from Eq. (156). We start optimizing from a point $x_0 = (u_0, 0)$, where $\|x_0\|^2 = D \gg 1$ (e.g. the solution found by gradient descent), around which we consider the domain $\mathcal{D}_\alpha := \{(u,v) \in \mathbb{R}^{d+1} : \|u\|^2 \in (\alpha D, D/\alpha)\}$ for some fixed $\alpha \in (0,1)$. We want to compare the long-term stochastic dynamics of PGD and Anti-PGD, as defined in Eqs. (146) and (147), in terms*

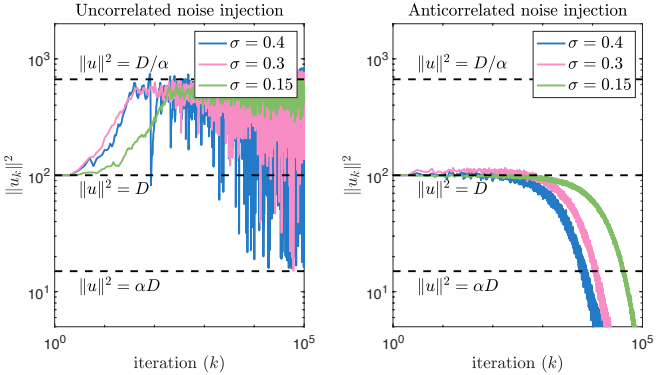


**Figure 57:** *Numerical illustration and verification of Thm. 6.3.2. Performance of PGD (**left**) and Anti-PGD (**right**) on the widening valley in Eq. (156). The setting and the notation is as described in Thm. 6.3.2, and the simulation confirms the result: that is, Anti-PGD effectively decreases $\|u\|^2$ below $\alpha D$, where for this plot we consider $\alpha = 0.25$, $\eta = \alpha/D$ and $d = 100$. Instead, the high problem dimensionality $d \geq 2/\alpha^2 = 32$ increases $\|u\|^2$ for standard PGD, which gets bigger than $D/\alpha$.*

*of where they exit $\mathcal{D}_\alpha$. As a noise model, we assume that the i.i.d. perturbations $\xi_k$ are distributed according to a symmetric centered Bernoulli distribution (i.e., $\sigma$ and $-\sigma$ have probability $1/2$) whose variance $\sigma^2$ is upper bounded by $\sigma^2 \in \left(0, \min\left\{\frac{\alpha^3 D}{2}, \frac{D}{8\alpha}\right\}\right]$. As a step size, we set $\eta = \frac{\alpha}{2D}$ which, for both methods, leads to stable dynamics inside of $\mathcal{D}_\alpha$. We find that (on average) PGD and Anti-PGD exit through different sides of $\mathcal{D}_\alpha$:*

1. ***In high dimensions, PGD diverges away from zero.** If $d \geq \frac{2}{\alpha^2}$, then it holds for any admissible $\sigma^2$ that*

$$\lim_{n\to\infty} \mathbb{E}\left[\|u_k\|^2\right] \geq D/\alpha, \tag{159}$$

   *where $u_k$ are the first $d$ coordinates of $x_k$ computed by PGD as in Eq. (146).*

2. ***Independently of dimensions, Anti-PGD converges to zero.** For any $d \in \mathbb{N}$, if we choose any admissible $\sigma^2$ such that $\sigma^2 \leq \frac{\alpha D}{2d}$, then*

$$\lim_{n\to\infty} \mathbb{E}\left[\|u_k\|^2\right] \leq \alpha D, \tag{160}$$

*where $u_k$ are the first $d$ coordinates of $x_k$, computed by Anti-PGD as in Eq. (147).*

As expected, this theorem implies that, as $n \to \infty$, Anti-PGD reduces the trace of Hessian while PGD increases it.

**Corollary 6.3.1.** *In the same setting as Thm. 6.3.2, let*

$$\eta = \frac{\alpha}{2D}, \quad \sigma^2 \in \left(0, \min\left\{\tfrac{\alpha^3 D}{2}, \tfrac{D}{8\alpha}, \tfrac{\alpha D}{2d}\right\}\right], \quad d \geq \frac{2}{\alpha^2}.$$

*If $\alpha \ll 1$, then*

$$\lim_{k\to\infty} \mathbb{E}[Tr(\nabla^2 f(x_k^{anti}))] \leq 16\alpha D \ll \mathbb{E}[Tr(\nabla^2 f(x_0))]$$
$$\lim_{k\to\infty} \mathbb{E}[Tr(\nabla^2 f(x_k^{un}))] \geq D/\alpha \gg \mathbb{E}[Tr(\nabla^2 f(x_0))],$$

*where $x_k^{un} = (u_k, v_k)$ and $x_k^{anti} = (u_k, v_k)$ are the weights returned by the algorithm Anti-PGD and PGD respectively.*

## 6.4 NOISE INJECTION IN THE WIDTH LIMIT

*No one shall expel us from the Paradise that Cantor has created.*
– David Hilbert.

In the last section, we showed that anticorrelated noise injection can be related to landscape smoothing. This connection, resulting in regularization with the trace of the Hessian, only holds **in expectation**. While this is enough to boost generalization in small models, the **regularizer variance blows up** as the network grows in width. In this section, we address this problem and produce an improved variant of Anti-PGD: **L-GRASP**.

**Question:** Anti-PGD was found to be effective, both in theory and in practice, in promoting flatness during optimization. Can we provide a better characterization of the regularization provided by this method? How does stochasticity affect regularization quality?

**Answer (Orvieto et al., 2023b):** The regularization provided by AntiPGD is related in simple settings to theoretically principled approaches such as Lasso and Nuclear Norm penalties. As the neural network grows in width, regularization is highly affected by noise. In this section, we show how to overcome this problem by slightly modifying the noise injection scheme.

Let us increase the level of precision in our analysis and consider here a loss criterion over $n$ datapoints $L : \mathbb{R}^n \to \mathbb{R}$, as well as a predictor $\Phi : \mathbb{R}^m \to \mathbb{R}^n$ with parameters $x$. We aim to minimize

$$f(x) = L(\Phi(x)) \tag{161}$$

with respect to the learnable parameters $x \in \mathbb{R}^m$. As can be seen in the example below, this formulation is very general, and we prove it to be very powerful for analysis in this section.

**Example 6.4.1** (ReLU networks)**.** *Consider input data $Z \in \mathbb{R}^{n\times d_0}$, with $L(\varphi) = \frac{1}{2n}\|Y^\top - \varphi\|_F^2$ for output data $Y \in \mathbb{R}^{n\times d_M}$ and $\varphi \in \mathbb{R}^{d_M\times n}$. For a ReLU network, $x = (W_1, W_2, \dots, W_M)$ with $W_i \in \mathbb{R}^{d_i\times d_{i-1}}$, we have*

$$\Phi(W_M, \dots, W_1) = W_M(W_{M-1}\dots(W_1 Z^\top)_+\dots)_+. \tag{162}$$

### 6.4.1 *Statistical Properties of Noise Injection*

In Section 6.3, we showed that the **Anti-PGD iteration is equivalent**, after a change of variables, to the following update: $x_{k+1} = x_k - \eta \nabla f(x_k + \sigma \xi_k)$, which we can write in our setting as

$$x_{k+1} = x_k - \eta \nabla L(\Phi(x + \sigma \xi_k)), \tag{GRASP}$$

where for each $k$, $\xi_k$ is sampled independently from a $d$ dimensional standard Gaussian and $\sigma > 0$ controls the noise variance. For reference and to avoid confusion with Perturbed Gradient Descent (PGD, $x_{k+1} = x_k - \eta \nabla f(x) + \sigma \xi_k$, see Xie et al., 2020b), we call this method **GRadient Argument Stochastic Perturbation (GRASP)**.

It is clear that $L(\Phi(x + \sigma \xi_k))$ is a stochastic object. This motivates us to study its statistical properties. Let us define

$$f_\sigma(x) = \mathbb{E}\big[L(\Phi(x + \sigma \xi))\big]. \tag{163}$$

We have the following result, which is proved in full generality in the appendix of Orvieto et al., 2023b.

**Theorem 6.4.1.** *Denote by $D\Phi(x) \in \mathbb{R}^{n \times m}$ the Jacobian (matrix of first-order derivatives) of $\Phi$, and $D^2\Phi(x) \in \mathbb{R}^{n \times m \times m}$ the tensor of second-order derivatives. Assume for simplicity of exposition that $L$ and $\Phi$ are three times differentiable functions with bounded third derivatives. In the appendix of Orvieto et al., 2023b, we discuss the case of functions where this is satisfied only piece-wise (to cover ReLU activations). We have*

$$\begin{aligned} f_\sigma(x) = L(\Phi(x)) &+ \frac{\sigma^2}{2} DL(\Phi(x)) D^2\Phi(x)[I] \\ &+ \frac{\sigma^2}{2} D^2 L(\Phi(x)) \big[D\Phi(x) D\Phi(x)^\top\big] + O(\sigma^3). \end{aligned}$$

*Where, for $\varphi \in \mathbb{R}^n$, $DL(\varphi) \in \mathbb{R}^{1 \times n}$ is the row-vector of first-order partial derivatives, and $D^2 L(\varphi) \in \mathbb{R}^{n \times n}$ the matrix of second-order partial derivatives, with the notation $D^2 L(\varphi)[M] = \sum_{a,b=1}^n D^2 L(\varphi)_{ab} M_{ab} \in \mathbb{R}$. Similarly, the operation $D^2\Phi(x)[M] \in \mathbb{R}^n$ is defined at coordinate $j$ as $D^2\Phi(x)[M]_j = \sum_{a,b=1}^m D^2\Phi(x)_{jab} M_{ab}$.*

*Proof sketch.* The Taylor expansion of $\Phi$ at $w$ can be written:

$$\Phi(x+\sigma\xi) = \Phi(x) + \sigma D\Phi(x)\xi + \tfrac{\sigma^2}{2} D^2\Phi(x)[\xi\xi^\top] + O(\sigma^3\|\xi\|^3),$$

We also need a Taylor expansion of $L$ around $\Phi(x)$:

$$L(\Phi(x)+\Delta) = L(\Phi(x)) + DL(\Phi(x))\Delta + \tfrac{1}{2} D^2 L(\Phi(x))[\Delta\Delta^\top] + O(\|\Delta\|^3). \tag{164}$$

To compute an expansion of $f_\sigma$, as defined in Eq. (163), we compose the two expansions and get, with the choice

$$\Delta = \sigma D\Phi(x)\xi + \frac{\sigma^2}{2} D^2\Phi(x)[\xi\xi^\top] + O(\sigma^3\|\xi\|^3). \tag{165}$$

the following result

$$\begin{aligned} L(\Phi(x+\sigma\xi)) &= L\big(\Phi(x) + \sigma D\Phi(x)\xi + \tfrac{\sigma^2}{2} D^2\Phi(x)[\xi\xi^\top] + O(\sigma^3\|\xi\|^3)\big) \\ &= L(\Phi(x)) + DL(\Phi(x))\big(\sigma D\Phi(x)\xi + \tfrac{\sigma^2}{2} D^2\Phi(x)[\xi\xi^\top] \\ &\quad + O(\sigma^3\|\xi\|^3)\big) + \tfrac{1}{2} D^2 L(\Phi(x))\big[\sigma D\Phi(x)\xi (\sigma D\Phi(x)\xi)^\top \\ &\quad + O(\sigma^3\|\xi\|^3)\big] + O(\sigma^3\|\xi\|^3). \end{aligned} \tag{166}$$

Taking expectations, using $\mathbb{E}[\xi] = 0$ and $\mathbb{E}[\xi\xi^\top] = I$, we conclude. □

### 6.4.2 *Simplified result and theoretical implications under overparametrization.*

In several situations, we obtain an **asymptotically equivalent cost function** $f_\sigma^{(\mathrm{eff})}(x)$, defined as

$$f_\sigma^{(\mathrm{eff})}(x) = f(x) + \frac{\sigma^2}{2} D^2 L(\Phi(x))\big[D\Phi(x) D\Phi(x)^\top\big], \tag{167}$$

in the sense that finding a minimizer of $f_\sigma(x) = \mathbb{E}\big[L(\Phi(x+\sigma\xi))\big]$ is essentially equivalent to finding a minimizer of $f_\sigma^{(\mathrm{eff})}(w)$ as $\sigma \to 0$. In other words, in practical settings the term $\frac{\sigma^2}{2} DL(\Phi(w)) D^2\Phi(w)[I]$ in Theorem 6.4.1 has no impact.

One simple sufficient condition for the term $\frac{\sigma^2}{2} DL(\Phi(w)) D^2\Phi(w)[I]$ to have no impact is that $D^2\Phi(w)[I] = 0$ for all $w \in \mathbb{R}^m$. This is satisfied for

certain models such as neural networks, where the second-order derivatives have no cross-product terms, or equivalently when $Tr(\Phi_j(x)) = 0$ for all $j = 1, \dots, m$; this is *e.g. the case when the ReLU activation is used*. For a more rigorous discussion, please refer to Orvieto et al., 2023b.
The following results shows that the order of approximation between $f$ and $f_\sigma$ is of order $\sigma^2$, while the one between $f_\sigma$ and $f_\sigma^{(\text{eff})}$ is of order $\sigma^3$. The proof can be found in Orvieto et al., 2023b.

**Theorem 6.4.2.** *Assume that (a) $L$ is three-times continuously differentiable with uniformly bounded third derivatives, (b) There exists a finite number of open convex sets $\{\Omega_i, i = 1, \dots, M\}$ with $\bigcup_{i\in I} \overline{\Omega_i} = \mathbb{R}^m$ such that $\Phi$ restricted to each $\Omega_i$ is three-times continuously differentiable with uniformly bounded third derivatives. Then, there exist constants $C$ and $C'$ (independent of $w$) such that*

$$\forall x \in \mathbb{R}^m, \qquad |f_\sigma(x) - f(x)| \leq C(1 + \|x\|^2)\sigma^2,$$

*while*

$$\forall x \in \mathbb{R}^m, \qquad |f_\sigma(x) - f_\sigma^{(eff)}(x)| \leq C'\sigma^3.$$

A more general condition is related to **overparametrization** –when the model is sufficiently rich to lead to the minimizer of $L$– and we make a formal statement below with simplified assumptions.

**Theorem 6.4.3.** *In the setting of Theorem 6.4.2 additionally assume that (a) $L$ is strongly convex with uniformly bounded second and third derivatives, with unique global minimizer $\varphi^*$, (b) there exists a (non-unique) $x_* \in \mathbb{R}^m$ such that $\Phi(x_*) = \varphi_*$ (**overparametrization condition**), and that (c) there exist minimizers $x_*^\sigma$ and $x_*^{\sigma,(eff)}$ of $f_\sigma$ and $f_\sigma^{(eff)}$ that lie in a compact set $\Omega \subset \mathbb{R}^m$. Then, if $x_*^\sigma$ is a minimizer of $f_\sigma$, and $x_*^{\sigma,(eff)}$ is a minimizer of $f_\sigma^{(eff)}$, we have*

$$\|\Phi(x_*^\sigma) - \varphi_*\|_2^2 = O(\sigma^2), \quad \|\Phi(x_*^{\sigma,(eff)}) - \Phi(x_*^\sigma)\|_2^2 = O(\sigma^3).$$

Note that (a) we characterize the prediction function $\Phi$ taken at the various minimizers to test asymptotic equivalence (but it is not possible to characterize a distance between parameters because in overparametrized models, $\Phi$ cannot be injective), and (b) when $\sigma$ tends to zero the minimizer should converge to the interpolator $\Phi(w) = \varphi_*$ with minimal $D^2L(\Phi(w))\left[D\Phi(w)D\Phi(w)^\top\right]$.

### 6.4.3 *Explicit Regularization by Noise Injection*

We here use the result of Theorem 6.4.1 to precisely characterize the minimizers found by gradient descent with noise injection, i.e.

$$x_{k+1} = x_k - \eta \nabla L(\Phi(x + \sigma \xi_k)). \tag{168}$$

on some simple yet insightful neural network models.

DIAGONAL LINEAR NETWORKS. Following Woodworth et al., 2020; Vaskevicius et al., 2019; Pesme et al., 2021, we consider diagonal networks. All of the three mentioned works achieve implicit sparsity regularization of SGD under sufficiently small initialization of the model parameter. Here we show that noise injection in the model also induces an $\ell_1$-regularization in the problem. The main advantage we have over the discussed work is that our approach is applicable to more complex models as we show below.

We consider $x = (w_1, w_2) \in \mathbb{R}^{2d}$, and $\Phi(x) = Z(w_1 \circ w_1 - w_2 \circ w_2)$ for data $Z \in \mathbb{R}^{n \times d}$, $L(\varphi) = \frac{1}{2n}\|y - \varphi\|_2^2$. When the model is overparametrized, that is $Z$ has rank $n$, we can apply Theorem 6.4.3, and we then get an equivalent risk:

$$f_\sigma^{(\text{eff})}(x) = \frac{1}{2n}\|y - Z(w_1 \circ w_1 - w_2 \circ w_2)\|_2^2 \tag{169}$$

$$+ 2\sigma^2 \operatorname{diag}(Z^\top Z/n)^\top (w_1 \circ w_1 + w_2 \circ w_2), \tag{170}$$

which is exactly the **Lasso** once considering $\beta = w_1 \circ w_1 - w_2 \circ w_2$, that is, minimizing

$$\tfrac{1}{2n}\|y - X\beta\|_2^2 \; + 2\sigma^2 \operatorname{diag}(X^\top X/n)^\top |\beta|. \tag{171}$$

LINEAR NETWORKS WITH TWO LAYERS. Following Baldi and Hornik, 1995; Arora et al., 2019; Saxe et al., 2019; Gidel et al., 2019, we consider $x = (W_1, W_2)$, with $W_1 \in \mathbb{R}^{d_1 \times d_0}$ and $W_2 \in \mathbb{R}^{d_2 \times d_1}$, and $\Phi(w) = W_2 W_1 X^\top$ with input data $Z \in \mathbb{R}^{n \times d_0}$. We consider the square loss for

simplicity, that is, $L(\varphi) = \frac{1}{2n}\|Y^\top - \varphi\|_F^2$ for a response $Y \in \mathbb{R}^{n\times d_2}$ and $\varphi \in \mathbb{R}^{d_2\times n}$. Applying Thm. 6.4.3 we get

$$f_\sigma^{(\text{eff})}(W_1, W_2) = \frac{1}{2n}\|Y^\top - W_2W_1X^\top\|_F^2 + \frac{\sigma^2}{2n}\left[d_2\|W_1X^\top\|_F^2 + \|W_2\|_F^2\cdot\|X\|_F^2\right]. \quad (172)$$

Given the matrix $M = W_2W_1 \in \mathbb{R}^{d_2\times d_0}$, we can optimize $\frac{1}{2}\left[d_2\|W_1X^\top\|_F^2 + \|W_2\|_F^2\cdot\|X\|_F^2\right]$ with respect to compatible matrices $W_1$ and $W_2$, leading to the penalty $\sqrt{d_2}\|X\|_F\cdot\|MX^\top\|_*$, where $\|\cdot\|_*$ is the nuclear norm (sum of singular values), which favors low-rank matrices. Thus, minimizing $f_\sigma^{(\text{eff})}$ above is equivalent to minimizing $\frac{1}{2n}\|Y^\top - MX^\top\|_F^2 + \frac{\sigma^2}{n}\sqrt{d_2}\|X\|_F\cdot\|MX^\top\|_*$. We thus obtain a **nuclear norm penalty**.

### 6.4.4 *Successful Noise Injection in Deep Learning*

Let us start with two-layers linear networks. Refreshing the notation, we consider $W_1 \in \mathbb{R}^{d_1\times d_0}$ and $W_2 \in \mathbb{R}^{d_2\times d_1}$, and $\Phi(W_1, W_2) = W_2W_1Z^\top$ with input data $Z \in \mathbb{R}^{n\times d_0}$. We consider $L(\varphi) = \frac{1}{2n}\|Y^\top - \varphi\|_F^2$ for $Y \in \mathbb{R}^{n\times d_2}$, the matrix of labels (we allow multi-dimensional outputs).
We consider the overparametrized limit $d_1 \to +\infty$ using initialization with random [3] weights of order $(W_1)_{ij} \sim \frac{1}{\sqrt{d_1d_0}}$, and $(W_2)_{ij} \sim \frac{1}{\sqrt{d_2d_1}}$. That is, $\|W_1\|_F^2$ and $\|W_2\|_F^2$ not exploding as $d_1$ grows.

EXPLODING VARIANCE OF HIGH-ORDER TERMS. For Gaussian perturbations $E_1$ and $E_2$ we have the explicit expansion:

$$\begin{aligned}\Phi(w+\sigma\xi) &= (W_2+\sigma E_2)(W_1+\sigma E_1)Z^\top \\ &= W_2W_1Z^\top + \sigma(W_2E_1 + E_2W_1)Z^\top + \sigma^2E_2E_1Z^\top. \end{aligned} \quad (173)$$

Taking expectations and using that $E_1, E_2$ have zero mean and are independent with elementwise unit variance, we get

$$\mathbb{E}\left[\|\Phi(w+\sigma\xi)\|_F^2\right] = \|\Phi(w)\|_F^2 + \sigma^2\left[\|W_2\|_F^2\|X\|_F^2 + d_2\|W_1X^\top\|_F^2\right] + \sigma^4d_2d_1\|X\|_F^2. \quad (174)$$

3 Similar results hold for Glorot initialization Glorot and Bengio, 2010.

We can now compute $f_\sigma$ as:

$$f_\sigma(W_1, W_2) = f(W_1, W_2) + \frac{\sigma^2}{2n}\left[\|W_2\|_F^2\|X\|_F^2 + d_2\|XW_1^\top\|_F^2\right] + \frac{\sigma^4}{2n}d_1d_2\|X\|_F^2. \quad (175)$$

We recover exactly Equation (172), that highlights a nuclear norm regularization, **but we have an extra term** $\frac{\sigma^4}{2n}d_2d_1\|X\|_F^2$, which is of superior order in $\sigma$, but **problematic** when $d_1 \to +\infty$.
Note that $\frac{\sigma^2}{2n}\|W_2\|_F^2\|X\|_F^2$ scales as $\frac{\sigma^2}{n}\|X\|_F^2$, while the term $\frac{\sigma^2}{n}d_2\|XW_1^\top\|_F^2$ scales as $d_2\frac{\sigma^2}{n}\|X\|_F^2$, with thus no explosion in $d_1$. However, the term $\frac{\sigma^4}{2n}d_2d_1\|X\|_F^2$ explodes when $d_1$ goes to infinity.

LAYER-WISE PERTURBATIONS SOLVE THE PROBLEM. A simple solution to the problem outlined in the last paragraph – first suggested by Prof. Francid Bach – is to inject noise only in specific layers before taking the gradient, randomizing the layer choice over iterations and adjusting $\sigma$ properly. Indeed, in the context of the previous paragraph, the expansion of the network map becomes

$$\begin{aligned}\Phi(w + \sigma\xi) &= (W_2 + \sigma\tilde{E}_2)(W_1 + \sigma\tilde{E}_1)Z^\top \\ &= W_2W_1Z^\top + \sigma(W_2\tilde{E}_1) + \tilde{E}_2W_1)Z^\top + \sigma^2\tilde{E}_2\tilde{E}_1Z^\top, \quad (176)\end{aligned}$$

with $\tilde{E}_1 = c \cdot E_1$, $\tilde{E}_2 = (1-c) \cdot E_2$, where $c$ has Bernoulli distribution (1 with probability $1/2$ and 0 otherwise) and again $E_1, E_2$ have zero mean and are independent with elementwise unit variance. Therefore, the problematic term

$$\sigma^2\tilde{E}_2\tilde{E}_1Z^\top = 0 \quad (177)$$

with probability one.
Note however that to obtain the **same regularization effect** as Equation 172, we need to **adapt** $\sigma$. Indeed, from Eq. 176 we get

$$\begin{aligned}&\mathbb{E}\left[\|\Phi(w + \sigma\xi)\|_F^2\right] \\ &= \frac{1}{2}\left(\|\Phi(w)\|_F^2 + \sigma^2\|W_2\|_F^2\|X\|_F^2\right) + \frac{1}{2}\left(\|\Phi(w)\|_F^2 + \sigma^2d_2\|W_1X^\top\|_F^2\right) \\ &= \|\Phi(w)\|_F^2 + \frac{\sigma^2}{2}\left[\|W_2\|_F^2\|X\|_F^2 + d_2\|W_1X^\top\|_F^2\right]. \quad (178)\end{aligned}$$

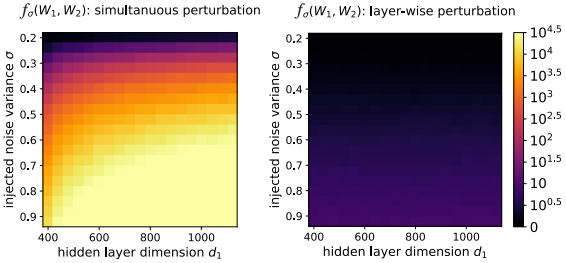


**Figure 58:** *Numerical illustration of exploding higher-order terms (see Eq. (175)) on a linear network with 1 hidden layer of dimension $d_1$. If noise with standard deviation $\sigma$ is added to all weights **(left)**, then the regularized loss $f_\sigma$ explodes as $d_1 \to \infty$ due to the variance term $\sigma^4 d_2 d_1 \|X\|_F^4$. Instead, perturbing $W_1$ and $W_2$ in alternation **(right)** with standard deviation $\sqrt{2}\sigma$ (see Eq. (178)) provides mathematically the same expected regularization but avoids the variance term and therefore provides a much more reliable regularization, as clear also from the experimental section. Runs are repeated 100 times, shown is the average.*

Therefore, to obtain the same regularization effect of Equation 172 we need the rescaling $\sigma \to \sqrt{2}\sigma$. We test the effect of this rescaling empirically in the next subsection and in Figure 58.

We call the noise-injection method resulting from this idea **Layerwise Gradient Argument Stochastic Perturbation (L-GRASP)**. Before testing its performance in the next subsection, we summarize the algorithms discussed in this chapter in Table 6.

| **Algorithm** | **Rule** | **Regularization** |
|---|---|---|
| GD | $x_{k+1} = x_k - \eta \nabla f(x_k)$ | e.g. Soudry et al., 2018 |
| PGD | $x_{k+1} = x_k - \eta \nabla f(x_k) + \sigma \xi_k$ | e.g. Xie et al., 2020b |
| Anti-PGD | $x_{k+1} = x_k - \eta \nabla f(x_k) + \sigma(\xi_k - \xi_{k-1})$ | Sec. 6.3 |
| GRASP | $x_{k+1} = x_k - \eta \nabla f(x_k + \sigma \xi_k)$ | Sec. 6.4 |
| L-GRASP | $x_{k+1} = x_k - \eta \nabla f(x_k + \sigma \xi_{k,p_k})$<br>$p_k$ randomly picked layer at iteration $k$ | Sec. 6.4 |

**Table 6:** *Summary of Algorithms analyzed and compared in this chapter.*

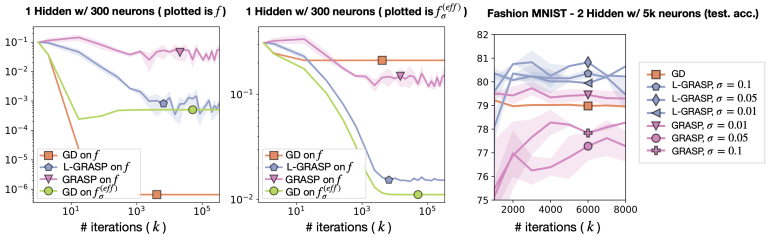


**Figure 59:** ***(Left and central panel)*** *MLP with 1 hidden layer (300 neurons) and linear activations on synthetic data (see the Appendix in Orvieto et al., 2023b for a full description). Only L-GRASP takes us close to a minimizer for the regularized loss $f_\sigma^{(eff)}$ (see Eq. (172)). For all methods, the learning rate is set to 0.1.* ***(Right)*** *Test accuracy for an MLP on FMNIST (2 Hidden Layers with 5000 neurons each). Comparison of perturbation effects on GRASP and L-GRASP: $\sigma$ is properly scaled in L-GRASP – i.e. the reported $\sigma$ refers to the regularization effect.*

### 6.4.5 *Empirical Performance of GRASP and L-GRASP*

The goal of this section is to provide experimental evidence to back-up the results of this section (Section 6.4). In particular, we compare gradient descent (GD) with the perturbed variants GRASP and L-GRASP (see Tb. 6). As seen in Section 6.4.4, GRASP and L-GRASP (with adapted $\sigma$) minimize a similar regularized loss, in expectation. However, as the degree of overparametrization (e.g., number of parameters) increases, the theory suggests that L-GRASP is preferable.

MINIMIZATION OF THE REGULARIZED LOSS. For a start, we consider a one hidden layer network with linear activations and 300 hidden neurons on a randomly generated sparse[4] synthetic regression data set with inputs in $\mathbb{R}^{10}$ and outputs in $\mathbb{R}$. In Figure 59 (left and central panels), we show the dynamics of the iterates of GD, GRASP, and L-GRASP on the original loss $f$ and regularized loss $f_\sigma^{(\text{eff})}$ derived in this section (Eq.172). As the theory predicts, the correct regularization in the strongly overparametrized setting is only achievable with L-GRAPS. As discussed above, to keep the same explicit regularization, the variance of

4 We consider 40 data points sampled from a Gaussian in 10 dimensions. The solution is sparse and is planted as the result of a linear prediction from the first two dimensions.

the noise in the layer-wise approach is doubled compared to the vanilla approach. This depth scaling is adopted for all further experiments.

EFFECT OF TUNING. The findings of the last paragraph are reported for one specific value of $\sigma$. We test the influence of $\sigma$ on a slightly more complex model: an MLP with 2 hidden layers (5000 neurons each) and ReLU activations on Fashion MNIST (Xiao et al., 2017) (classification). Given that the data is relatively easy to fit, we train on a subset of 1024 data points — to induce **heavy overparametrization**. We run full-batch gradient descent with learning rate 0.005, and plot the test accuracy evolution (computed using $10K$ data points) for different values of $\sigma$. Figure 59 (Right Panel) shows that, for this wide model, L-GRASP induces an effective regularization that is able to increase the test accuracy. This is in contrast GRASP. A similar result also holds true for CIFAR10 (Krizhevsky, Hinton, et al., 2009) on ResNet18 (see Figure 61).

DEEP MLPS. Next, we test how the findings carry over to **deeper networks**. In the same data setting as the last paragraph, we now consider a ReLU MLP with 5 hidden layers and either 1000 (narrow) or 5000 (wide) hidden neurons. In Figure 60 we test our methods ($\sigma = 0.05$) against full-batch GD, with a step size of 0.005 in the narrow setting and 0.001 in the wide setting. As can be seen, again L-GRASP yields the best accuracy result — specifically in the wide setting. This is also reflected in the size of the regularizer (trace of Hessian), which is minimized by the best-performing method in terms of test error.

DEEP RESIDUAL NETWORKS. To conclude, we test noise injection (layer-wise or in all layers simultaneously) on a ResNet18 (around 11M parameters) (He et al., 2016) with batch normalization. We use for this the basiline SGD configuration in the popular repository `https://github.com/kuangliu/pytorch-cifar`. On this baseline, which reaches around 94.4% test accuracy, we simply add noise[5] injection at every step. In Figure 61 we tested different noise injection standard deviations $\sigma$ and plotted the mean and standard error of the mean (3 runs) for the final test accuracy and Hessian trace after 150 epochs. We stopped noise injection at epoch

5 Here by "layers" we mean each network parameter group. That is, noise is also injected in the batch-norm parameters.

135 to allow the networks to converge and use cosine annealing, batch-size 128 and learning rate of 0.01 in all methods. The results clearly show that injecting noise in all parameters (GRASP) is able to only regularize the objective and improve test accuracy for very small $\sigma$, but for $\sigma > 0.1$ it hurts performance. Instead, injecting layer-wise noise (L-GRASP) provides a much more consistent regularization and is able to **improve test performance** by $+0.3\%$, which is a sizable margin given the strong SGD baseline. The **poor performance of standard noise injection** is predicted by the theory in Section 6.4.4, which explains the sharp increase of the Hessian trace (for SGD + noise) observed in the second panel at $\sigma = 0.1$.

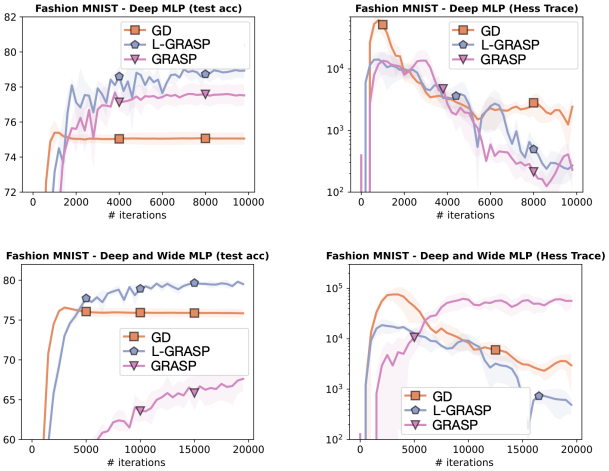


**Figure 60:** *MLP with 5 hidden Layers and ReLU activations on a subset of Fashion MNIST. The hidden layers each have 1000 neurons (narrow, **left** panels) or 5000 neurons (wide, **right** panels). The results show that L-GRASP results in improved regularization (plotted is also the trace of the Hessian, computed with PyHessian (Yao et al., 2020)) and test accuracy.*

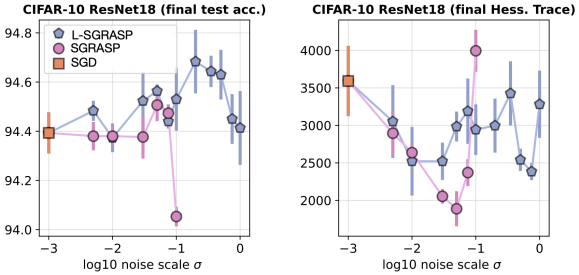


**Figure 61:** *Final Test accuracy **(left)** and Hessian trace **(right)** for SGD on a CIFAR10 ResNet18 with batch normalization. Both plots show mean and standard error of the mean (3 runs). SGD is plotted at $\sigma = 1e-3$ instead of $\sigma = 0$ for better visualization. Layer-wise noise injection is able to boost performance by explicit regularization with a high $\sigma$. Instead, for such high $\sigma$, injecting noise simultaneously in all parameters results in instabilities. We remind that the total noise injection variance is normalized in the two methods.*

# 7 CONCLUSION

> *Then your charitable speeches may find vent; then you may remember and pity the toil and the struggle and the failure; then you may give due honour to the work achieved; then you may find extenuation for errors, and may consent to bury them.*
>
> – George Eliot (Mary Anne Evans).

In this thesis, I summarized a subset of the works produced with the help of many collaborators during my Ph.D. at ETH Zurich. Along this journey, I had the privilege of engaging with exceptional individuals — it was a wonderful and exciting experience.

While the topics of my investigations were never static due to my curiosity and to the rapidly changing literature landscape, my *obsession* always has been understanding the mysterious performance of the state-of-the-art adaptive optimizers for deep learning — bridging the gap between theory and practice. While Adam (Kingma and Ba, 2014) might not be the most theoretically grounded or mathematically beautiful method, it revolutionized deep learning, and I humbly believe its analysis deserves more attention in the optimization community. Despite its complex nature and lack of convergence, we can learn a lot from Adam about successful optimization of nonconvex functions and about the intricate structure of deep learning problems.

To this day, a theoretically principled method to optimize transformers remains to be discovered. Despite our dedicated efforts in Chapter 3, it is to be recognized that Adam remains the most straightforward and widely used choice among practitioners. As such, my primary goal throughout this volume has been to ignite the reader's enthusiasm for its intriguing dynamics. I hope to have conveyed in Chapter 3 that Adam does indeed solve some concrete challenges faced when optimizing deep models. Nonetheless, I also demonstrated that improvements upon adaptive optimizers are possible: in Chapter 4, I gave details of a new method with unprecedented convergence guarantees and solid performance. In

Chapter 5, I described how to design new neural networks in symbiosis with optimizers (i.e. changing parametrization), yielding state-of-the-art performance on long-range reasoning tasks. In Chapter 6, I reported our findings on improving test performance by incorporation of noise injection.

Above all, it is important to remind ourselves that we live in exciting times, both for AI and optimization. This year we have seen the rise of powerful language models (OpenAI, 2023; Touvron et al., 2023) and increasing interest in the derivation of scaling laws (Hoffmann et al., 2022; Bachmann et al., 2023). While all the works cited above use the AdamW optimizer (Loshchilov and Hutter, 2017), we also witnessed this year a rising effort in designing new — better — optimizers such as Lion (Chen et al., 2023) and Sophia (Liu et al., 2023). Moreover an ICML Outstanding Paper award was given in 2023 to a new adaptive method: D-adaptation (Defazio and Mishchenko, 2023).

I will be lucky to continue to work on these exciting topics for the next few years in Tübingen, hoping to directly contribute to making the design and training of neural networks more accessible to scientists and engineers.

# A

# MATHEMATICAL PREREQUISITES

*It is remarkable that a science which began with the consideration of games of chance should have become the most important object of human knowledge. The most important questions of life are indeed, for the most part, really only problems of probability.*

– Pierre Simon Laplace.

We review here the theory of differential equations and convex/smooth optimization. We assume the reader to be familiar with basic analysis and measure-theoretic probability[1]. We start with some results in the theory of Ordinary Differential Equations (ODEs), which we will use heavily in chapters 2 and 6. We give an introduction to stochastic differential equations (SDEs) and, in particular, we present Itô's lemma, which is the workhorse in chapter 2. We conclude this chapter with a review of some basic definitions in the field of smooth optimization (Nesterov, 2018), which we are going to use throughout the thesis.

## A.1 ORDINARY DIFFERENTIAL EQUATIONS

Most of the definitions and results in this section are reported from Monti, 2010, Coleman, 2012, and Königsberger, 2013.

### A.1.1 *General Theory*

In this chapter, we indicate as $\mathcal{C}^r(\mathcal{X}; \mathbb{R}^d)$ the family of functions taking values in $\mathbb{R}^d$ which are $r$ times continuously differentiable in $x \in \mathcal{X}$, with $\mathcal{X} \subseteq \mathbb{R}^d$ an open set.

1 As introductory textbooks on these topics, we recommend Coleman, 2012 and Durrett, 2010.

Consider the Cauchy problem (CP) in the joint time-space set $\Omega \subseteq \mathbb{R}^{d+1}$, where $x \in \mathbb{R}^d$ and $t \in \mathbb{R}$:

$$\begin{cases} \dot{x} = g(t,x) \\ x(t_0) = x_0, \end{cases} \qquad \text{(CP)}$$

where $g(t,x) \in \mathcal{C}(\Omega, \mathbb{R}^d)$. We define now the concept of solution to the CP.

**Definition A.1.1.** *(Solution to the CP) A function $x \in \mathcal{C}^1(I, \mathbb{R}^d)$ is called a solution to the CP if*

1. *$I \in \mathbb{R}$ is an interval s.t $x_0 \in I$;*
2. *$(t, x(t)) \in \Omega$ for all $t \in I$;*
3. *$\dot{x} = g(t,x)$ for all $t \in I$;*
4. *$x(t_0) = x_0$.*

As a consequence of the fundamental theorem of calculus, any solution of the CP in $I$ can be written as

$$x(t) = x_0 + \int_{t_0}^{t} g(\tau, x(\tau)) d\tau.$$

To guarantee the existence of a solution, we are going to assume some regularity of $g$.

**Definition A.1.2** (Local Lipschitz condition)**.** *Let $\Omega \subseteq \mathbb{R}^{d+1}$ be an open set. We say that a function $g \in \mathcal{C}(\Omega, \mathbb{R}^d)$ is locally Lipschitz with respect to $x$ if for any compact set $K \subset \Omega$ there exists a constant $L_K > 0$ such that*

$$\forall (t,x_1), (t,x_2) \in K, \qquad ||g(t,x_1) - g(t,x_2)|| \leq L_K ||x_1 - x_2||.$$

If $g$ has some additional regularity, the local Lipschitzianity is easy to verify.

**Proposition A.1.1.** *Let $g \in \mathcal{C}(\Omega, \mathbb{R}^d)$ be continuously differentiable with respect to $x$, uniformly in $t$. $g$ has the local Lipschitz property in $x$.*

The following is a direct consequence of Banach fixed point theorem.

**Theorem A.1.1** (Picard–Lindelöf). *Let $\Omega \subseteq \mathbb{R}^{d+1}$ be an open set, $(t_0, x) \in \Omega$ and let $g \in \mathcal{C}(\Omega, \mathbb{R}^d)$ satisfy the local Lipschitz property in $x$. Then, there exists $\delta > 0$ such that the solution to the CP exists unique in $I = (t_0 - \delta, t_0 + \delta)$, and $x \in \mathcal{C}(I; \mathbb{R}^d)$.*

**Example A.1.1.** *(Counterexample for Picard–Lindelöf) Take the Cauchy problem*

$$\dot{x} = x^{1/3},\ x(a) = 0.$$

*$x^{1/3}$ is continuous but not locally Lipschitz at the starting condition. The solutions[2] are*

$$x(t) = \pm \left(\frac{2}{3}(t-a)\right)^{3/2}, \quad a > 0.$$

Now, by "gluing" together the intervals $I$ from the Picard–Lindelöf theorem, one has the following result.

**Theorem A.1.2** (Existence in the large). *Let $I = (a_0, b_0)$, with $-\infty \leq a_0 < b_0 \leq \infty$. Let $g \in \mathcal{C}(\Omega, \mathbb{R}^d)$, with $\Omega = I \times \mathbb{R}^d$, be locally Lipschitz with respect to. $x$ and assume additionally that for any compact $K \subset I$ there exists a constant $C$ such that, for all $x \in \mathbb{R}^d$ and $t \in K$*

$$||g(t,x)|| \leq C(1 + ||x||).$$

*Then, the CP with $(t_0, x_0) \in \Omega$ has a unique global solution in $I$.*

Notice that we have here a new condition to verify, which goes under the name of linear growth.

**Example A.1.2** (Counterexample for existence in the large). *Take the Cauchy problem*

$$\dot{x} = x^2,\ x(0) = 1.$$

*$x^2$ is locally Lipschitz but clearly not linearly growing. The unique solution is $x(t) = 1/(1-t)$ and explodes in finite time[3] at the boundary of the domain where linear growth holds.*

We conclude this section stating a very useful property.

**Theorem A.1.3.** *If $g(t,x) \in \mathcal{C}^k(\Omega, \mathbb{R}^d)$, then any solution $x(t)$ of the CP is at least $\mathcal{C}^{k+1}(\mathbb{R}, \mathbb{R}^d)$.*

2 Source : Wikipedia, Picard–Lindelöf theorem. Visited June 4th, 2023

3 one can show that in these situations, the solution always has to explode in finite time at the boundary. This result is known as the continuation criterion.

### A.1.2 *Flow of a Vector Field*

Let us now consider functions $g : \mathbb{R}^d \to \mathbb{R}^d$ that do not have an explicit dependency on time. The CP in this case is:

$$\begin{cases} \dot{x} = g(x) \\ x(t_0) = x_0 \end{cases} .$$

In this case, we say the CP is autonomous.

**Definition A.1.3** (Global Lipschitz condition)**.** *We say that a vector field $g \in \mathcal{C}(\mathbb{R}^d, \mathbb{R}^d)$ is globally Lipschitz if there exists a constant $L > 0$ such that*

$$\forall x_1, x_2 \in \mathbb{R}^d, \quad ||g(x_1) - g(x_2)|| \leq L||x_1 - x_2||.$$

It is clear that any globally Lischitz vector field is also locally Lipschitz. In addition, if the Lipschitzianity is global, linear growth is automatically satisfied in $\mathbb{R}$.

**Lemma A.1.1** (Linear Growth)**.** *Let $g \in \mathcal{C}(\mathbb{R}^d, \mathbb{R}^d)$ be globally Lipschitz. Then $g$ grows at most linearly, i.e there exists $C > 0$ such that for every $x \in \mathbb{R}^d$*

$$||g(x)|| \leq C(1 + ||x||).$$

*Proof.* We use first global Lipschitzianity and then the reverse triangle inequality,

$$\begin{aligned} L||x|| &\geq ||g(x) - g(0)|| \\ &\geq |||g(x)|| - ||g(0)||| \\ &\geq ||g(x)|| - ||g(0)|| \end{aligned}$$

hence, $||g(x)|| \leq \max\{g(0), L\}(1 + ||x||)$. □

Using this Lemma and Theorem A.1.2, we immediately get the following result.

**Corollary A.1.1** (Global existence and uniqueness). *Let $g \in \mathcal{C}(\mathbb{R}^d, \mathbb{R}^d)$ be Lipschitz continuous. Then, the autonomous CP has a unique global solution in $\mathbb{R}$ for any initial condition.*

This corollary motivates the definition of flow.

**Definition A.1.4** (Flow). *The flow of a globally Lipschitz vector field $g(x) \in \mathcal{C}(\mathbb{R}^d; \mathbb{R}^d)$ is the mapping $\Phi : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$ defined as $\Phi(t, x_0) = x(t)$, where $x(t) \in \mathcal{C}^1(\mathbb{R}; \mathbb{R}^d)$ is the solution to the autonomous CP. Moreover, for any $t \in \mathbb{R}$, we define the mapping $\Phi_t : \mathbb{R}^d \to \mathbb{R}^d$ by $\Phi_t(x_0) := \Phi(t, x_0)$.*

One can also write

$$\Phi(t, x_0) = x(t) = x_0 + \int_0^t g(x(\tau))d\tau.$$

For some additional insights on the topology of the flow of autonomous differential equations, one can check Asimov, 1993. The following properties follow immediately from Corollary A.1.1 and A.1.3.

**Proposition A.1.2.** *Let $\Phi$ be the flow of a globally Lipschitz vector field $g(x) \in \mathcal{C}^1(\mathbb{R}^d; \mathbb{R}^d)$*

1. *$\Phi \in \mathcal{C}^1(\mathbb{R}^{d+1}; \mathbb{R}^d)$. In particular $\Phi_t \in \mathcal{C}^1(\mathbb{R}^d; \mathbb{R}^d)$, for all $t$;*
2. *$\Phi_0(x_0) = \Phi(0, x_0) = x_0$. Hence $\Phi_0$ is the identity;*
3. *The flow has the group property $\Phi_{t+s} = \Phi_t \circ \Phi_s$ for all $t, s \in \mathbb{R}$. In particular $\Phi_t$ is invertible and has inverse $\Phi_{-t}$.*

**Corollary A.1.2.** *Fix a time step $\Delta t > 0$, $\Phi_{\Delta t}$ is a diffeomorphism of $\mathbb{R}^d$.*

*Proof.* Just notice that because of the previous proposition, $\Phi_{\Delta t}$ and its inverse $\Phi_{-\Delta t}$ are of class $\mathcal{C}^1(\mathbb{R}^d; \mathbb{R}^d)$. □

To conclude, we denote with $\mathcal{J}_x[f(x)] \in \mathbb{R}^{m \times n}$ the Jacobian matrix of a function $f : \mathbb{R}^m \to \mathbb{R}^n$ evaluated at $x \in \mathbb{R}^m$. We have the following theorem.

**Theorem A.1.4** (Properties of the flow). *Let $\Phi$ be the flow of a globally Lipschitz vector field $g(x) \in \mathcal{C}^1(\mathbb{R}^d; \mathbb{R}^d)$. Fix a time step $\Delta t > 0$, $\Phi_{\Delta t}$ is a diffeomorphism of $\mathbb{R}^d$ and maps sets of measure 0 to sets of measure 0. Moreover, $\mathcal{J}_x[\Phi_{\Delta t}(x)] = \varphi(\Delta t)$, where $\varphi$ is the solution of the variational equations (VE)*

$$\frac{d\varphi(t)}{dt} = \mathcal{J}_x\left[g(\Phi_t(x))\right]\varphi(t), \quad \varphi(0) = I. \quad \text{(VE)}$$

## A.2 STOCHASTIC CALCULUS

Most of basic definitions and results here can be found in Chapters 1, 2 and 5 in Mao, 2007, and in Oksendal, 2003.

### A.2.1 *Probability Spaces*

We start with the definition of measurable space.

**Definition A.2.1** (Measurable space). *Let $\Omega$ be any set. A $\sigma$-algebra on $\Omega$ is a family $\mathcal{F}$ of subsets of $\Omega$ such that*

1. $\emptyset \in \mathcal{F}$;
2. $A \in \mathcal{F} \Rightarrow \Omega / A \in \mathcal{F}$;
3. $\{A_i\}_{i>1} \subseteq \mathcal{F} \Rightarrow \cup_{i=1}^{\infty} A_i \in \mathcal{F}$.

*$(\Omega, \mathcal{F})$ is called a measurable space.*

We say $\hat{\mathcal{F}}$ is a *sub-$\sigma$-algebra* of $\mathcal{F}$ if $(\Omega, \hat{\mathcal{F}})$ is a measurable space and $\hat{\mathcal{F}} \subset \mathcal{F}$. We can specify a probability for each measurable event (i.e. each element of $\mathcal{F}$).

**Definition A.2.2** (Probability space). *A probability measure $\mathbb{P}$ on the measurable space $(\Omega, \mathcal{F})$ is a function $\mathcal{F} \to [0,1]$ such that*

1. $\mathbb{P}(\Omega) = 1$;
2. *For any $\{A_i\}_{i>1} \subseteq \mathcal{F}$ with $A_i \cap A_j = \emptyset$ if $i \neq j$, $\mathbb{P}\left(\cup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \mathbb{P}(A_i)$.*

*$(\Omega, \mathcal{F}, \mathbb{P})$ is called a probability space.*

We now complete this space.

**Definition A.2.3.** *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. The $\sigma-$algebra*

$$\mathcal{F}^* := \{A \subseteq \Omega \ : \ \exists B, C \in \mathcal{F} \text{ such that } B \subseteq A \subseteq C, \ \mathbb{P}(B) = \mathbb{P}(C)\}$$

*is called the completion of $\mathcal{F}$.*

We can easily induce the probability measure from $(\Omega, \mathcal{F}, \mathbb{P})$ to $(\Omega, \mathcal{F}^*, \mathbb{P})$ by setting $\mathbb{P}(A) = \mathbb{P}(B) = \mathbb{P}(C)$ in the setting of the definition above. The probability space $(\Omega, \mathcal{F}^*, \mathbb{P})$ is called *complete*. From now on we always assume $(\Omega, \mathcal{F}, \mathbb{P})$ to be complete.

**Definition A.2.4** (Random variable)**.** *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. A function $X : \Omega \to \mathbb{R}^{d\times m}$ is called a ($\mathcal{F}$-measurable) random variable if for each component $X_{ij} : \Omega \to \mathbb{R}$*

$$\{\omega : X_{ij}(\omega) \leq a\} \in \mathcal{F} \quad \textit{for all } a \in \mathbb{R}.$$

One can deepen their understanding of random variables by elaborating on the following quote by Gian-Carlo Rota:

*"A random variable is neither random nor variable".*

Let $\mathcal{C}$ be a family of subsets of $\Omega$ and $\sigma(\mathcal{C})$ be the smallest $\sigma$-algebra on $\Omega$ which contains $\mathcal{C}$. Let $\Omega = \mathbb{R}^{d\times m}$ and $\mathcal{C}$ be the famility of all open sets of $\mathbb{R}^{d\times m}$, the Borel $\sigma$-algebra on $\mathbb{R}^{d\times m}$ is $\mathcal{B}^{d\times m} := \sigma(\mathcal{C})$. If a random variable is defined over $(\mathbb{R}^{d\times m}, \mathcal{B}^{d\times m})$ we say this is a *Borel-measurable function*.
Next, notice that a random variable $X : \Omega \to \mathbb{R}^{d\times m}$ induces a probability measure $\nu_X$ on $(\mathbb{R}^{d\times m}, \mathcal{B}^{d\times m})$. $\nu_X$ is called the *distribution* of $X$. What's more we can do this the other way around: a random variable $X$ induces a $\sigma$-algebra $\sigma(X)$, which is assumed to be the smallest possible, on $\Omega$.
We now proceed with a definition.

**Definition A.2.5** (Absolute continuity)**.** *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space and $X : \Omega \to \mathbb{R}^{d\times m}$ be a random variable with distribution $\nu_X$. Let $\mu^{d\times m}$ denote the Lebesgue measure on $\mathbb{R}^{d\times m}$. $\nu_X$ is called absolutely continuous with respect to $\mu^{d\times m}$ ($\nu_X \ll \mu^{d\times m}$) if for any $A \in \mathcal{B}^{d\times m}$, $\mu^{d\times m}(A) = 0 \Rightarrow \nu_X(A) = 0$.*

We have this very important theorem.

**Theorem A.2.1** (Radon–Nikodym derivative)**.** *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space and $X : \Omega \to \mathbb{R}^{d\times m}$ be a matrix valued random variable with distribution $\nu_X \ll \mu^{d\times m}$, then there exists a measurable function $f : \mathbb{R}^{d\times m} \to [0, \infty)$ such that for any $A \in \mathcal{B}^{d\times m}$,*

$$\nu_X(A) = \int_A f d\mu^{d\times m}.$$

*$f$ is called the probability density of $X$ or the Radon–Nikodym derivative of the measure $d\nu_X$ with respect to. $d\mu^{d\times m}$, $f =: \frac{d\nu_X}{d\mu^{d\times m}}$.*

Let $X : \Omega \to \mathbb{R}^{d\times m}$ be absolutely continuous with density $f$. The expectation is a matrix of Lebesgue integrals ($dx := d\mu^{d\times m}$)

$$\mathbb{E}[X] := \begin{bmatrix} \int_{\mathbb{R}^{d\times m}} x_{11} f(x)dx & \int_{\mathbb{R}^{d\times m}} x_{12} f(x)dx & \dots & \int_{\mathbb{R}^{d\times m}} x_{1m} f(x)dx \\ \int_{\mathbb{R}^{d\times m}} x_{21} f(x)dx & \int_{\mathbb{R}^{d\times m}} x_{22} f(x)dx & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \int_{\mathbb{R}^{d\times m}} x_{d1} f(x)dx & \dots & \dots & \int_{\mathbb{R}^{d\times m}} x_{dm} f(x)dx \end{bmatrix}.$$

Moreover, if $X : \Omega \to \mathbb{R}^d$ we can define its covariance matrix like

$$\mathrm{Var}[X] := \mathbb{E}[(X - \mathbb{E}[X])(X - \mathbb{E}[X])^\top].$$

Next, we define the normal distribution, also called Gaussian distribution.

**Definition A.2.6** (Normal distribution)**.** *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. The random variable $X : \Omega \to \mathbb{R}^d$ has normal distribution if it is absolutely continuous and there exists a vector $\mu \in \mathbb{R}^d$ and a matrix $\Sigma \in \mathbb{R}^{d\times d}$ such that the density can be written as*

$$f(x) = \frac{1}{\sqrt{\det(2\pi\Sigma)}} \exp\left(\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right).$$

*We will write fact as $X \sim \mathcal{N}(\mu, \Sigma)$.*

If $\mu$ is the null vector and $\Sigma = I$ (the identity matrix) we call the distribution *standard normal*. We end this subsection listing 3 useful properties of a random variable $X$ having normal distribution $X \sim \mathcal{N}(\mu, \Sigma)$.

1. $\mathbb{E}[X] = \mu$;
2. $\mathrm{Var}[X] = \Sigma$;
3. For any matrix $A \in \mathbb{R}^{m\times d}$ vector $w \in \mathbb{R}^m$, $AX + w$ is an $\mathcal{F}$-adapted random variable taking values in $\mathbb{R}^m$ and distribution $\mathcal{N}(A\mu + w, A\Sigma A^\top)$;

We end this subsection with the definition of independence.

**Definition A.2.7** (Independence). *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space and $I$ be an index set. A collection of sub-$\sigma$-algebras $\{\mathcal{F}_i : i \in I\}$ is said to be independent if for every possible choice of indices $i_1, ..., i_k \in I$,*

$$\mathbb{P}(A_{i_1} \cap ... \cap A_{i_k}) = \mathbb{P}(A_{i_1}) \cdots \mathbb{P}(A_{i_k}),$$

*Holds for all $A_{i_1} \in \mathcal{F}_{i_1}, ..., A_{i_k} \in \mathcal{F}_{i_k}$.*

Moreover, a family of random variables $\{X_i : i \in I\}$ is said to be independent if the corresponding family of induced sub-$\sigma$-algebras are independent.

**Proposition A.2.1** (Correlation and independence). *Let $X_1$ and $X_2$ be two independent real valued random variables. Then, $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$. Moreover, if $X$ and $Y$ have normal distribution, the converse is also true.*

### A.2.2 *Brownian Motion*

We start with a very general definition.

**Definition A.2.8** (Stochastic process). *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. A stochastic process $\{X(t)\}_{a \leq t \leq b}$ in $\mathbb{R}^{d \times m}$ with $0 \leq a \leq b \leq \infty$ is a family of $\mathcal{F}$-measurable random variables taking values in $\mathbb{R}^{d \times m}$.*

As of right now, concepts of time and information flow are not well defined. For this, we need an additional definition.

**Definition A.2.9** (Filtration). *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. A filtration is a family $\{\mathcal{F}(t)\}_{t \geq 0}$ of increasing[4] sub-$\sigma$-algebras of $\mathcal{F}$. A filtration is called right-continuous if $\mathcal{F}(t) = \cup_{s>t}\mathcal{F}(s)$. Moreover, a filtration is said to satisfy the usual conditions if $\mathcal{F}$ is complete, $\{\mathcal{F}(t)\}_{t \geq 0}$ is right-continuous and $\mathcal{F}(0)$ contains all sets of measure 0. We call $(\Omega, \mathcal{F}, \{\mathcal{F}(t)\}_{t \geq 0}, \mathbb{P})$ a filtered probability space.*

A stochastic process $\{X(t)\}_{a \leq t \leq b}$ is called $\mathcal{F}(t)$-adapted if, for every $t \in [a, b]$, $X(t)$ is $\mathcal{F}(t)$-measurable.
We now proceed with the definition of a Brownian motion.

**Definition A.2.10** (Brownian Motion). *Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. A (standard) one dimensional Brownian motion is a real-valued continuous process $\{B(t)\}_{t \geq 0}$ with the following properties:*

4 We mean that if $t \leq s$ then $\mathcal{F}_t \subseteq \mathcal{F}_s$.

1. $B(0) = 0$ *a.s.;*
2. *for* $0 \leq s \leq t < \infty$, $B(t) - B(s) \sim \mathcal{N}(0, t-s)$;
3. *for each sequence* $0 \leq t_0 \leq t_1, \leq t_2 ... \leq t_k \leq \infty$, *the increments* $B(t_i) - B(t_{i-1})$ *are independent for each* $i \in [k]$.

It is simple to see that $B(t)$ induces a filtration (called the natural filtration) on $(\Omega, \mathcal{F}, \mathbb{P})$, defined as

$$\mathcal{F}^B(t) = \sigma(B(s) \ : \ 0 \leq s \leq t).$$

**Proposition A.2.2.** *Let* $(\Omega, \mathcal{F}, \mathbb{P})$ *be a probability space and* $B(t)$ *a Brownian Motion on this space. Then, for each* $0 \leq s \leq t \leq \infty$, $B(t) - B(s)$ *is independent of* $\mathcal{F}(s)$. *Hence, B(t) is a Markov (memoryless) process.*

Moreover, it turns out that $\mathcal{F}^B(t)$ can be modified to satisfy the usual conditions while keeping $B(t)$ adapted.
The definition of $d$-dimensional Brownian Motion is straightforward.

**Definition A.2.11.** *A d-dimensional process* $B(t) = (B_1(t), \cdots B_d(t))$ *is called a d-dimensional Brownian Motion if the* $B_i(t)$ *are one-dimensional independent Brownian motions.*

Again, we can easily build a filtered probability space for a d-dimensional Brownian motion using the product space.
Since in this thesis we mainly deal with Brownian motions in calculations (except in Ch. 6 where we consider the Fractional Brownian Motion but do no perform heavy computations), we always implicitly assume $(\Omega, \mathcal{F}, \{\mathcal{F}(t)\}_{t \geq 0}, \mathbb{P})$ is built from the natural filtration and satisfies the usual conditions.

### A.2.3 *Stochastic Integration*

Let $(\Omega, \mathcal{F}, \{\mathcal{F}(t)\}_{t \geq 0}, \mathbb{P})$ be a complete filtered probability space, we call $\mathcal{L}^p([a,b], \mathbb{R}^d)$, with $p > 0$, the family of $\mathbb{R}^d$-valued $\mathcal{F}(t)$-adapted processes $\{f(t)\}_{a \leq t \leq b}$ such that

$$\int_a^b \|f(t)\|^p dt \leq \infty.$$

Moreover, we denote by $\mathcal{M}^p([a,b],\mathbb{R}^d)$, with $p > 0$, the family of $\mathbb{R}^d$-valued processes $\{f(t)\}_{a\leq t\leq b}$ in $\mathcal{L}([a,b],\mathbb{R}^d)$ such that $\mathbb{E}\left[\int_a^b \|f(t)\|^p dt\right] \leq \infty$. The same definition holds for matrix-valued functions using the Frobenius norm. We first define the Itô integral on the family $\mathcal{M}^0([a,b],\mathbb{R})$, which we define below.

**Definition A.2.12** (Simple process)**.** *A $\mathbb{R}$-valued $\mathcal{F}(t)$- adapted processes $\{g(t)\}_{a\leq t\leq b}$ is in $\mathcal{M}^0([a,b],\mathbb{R}^d)$ if there esists a deterministic partition $a = t_0 < t_1 < ... < t_k = b$ and a set of random variables $\{\xi_i\}_{i=1}^k$ such that $\xi_i$, for each $i \in [k]$, is $\mathcal{F}(t)$-measurable and*

$$g(t) = \xi_0 \mathbb{1}_{[t_0,t_1]}(t) + \sum_{i=1}^{k-1} \xi_i \mathbb{1}_{[t_i,t_i+1]}(t),$$

*where $\mathbb{1}$ is the indicator function.*

One can easily see that $\mathcal{M}^0([a,b],\mathbb{R}) \subset \mathcal{M}^2([a,b],\mathbb{R})$. We first define the Itô integral on this family of simple functions.

**Definition A.2.13.** *Let $g \in \mathcal{M}^0([a,b],\mathbb{R})$, we define*

$$\int_a^b g(t)dB(t) := \sum_{i=}^{k-1} \xi_i (B(t_{i+1}) - B(t_i)).$$

**Remark A.2.1.** *Notice that the just defined stochastic integral is a random variable with expectation 0, because $\xi_i$ and $(B(t_{i+1}) - B(t_i))$ are independent, and $(B(t_{i+1}) - B(t_i))$ has mean zero. In particular, we like to point out that using $\xi_{i+1}$ in the previous definition would change the properties of this integral completely.*

Next, one has to find a way to extend this definition to $\mathcal{M}^2([a,b],\mathbb{R})$. The next lemma will bring a connection.

**Lemma A.2.1.** *For any $f \in \mathcal{M}^2([a,b],\mathbb{R})$ there exist a sequence $\{g_n\}$ of simple processes such that*

$$\lim_{n\to\infty} \mathbb{E}\left[\int_a^b \|f(t) - g_n(t)\|^2 dt\right] = 0.$$

We are ready to define now the Itô integral in $\mathcal{M}^2([a,b],\mathbb{R})$.

**Definition A.2.14** (1 dimensional Itô integral)**.** *Let $f \in \mathcal{M}^2([a,b],\mathbb{R})$ and $\{g_n\}$ a sequence of simple processes such that*

$$\lim_{n\to\infty} \mathbb{E}\left[\int_a^b \|f(t) - g_n(t)\|^2 dt\right] = 0.$$

*The Itô integral of $f$ with respect to. $\{B(t)\}$ is defined by*

$$\int_a^b f(t)dB(t) := \lim_{n\to\infty} \int_a^b g_n(t)dB(t),$$

*where by "*lim*" we mean $\int_a^b f(t)dB(t)$ is the random variable $C$ such that*

$$\lim_{n\to\infty} \mathbb{E}\left[\left\|C - \int_a^b g_n(t)dB(t)\right\|^2\right] = 0.$$

It turns out this definition is independent of the sequence $\{g_n\}$. The extension to multiple dimensions is the same as done in standard analysis.

**Definition A.2.15** (Multidimensional Itô integral)**.** *Let $f \in \mathcal{M}^2([a,b],\mathbb{R}^{d\times m})$. Then the Itô integral $\int_a^b f(t)dB(t)$ with respect to. the m- dimensional Brownian motion $\{B(t)\}$ is a random vector $C$ where each component $C_i$ is defined with a simple matrix multiplication-like rule:*

$$C_i := \sum_{j=0}^{m} \int_a^b f_{i,j}(t)dB_j(t).$$

We end this subsection with a collection of important properties.

**Theorem A.2.2.** *Let $f \in \mathcal{M}^2([a,b],\mathbb{R}^{d\times m})$ and $t \leq \infty$. Then,*

1. $\mathbb{E}\left[\int_0^t f(s)dB(s)\right] = 0$;
2. $\mathbb{E}\left[\left\|\int_0^t f(s)dB(s)\right\|^2\right] = \mathbb{E}\left[\int_0^t \|f(s)\|^2 ds\right]$.

The last property is also called Itô isometry.

### A.2.4 *Itô's Lemma and Dynkin's Formula*

Let us indicate as $\partial_x f(x,t)$ the $d$-dimensional vector of partial derivatives of a scalar function $f : \mathbb{R}^d \times [0,\infty) \to \mathbb{R}$ with respect to each component of $x$. Moreover, we indicate as $\partial_{xx} f(x,t)$ the $d \times d$-matrix of partial derivatives of each component of $\partial_x f(x,t)$ with respect to each component of $x$.
Let $(\Omega, \mathcal{F}, \{\mathcal{F}(t)\}_{t\geq 0}, \mathbb{P})$ be a filtered probability space and $\{h(t)\}_{a\leq t\leq b}$ taking values in $\mathbb{R}^d$ be a $\mathcal{F}$-adapted stochastic process in this space. We will write $h \in \mathcal{L}^p\left(\mathbb{R}_+, \mathbb{R}^d\right)$, with $p > 0$, if $h \in \mathcal{L}^p\left([0,T], \mathbb{R}^d\right)$ for every $T > 0$. Same definition holds for matrix valued functions using the Frobenius norm.

**Definition A.2.16.** *A $d$-dimensional Itô process is an $\mathbb{R}^d$-valued continuous $\mathcal{F}(t)$-adapted process of the form*

$$X(t) = x_0 + \int_0^t f(s)ds + \int_0^t \sigma(s)dB(s),$$

*with $b(t) \in \mathcal{L}^1\left(\mathbb{R}_+, \mathbb{R}^d\right)$ and $\sigma(t) \in \mathcal{L}^2\left(\mathbb{R}_+, \mathbb{R}^{d\times m}\right)$. We shall say that $X(t)$ has the stochastic differential*

$$dX(t) = f(t)dt + g(t)dB(t).$$

We are ready now to show Itô's formula, which appeared for the first time in Itô, 1944.

**Theorem A.2.3.** *(Itô's lemma) Let $X(t)$ be an Itô process with stochastic differential $dX(t) = f(t)dt + g(t)dB(t)$. Let $\mathcal{E}(x,t)$ be twice continuously differentiable in $x$ and continuously differentiable in $t$, taking values in $\mathbb{R}$. Then, $\mathcal{E}(X(t),t)$ is again an Itô process with stochastic differential*

$$d\mathcal{E}(X(t),t) = \partial_t \mathcal{E}(X(t),t))dt + \langle \partial_x \mathcal{E}(X(t),t), f(t)\rangle dt + \frac{1}{2} Tr\left(\partial_{xx}\mathcal{E}(X(t),t)\ \sigma(t)\sigma(t)^\top\right) dt + \langle \mathcal{E}_x(x(t),t), \sigma(t)\rangle dB(t)$$

*Which we sometimes quickly write as*

$$d\mathcal{E} = \partial_t \mathcal{E} dt + \langle \partial_x \mathcal{E}, dX\rangle + \frac{1}{2} Tr\left(\partial_{xx}\mathcal{E}\ \sigma\sigma^\top\right) dt.$$

Following Section 3.7.3 in Geering et al., 2011 and Chapter 4 of Mao, 2007, in the setting of Itô's lemma we can define Itô diffusion differential operator $\mathscr{A}$ as

$$\mathscr{A}(\cdot) = \partial_t(\cdot) + \langle \partial_x(\cdot), b(t) \rangle + \frac{1}{2} Tr\left( \partial_{xx}(\cdot)\, \sigma(t)\sigma(t)^\top \right). \tag{179}$$

It is then clear that, thanks to Itô's lemma,

$$d\mathcal{E}(X(t),t) = \mathscr{A}\mathcal{E}(X(t),t)dt + \langle \mathcal{E}_x(x(t),t), \sigma(t) \rangle dB(t).$$

**Remark A.2.2.** *Sometimes, in applied research (see e.g. Krichene and Bartlett, 2017, He et al., 2018 and Mertikopoulos and Staudigl, 2018) $\mathscr{A}\mathcal{E}(X(t),t)dt$ is denoted as $\mathbb{E}[d\mathcal{E}(X(t),t)]$, using $\mathbb{E}[dB] := 0$. In this thesis, we will use the precise notation.*

For more information on $\mathscr{A}$ and the connection to parabolic PDEs, one can check Chapter 3 of Stroock and Varadhan, 2007.
By Definition A.2.16, we know that, at any time $t > 0$,

$$\mathcal{E}(X(t),t) = \mathcal{E}(x_0,0) + \int_0^t \mathscr{A}\mathcal{E}(X(t),t)dt + \int_0^t \langle \mathcal{E}_x(x(t),t), \sigma(t) \rangle dB(t).$$

Taking the expectation, since $\langle \mathcal{E}_x(x(t),t), \sigma(t) \rangle \in \mathcal{M}^2([0,T],\mathbb{R})$ by Definition A.2.16, the stochastic integral vanishes using Theorem A.2.2, and we have

$$\mathbb{E}[\mathcal{E}(X(t),t)] - \mathcal{E}(x_0,0) = \mathbb{E}\left[ \int_0^t \mathscr{A}\mathcal{E}(X(t),t)dt \right].$$

This result can be generalized for stopping times and is known as Dynkin's formula (Dynkin, 1965).
It is often useful to find an upper bound to the function $\mathbb{E}[\mathcal{E}(X(t),t)]$. One way to find it, which is indeed very useful, is to integrate an upper bound of the diffusion operator.

**Definition A.2.17.** *$u(\mathcal{E}(X,t))$ is called an upper bound on $\mathscr{A}\mathcal{E}(X,t)$ if for any $X \in \mathbb{R}^d$ and any $t > 0$*

$$\mathscr{A}\mathcal{E}(X,t) \leq u(\mathcal{E}(X,t)).$$

We have the following lemma, following directly from Dynkin formula.

**Lemma A.2.2.** *Let $u(\mathcal{E}(X,t))$ be an upper bound on $\mathscr{A}\mathcal{E}(X,t)$, then*

$$\mathbb{E}[\mathcal{E}(X(t),t)] - \mathcal{E}(x_0,0) \leq \mathbb{E}\left[\int_0^t u(\mathcal{E}(X(t),t))dt\right].$$

### A.2.5 *Stochastic Differential Equations*

Stochastic Differential Equations (SDEs) are equations of the form

$$dX(t) = b(X(t),t)dt + \sigma(X(t),t)dB(t).$$

Notice that this is different from what we wrote in Theorem A.2.3, since $X(t)$ also appears on the right hand side. Hence, we need to define what does it mean that $X(t)$ solves an SDE.

**Definition A.2.18** (Solution to an SDE)**.** *Let $X(t)$ be stochastic process defined for $0 \leq t \leq T$, taking values in $\mathbb{R}^d$ and deterministic initial condition $X(0) = x_0$. Let $b(X(t),t)$ and $\sigma(X(t),t)$ be Borel measurable, $X(t)$ is called a solution to the corresponding SDE if*

1. *$X(t)$ is continuous and $\mathcal{F}(t)$-adapted;*
2. *$b(X(t),t) \in \mathcal{L}^1\left([0,T],\mathbb{R}^d\right)$;*
3. *$\sigma(X(t),t) \in \mathcal{L}^2\left([0,T],\mathbb{R}^{d\times m}\right)$;*
4. *For every $t \in [0,T]$*

$$X(t) = x_0 + \int_0^t b(X(t),t)dt + \int_0^t \sigma(X(t),t)dB(t) \quad a.s.$$

*Moreover, the solution $X(t)$ is said to be unique if any other solution $X^*(t)$ is such that*

$$\mathbb{P}\left\{X(t) = X^*(t), \text{ for all } -\tau \leq t \leq T\right\} = 1.$$

Notice that the solution to a SDE is an Itô process, and we are allowed to use Theorem A.2.3. The following theorem gives a sufficient condition on $b(X(t),t)$ and $\sigma(X(t),t)$ for the existence of a solution to the corresponding SDE. This is the analogue of Theorem A.1.2.

**Theorem A.2.4.** *Assume that there exist two positive constants $\bar{K}$ and $K$ such that*

1. *(Global Lipschitz condition) for all $x, y \in \mathbb{R}^d$ and $t \in [0, T]$*

$$\max\{||b(x,t) - b(y,t)||,\ ||\sigma(x,t) - \sigma(y,t)||\} \leq \bar{K}||x - y||^2;$$

2. *(Linear growth condition) for all $x \in \mathbb{R}^d$ and $t \in [0, T]$*

$$\max\{||b(x,t)||,\ ||\sigma(x,t)||\} \leq K(1 + ||x||).$$

*Then, there exists a unique solution $X$ to the corresponding SDE , and $X \in \mathcal{M}^2([0,T], \mathbb{R}^d)$.*

In Mao, 2007, these two conditions are written a bit differently, but they are completely equivalent to ours. Here we write them in this way to match the ones given for ODEs and to avoid confusing the reader.
Next, we discuss simulation of SDEs.

**Definition A.2.19** (The Euler–Maruyama simulation)**.** *The Euler–Maruyama discretization method for discretization an SDE on the interval $[0, T]$ works as follows :*

1. *fix a stepsize $\Delta t$,*
2. *Initialize $\hat{x}_0 = x_0$,*
3. *solve the following equation recursively with the use of a random number generator, until $\Delta t(k+1) \geq T$ :*

$$\hat{x}_{k+1} = \hat{x}_k + \Delta t\ b(\hat{x}_k, k\Delta t) + \sqrt{\Delta t}\ \sigma(\hat{x}_k, k\Delta t) Z_k, \quad Z_k \sim \mathcal{N}(0, I). \tag{180}$$

It is well a well-known fact that the Euler-Maruyama discretization approaches the true process uniformly in the limit $\Delta t \to 0$.
To end this subsection, we are going to see what happens to an SDE if we define a different time variable $\varphi(t)$ using a positive nonincreasing function $\eta(t)$. We define $\varphi(t) = \int_0^t \eta(s)ds$. The following theorem can be found in Øksendal, 1990.

**Theorem A.2.5** (Time change). *Let $X(t)$ be the solution to*

$$dX(t) = b(X(t)) + \sigma(X(t))dB(t).$$

*Then, for any $t > 0$, $X(t)$ has the same distribution as $Y\left(\varphi(t)^{-1}\right)$, where $Y(t)$ is the solution to*

$$dY(t) = \eta(t)b(X(t))dt + \sqrt{\eta(t)}\sigma(X(t))dB(t).$$

**Example A.2.1.** *Take for instance $\eta(t) = 1/t$, then $\varphi(t) = \log(t)$. This means the solution to*

$$dY(t) = \frac{1}{t}b(X(t))dt + \frac{1}{\sqrt{t}}\sigma(X(t))dB(t)$$

*is such that $Y(e^t) = X(t)$ in distribution, for every $t$. This means, $Y(t)$ is the same process, but exponentially slowed down.*

## A.3 OPTIMIZATION ON SMOOTH FUNCTIONS

In this section, we are going to study some of the properties of the function classes used by the optimization community. Most of the propositions and results below can be found in Nesterov, 2013.
In this subsection, $f$ is a function defined in an open set $\mathcal{X} \subseteq \mathbb{R}^d$, taking values in $\mathbb{R}$. We denote by $\mathcal{C}^p(\mathcal{X}, \mathbb{R})$ the family of functions from $\mathcal{X}$ to $\mathbb{R}$ that are $p$-times continuously differentiable.

### A.3.1 *Smoothness and Convexity*

We start from a basic definition.

**Definition A.3.1** (Smoothness). *$f \in \mathcal{C}^p(\mathcal{X}, \mathbb{R})$ is $L$-smooth of order $p$ if*

$$\forall x, y \in \mathcal{X}, \quad ||\nabla^p f(x) - \nabla^p f(y)|| \leq L||x - y||.$$

As a special case, if $p = 1$, we will call the function $L$-smooth (or just smooth). Smoothness is very powerful when combined with convexity.

**Definition A.3.2** (Convexity). *$f \in \mathcal{C}^1(\mathcal{X}, \mathbb{R})$ is called convex[5] if*

$$\forall x, y \in \mathcal{X}, \quad f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle.$$

Most of the times, we will use this inequality in reversed form, for the particular case $y = x^*$, where $x^*$ is a global minimum of $f$:

$$\forall x \in \mathcal{X}, \quad \langle \nabla f(x), x - x^* \rangle \geq f(x) - f(x^*).$$

Here is also an equivalent definition.

**Theorem A.3.1.** *$f \in \mathcal{C}^1(\mathcal{X}, \mathbb{R})$ is convex if and only if*

$$\forall x, y \in \mathcal{X}, \quad \langle \nabla f(x) - \nabla f(y), x - y \rangle \geq 0.$$

We recall a useful property un convex functions in the next theorem.

**Theorem A.3.2** (Jensen inequality). *Let $g : [a, b] \to \mathbb{R}$ be a Lebesgue-integrable function, and let $f : \mathbb{R} \to \mathbb{R}$ be a convex function. Then*

$$f\left(\frac{1}{b-a}\int_a^b g(t)dt\right) \leq \frac{1}{b-a}\int_a^b f(g(t))dt.$$

Smoothness has several consequences.

**Proposition A.3.1.** *Let $f \in \mathcal{C}^1(\mathcal{X}, \mathbb{R})$ be smooth, the following are equivalent:*

$$\forall x, y \in \mathcal{X}, \quad \langle \nabla f(x) - \nabla f(y), x - y \rangle \leq L\|x - y\|^2; \tag{181}$$

$$\forall x \in \mathcal{X}, \quad \frac{L}{2}||x||^2 - f(x) \text{ is convex}; \tag{182}$$

$$\forall x, y \in \mathcal{X}, \quad f(y) \leq f(x) + \langle \nabla f(y), x - y \rangle + \frac{L}{2}\|x - y\|^2. \tag{183}$$

On the other hand, some conditions imply (are sufficient for) smoothness.

**Proposition A.3.2.** *Let $f \in \mathcal{C}^1(\mathcal{X}, \mathbb{R})$. Define the following properties:*

$$\forall x, y \in \mathcal{X}, \quad f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{1}{2L}\|\nabla f(y) - \nabla f(x)\|^2; \tag{184}$$

$$\forall x, y \in \mathcal{X}, \quad \langle \nabla f(x) - \nabla f(y), x - y \rangle \geq \frac{1}{L}\|\nabla f(x) - \nabla f(y)\|^2. \tag{185}$$

*We have that Equation (184) is sufficient for Equation (185) and Equation (185) is sufficient for smoothness.*

5 This definition can be extended to general functions using the concept of subgradient.

**Proposition A.3.3.** *Let $f \in \mathcal{C}^1(\mathcal{X}, \mathbb{R})$ be convex, then Eq. (184), Eq. (185), Eq. (181), Eq. (182) and Eq. (183) are equivalent.*

**Proposition A.3.4.** *$f \in \mathcal{C}^2(\mathcal{X}, \mathbb{R})$ is smooth if and only if $\nabla^2 f \preceq LI$.*

**Proposition A.3.5.** *$f \in \mathcal{C}^2(\mathcal{X}, \mathbb{R})$ is convex if and only if $\nabla^2 f \succeq 0$.*

Let $M$ be any square matrix, we have

$$M \preceq \alpha I \Rightarrow \max(\mathrm{Spec}(M)) \leq \alpha;$$
$$M \succeq \alpha I \Rightarrow \min(\mathrm{Spec}(M)) \geq \alpha.$$

Hence, Propositions A.3.5 and A.3.4 can be easily converted into statements involving the eigenvalues of $\nabla^2 f$.
One can require the landscape to be sufficiently curved or, equivalently, the gradients to be sufficiently big away from the optimum.

**Definition A.3.3** (Strong convexity, $\mu$-SC)**.** *$f \in \mathcal{C}^1(\mathcal{X}, \mathbb{R})$ is said to be strongly convex ($\mu$-SC) if there exist $\mu > 0$ such that*

$$\forall x, y \in \mathcal{X}, \quad f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2}||y - x||^2.$$

One can give some alternative definitions of strong convexity.

**Proposition A.3.6.** *The following conditions are all equivalent to strong convexity:*

$$\forall x \in \mathcal{X}, \quad f(x) - \frac{\mu}{2}||x||^2 \text{ is convex;} \tag{186}$$
$$\forall x, y \in \mathcal{X}, \quad \langle \nabla f(x) - \nabla f(y), x - y \rangle \geq \mu||x - y||^2. \tag{187}$$

It is trivial to see that a strongly convex function can have just one global minimizer, which we denote by $x^*$. Also, strong convexity implies the following properties.

**Proposition A.3.7.** *Let $f \in \mathcal{C}^1(\mathcal{X}, \mathbb{R})$ be $\mu$-SC, then*

$$\forall x \in \mathcal{X}, \quad \frac{1}{2}||\nabla f(x)||^2 \geq \mu(f(x) - f(x^*)); \tag{188}$$
$$\forall x, y \in \mathcal{X}, \quad ||\nabla f(x) - \nabla f(y)|| \geq \mu||x - y||; \tag{189}$$
$$\forall x, y \in \mathcal{X}, \quad f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{1}{2\mu}\|\nabla f(y) - \nabla f(x)\|^2; \tag{190}$$
$$\forall x, y \in \mathcal{X}, \quad \langle \nabla f(x) - \nabla f(y), x - y \rangle \leq \frac{1}{\mu}||\nabla f(x) - \nabla f(y)||^2. \tag{191}$$

Condition Equation (188) is also called Polyak-Łojasiewicz (PŁ) inequality. We are going to work more on this condition in the next subsection.

**Proposition A.3.8.** *$f \in \mathcal{C}^2(\mathcal{X}, \mathbb{R})$ is $\mu$-SC if and only if $\nabla^2 f \succeq \mu I$.*

Again, this proposition can be easily translated into statements involving the eigenvalues of $\nabla^2 f$.

# B

## APPENDIX TO CHAPTER 2

*These motions were such as to satisfy me, after frequently repeated observation, that they arose neither from currents in the fluid, nor from its gradual evaporation, but belonged to the particle itself.*

– Robert Brown.

We start with a recap of some results and notation outlined in Chapter A, applied to the Memory SDE studied in Section 2.3.

### B.1 PROOFS FOR SECTION 2.3

We first review some tools in stochastic analysis and then provide proofs for the two theorems presented in this section.

#### B.1.1 *Stochastic Calculus for the Memory SDE*

Consider the memory SDE

$$\begin{cases} dX(t) = V(t)dt \\ dV(t) = -\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}V(t)dt - \frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\nabla f(X(t))dt - \frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\sigma(X(t),t)dB(t) \end{cases}.$$

We can rewrite this in vector notation ($0_{d\times d}$ is the $d \times d$ of all zeros)

$$\begin{aligned} &\begin{pmatrix} dX(t) \\ dV(t) \end{pmatrix} \\ &= \begin{pmatrix} V(t) \\ -\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}V(t) - \frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\nabla f(X(t)) \end{pmatrix} dt + \begin{pmatrix} 0_{d\times d} & 0_{d\times d} \\ 0_{d\times d} & -\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\sigma(X(t),t) \end{pmatrix} dB(t) \\ &= b(X(t),V(t),t)dt + \xi(X(t),V(t),t)dB(t), \end{aligned} \tag{192}$$

where $\{B(t)\}_{t\geq 0}$ is a d-dimensional Brownian Motion. We write the right-hand-side for simplicity as $b(t)dt + \xi(t)dt$.
Let $\mathcal{E} : \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}$ be twice continuously differentiable jointly in the first two variables (indicated as $x$ and $v$) and continuously differentiable in the last (which we indicate as $t$). Then, by Itô's lemma (Mao, 2007), the stochastic process $\{\mathcal{E}(X(t), V(t), t)\}_{t\geq 0}$ satisfies the following SDE:

$$\begin{aligned} d\mathcal{E}(X(t), V(t), t) = \partial_t \mathcal{E}(X(t), V(t), t))dt + \langle \partial_{(x,v)} \mathcal{E}(X(t), V(t), t), b(t)\rangle dt \\ + \langle \partial_{(x,v)} \mathcal{E}(X(t), V(t), t), \xi(t) dB(t)\rangle \\ + \frac{1}{2} Tr\left(\xi(t)\xi(t)^\top \partial^2_{(x,v)} \mathcal{E}(X(t), V(t), t)\right) dt. \end{aligned}$$

where $\partial_{(x,v)}$ is the partial derivative with respect to $(x, v)$ and $\partial^2_{(x,v)}$ the matrix of second derivatives with respect to $(x, v)$. Notice that, in the deterministic case $\xi(t) = 0$, the equation reduces to standard differentiation using the chain rule:

$$\frac{d\mathcal{E}(X(t), V(t), t)}{dt} = \partial_t \mathcal{E}(X(t), V(t)t)) + \langle \partial_{(x,v)} \mathcal{E}(X(t), V(t), t), b(t)\rangle.$$

The Itô diffusion differential operator $\mathscr{A}$ associated with Eq. (192) :

$$\mathscr{A}(\cdot) = \partial_t(\cdot) + \langle \partial_{(x,v)}(\cdot), b(t)\rangle + \frac{1}{2} Tr\left(\xi(t)\xi(t)^\top \partial^2_{(x,v)}(\cdot)\right).$$

It is then clear that, thanks to Itô's lemma (see also Eq. (179) in Chapter A),

$$d\mathcal{E}(X(t), V(t), t) = \mathscr{A}\mathcal{E}(X(t), V(t), t)dt + \langle \partial_{(x,v)} \mathcal{E}(X(t), V(t), t), \xi(t) dB(t)\rangle.$$

The Itô diffusion differential operator generalizes the concept of derivative: in fact, with a slight abuse of notation:

$$\frac{\mathbb{E}\left[d\mathcal{E}(X(t), V(t), t)\right]}{dt} = \mathscr{A}\mathcal{E}(X(t), V(t), t).$$

Moreover, by the definition of the solution to an SDE (see Mao, 2007), we know that at any time $t > 0$,

$$\begin{aligned} &\mathcal{E}(X(t), V(t), t) \\ &= \mathcal{E}(x_0, v_0, 0) + \int_0^t \mathscr{A}\mathcal{E}(X(s), V(s), s)ds + \int_0^t \partial_x \mathcal{E}(X(s), V(s), s)^\top \sigma(s) dB(s). \end{aligned}$$

Taking the expectation the stochastic integral vanishes[1] and we have

$$\mathbb{E}[\mathcal{E}(X(t),V(t),t)] - \mathcal{E}(x_0,0) = \mathbb{E}\left[\int_0^t \mathscr{A}\mathcal{E}(X(s),V(s),s)ds\right].$$

This result is known as **Dynkin's formula** and generalizes the fundamental theorem of calculus to the stochastic setting.
The next fundamental lemma can also be found in Krichene and Bartlett, 2017, and we will make heavy use of it in our proofs.

**Lemma B.1.1.** *Consider two symmetric $d-$dimensional square matrices $P$ and $Q$. We have*

$$Tr(PQ) \leq d \cdot \|P\|_s \cdot \|Q\|_s,$$

*where $\|\cdot\|_s$ denotes the spectral norm.*

*Proof.* Let $P_j$ and $Q_j$ be the j-th row(column) of $P$ and $Q$, respectively.

$$Tr(PQ) = \sum_{j=1}^{d} P_j^\top Q_j \leq \sum_{j=1}^{d} \|P_j\| \cdot \|Q_j\| \leq \sum_{j=1}^{d} \|P\|_s \cdot \|Q\|_s = d \cdot \|P\|_s \cdot \|Q\|_s,$$

where we first used the Cauchy-Schwarz inequality, and then the following inequality:

$$\|A\|_s = \sup_{\|z\|\leq 1} \|Az\| \geq \|Ae_j\| = \|A_j\|,$$

where $e_j$ is the j-th vector of the canonical basis of $\mathbb{R}^d$. □

### B.1.2 *Proofs of Convergence for Memory Systems*

*Proof of Lemma 2.3.2.* Consider the following Lyapunov function, inspired from Su et al., 2016:

$$\mathcal{E}(x,v,t) = r(t)(f(x) - f(x^*)) + \frac{1}{2}\|x - x^* + \lambda(t)v\|^2,$$

where $r : \mathbb{R} \to \mathbb{R}$ and $\lambda : \mathbb{R} \to \mathbb{R}$ are two differentiable functions which we will fix during the proof. First, we find a bound on the infinitesimal diffusion generator of the stochastic process $\{\mathcal{E}(X(t),V(t),t)\}_{t\geq 0}$. Ideally,

1 see e.g. Thm. 1.5.8 in Mao, 2007

we want this bound to be independent of the dynamics (i.e the solution $\{(X(t), V(t))\}_{t\geq 0}$) of the problem, so that we can integrate it and get a rate.

By Itô's lemma, we know that

$$\begin{aligned}\mathscr{A}\mathcal{E}(X(t),V(t),t) &= \partial_t \mathcal{E}(X(t),V(t),t))dt \\ &\quad + \langle \partial_{(x,v)}\mathcal{E}(X(t),V(t),t), b(t)\rangle dt \\ &\quad + \frac{1}{2} Tr\left(\xi(t)\xi(t)^\top \partial_x^2 \mathcal{E}(X(t),V(t),t)\right) dt.\end{aligned}$$

Plugging in the SDE definition and the definition of $\mathcal{E}$,

$$\begin{aligned}&\mathscr{A}\mathcal{E}(X(t),V(t),t) \\ &= \dot{r}(t)(f(X-f(x^*))dt + 2\langle X - x^* + \lambda(t)V, \dot{\lambda}(t)V\rangle dt \\ &\quad + r(t)\langle \nabla f(X), V\rangle dt + \langle X - x^* + \lambda(t)V, V\rangle dt \\ &\quad + \lambda(t)\left\langle X - x^* + \lambda(t)V, -\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}V - \frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\nabla f(X)\right\rangle dt \\ &\quad + \frac{1}{2} Tr\left(\xi(t)\xi(t)^\top \partial^2_{(x,v)} \mathcal{E}(X(t),V(t),t)\right) dt.\end{aligned}$$

Next, we group some terms together,

$$\begin{aligned}\mathscr{A}\mathcal{E}(X(t),V(t),t) &= \dot{r}(t)(f(X)-f(x^*))dt - \lambda(t)\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\langle \nabla f(X), X - x^*\rangle dt \\ &\quad + \left(\dot{\lambda}(t) + 1 - \lambda(t)\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\right)\langle X - x^* + \lambda(t)V, V\rangle\, dt \\ &\quad + \left(r(t) - \lambda(t)^2\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\right)\langle \nabla f(X), V\rangle\, dt \\ &\quad + \frac{1}{2} Tr\left(\xi(t)\xi(t)^\top \partial^2_{(x,v)} \mathcal{E}(X(t),V(t),t)\right) dt.\end{aligned}$$

Using **(H1')** ($\tau$-weak quasi-convexity), we conclude

$$\begin{aligned}\mathscr{A}\mathcal{E}(X(t),V(t),t) \leq & \left(\dot{r}(t)-\tau\lambda(t)\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\right)(f(X)-f(x^*))dt\\ &+\left(\dot{\lambda}(t)+1-\lambda(t)\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\right)\langle X-x^*+\lambda(t)V,V\rangle\,dt\\ &+\left(r(t)-\lambda(t)^2\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\right)\langle\nabla f(X),V\rangle\,dt\\ &+\frac{1}{2}Tr\left(\xi(t)\xi(t)^\top\partial^2_{(x,v)}\mathcal{E}(X(t),V(t),t)\right)dt.\end{aligned}$$

Under the hypotheses of this lemma, since $\dot{\lambda}(t)=\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\lambda(t)-1$ if and only if $\lambda(t)=-\mathfrak{m}(t)\int\frac{1}{\mathfrak{m}(t)}dt$, we are left with

$$\begin{aligned}\mathscr{A}\mathcal{E}(X(t),V(t),t) &\leq \frac{1}{2}Tr\left(\xi(t)\xi(t)^\top\partial^2_{(x,v)}\mathcal{E}(X(t),V(t),t)\right)dt\\ &=\frac{d}{2}\left(\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\right)^2 Tr\left(\sigma(t)\sigma(t)^\top\partial^2_v\mathcal{E}(X(t),V(t),t)\right)dt\\ &\leq\frac{d}{2}\left(\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\right)^2\|\sigma(t)\sigma(t)^\top\|_s\|\partial^2_v\mathcal{E}(X(t),V(t),t)\|_s dt\\ &\leq\frac{d}{2}\sigma_*^2\lambda(t)^2\left(\frac{\dot{\mathfrak{m}}(t)}{\mathfrak{m}(t)}\right)^2 dt,\end{aligned}$$

where in the first inequality we used Lemma B.1.1 and the definition of $\sigma_*^2$ in **(H0')** (see Sec. 2.3). Finally, by Dynkin's formula

$$\mathbb{E}[\mathcal{E}(X(t),V(t),t)]-\mathcal{E}(x_0,0)\leq\frac{d\sigma_*^2}{2}\int_0^t\lambda(s)^2\left(\frac{\dot{\mathfrak{m}}}{\mathfrak{m}}(s)\right)^2 ds;$$

therefore

$$\begin{aligned}&r(t)\mathbb{E}[f(X(t))-f(x^*)]+\mathbb{E}\left[\frac{1}{2}\|X(t)-x^*+\lambda(t)V\|^2\right]\\ &\leq r(0)(f(x_0)-f(x^*))+\frac{1}{2}\|x_0-x^*\|^2+\frac{d\sigma_*^2}{2}\int_0^t\lambda(s)^2\left(\frac{\dot{\mathfrak{m}}}{\mathfrak{m}}(s)\right)^2 ds,\end{aligned}$$

which implies

$$r(t)\mathbb{E}[f(X) - f(x^*)]$$
$$\leq r(0)(f(x_0) - f(x^*)) + \frac{1}{2}\|x_0 - x^*\|^2 + \frac{d\sigma_*^2}{2}\int_0^t \lambda(s)^2 \left(\frac{\dot{\mathfrak{m}}}{\mathfrak{m}}(s)\right)^2 ds.$$

□

*Proof of Lemma 2.3.3.* Consider the Lyapunov function **inspired by the continuous-time** setting in Lemma 2.3.1.

$$\mathcal{E}_k = r_k(f(x_k) - f(x^*)) + \frac{1}{2}\|x_{k+1} - x^* + \lambda_k m_{k+1}\|^2. \tag{193}$$

First, notice that

$$\begin{aligned} x_{k+1} - x^* + \lambda_{k+1} m_{k+1} &\overset{(39)}{=} x_k - x^* + (1 + \lambda_{k+1}) m_{k+1} \\ &\overset{(38)}{=} x_k - x^* + (1 + \lambda_{k+1})(\beta_k m_k - \delta_k \eta \nabla f_{i_k}(x_k)) \\ &= x_k - x^* + \lambda_k m_k - \eta \nabla f_{i_k}(x_k)), \end{aligned}$$

where in the last line we chose $\lambda_k = (\lambda_{k+1} + 1)\beta_k$ and $\delta_k = \frac{1}{\lambda_k + 1}$. Consider $\zeta_k := \nabla f_{i_k}(x_k) - \nabla f(x_k)$, then

$$\begin{aligned} &\mathbb{E}\left[\|x_{k+1} - x^* + \lambda_{k+1} m_{k+1}\|^2\right] \\ &= \mathbb{E}\left[\|x_k - x^* + \lambda_k m_k - \eta \nabla f(x_k)) + \eta \zeta_k\|^2\right] \\ &= \mathbb{E}\left[\|x_k - x^* + \lambda_k m_k\|^2\right] + \mathbb{E}\left[\|\eta \nabla f(x_k) + \eta \zeta_k\|^2\right] \\ &\quad - 2\eta \mathbb{E}\left[\langle \nabla f(x_k), x_k - x^* + \lambda_k m_k \rangle\right] \\ &= \mathbb{E}\left[\|x_k - x^* + \lambda_k m_k\|^2\right] + \eta^2 \mathbb{E}\left[\|\nabla f(x_k)\|^2\right] + \eta^2 \mathbb{E}\left[\|\zeta_k\|^2\right] \\ &\quad - 2\eta \mathbb{E}\left[\langle \nabla f(x_k), x_k - x^* \rangle\right] - 2\eta\lambda_k \mathbb{E}\left[\langle f(x_k), m_k \rangle\right]. \end{aligned}$$

Since $f(\cdot)$ is convex and smooth, it follows from Thm. 2.1.5 in Nesterov et al., 2018 that

$$\begin{aligned} \frac{1}{L}\|\nabla f(x_k)\|^2 &\leq \langle x_k - x^*, \nabla f(x_k) \rangle, \\ f(x_k) - f(x^*) + \frac{1}{2L}\|\nabla f(x_k)\|^2 &\leq \langle x_k - x^*, \nabla f(x_k) \rangle, \\ f(x_k) - f(x_{k-1}) &\leq \langle x_k - x_{k-1}, \nabla f(x_k)) \rangle \end{aligned}$$

for all $x_k$. Next, notice that $\zeta_k$ is a random variable with mean 0 and we denote its covariance by $\Sigma(x_k) \geq 0$. Moreover,

$$\mathbb{E}\left[\|\zeta_k\|^2\right] = \mathbb{E}\left[Tr(\zeta_k\zeta_k^\top)\right] = Tr(\mathbb{E}\left[\zeta_k\zeta_k^\top\right]) = Tr(\Sigma(x_k)) \leq d\|\Sigma(x_k)\|_s = d\varsigma_*^2.$$

Let us assume $k \geq 1$, then

$$\begin{aligned}
&\mathbb{E}\left[\|x_{k+1} - x^* + \lambda_{k+1}m_{k+1}\|^2\right] \\
&= \mathbb{E}\left[\|x_k - x^* + \lambda_k m_k\|^2\right] + \eta^2\mathbb{E}\left[\|\nabla f(x_k)\|^2\right] + d\eta^2\varsigma_*^2 \\
&\quad - 2\eta\mathbb{E}\left[\langle\nabla f(x_k), x_k - x^*\rangle\right] - 2\eta\lambda_k\mathbb{E}\left[\langle\nabla f(x_k), x_k - x_{k-1}\rangle\right] \\
&\leq \mathbb{E}\left[\|x_k - x^* + \lambda_k m_k\|^2\right] + \eta^2\mathbb{E}\left[\|\nabla f(x_k)\|^2\right] + d\eta^2\varsigma_*^2 \\
&\quad - 2\eta\mathbb{E}\left[f(x_k) - f(x^*) + \frac{1}{2L}\|\nabla f(x_k)\|^2\right] - 2\eta\lambda_k\mathbb{E}\left[f(x_k) - f(x_{k-1})\right] \\
&\leq \mathbb{E}\left[\|x_k - x^* + \lambda_k m_k\|^2\right] + \eta\left(\eta - \frac{1}{L}\right)\mathbb{E}\left[\|\nabla f(x_k)\|^2\right] + d\eta^2\varsigma_*^2 \\
&\quad - 2\eta(1+\lambda_k)\mathbb{E}\left[f(x_k) - f(x^*)\right] + 2\eta\lambda_k\mathbb{E}\left[f(x_{k-1}) - f(x^*)\right].
\end{aligned}$$

Then, let $\eta \leq 1/L$, note that $\mathbb{E}\left[f(x_k) - f(x^*)\right] \geq 0, \forall k$ and assume $\lambda_k \leq \lambda_{k-1} + 1$. As a result, we have

$$\begin{aligned}
&\mathbb{E}\left[\eta(1+\lambda_k)(f(x_k) - f(x^*)) + \frac{1}{2}\|x_{k+1} - x^* + \lambda_{k+1}m_{k+1}\|^2\right] \\
&\leq \mathbb{E}\left[\eta(1+\lambda_{k-1})(f(x_{k-1}) - f(x^*)) + \frac{1}{2}\|x_k - x^* + \lambda_k m_k\|^2\right] + \frac{d\eta^2\varsigma_*^2}{2}.
\end{aligned}$$

From this we immediately get, choosing $k = 0$ and dropping the first term, an important inequality,

$$\begin{aligned}
&\mathbb{E}\left[\frac{1}{2}\|x_1 - x^* + \lambda_1 m_1\|^2\right] \\
&\qquad \leq \eta(1+\lambda_0)(f(x_0) - f(x^*)) + \frac{1}{2}\|x_0 - x^*\|^2 + \frac{d\eta^2\varsigma_*^2}{2}, \quad (194)
\end{aligned}$$

where we used the fact that $m_0 = 0$.

Recalling the definition of our Lyapunov function in Eq. (193) and choosing $r_k := 1 + \lambda_k$, the last inequality reads as $\mathbb{E}[\mathcal{E}_k - \mathcal{E}_{k-1}] \leq \frac{d\eta^2}{2}\varsigma_*^2$ for $k \geq 1$. Summing over $k = 1, \cdots, K$

$$\sum_{k=1}^{K} \mathbb{E}\left[\eta(1+\lambda_k)(f(x_k) - f(x^*)) + \frac{1}{2}\|x_{k+1} - x^* + \lambda_{k+1}m_{k+1}\|^2\right]$$
$$\leq \sum_{k=1}^{K} \mathbb{E}\left[\eta(1+\lambda_{k-1})(f(x_{k-1}) - f(x^*)) + \frac{1}{2}\|x_k - x^* + \lambda_k m_k\|^2\right] + \frac{d\eta^2\varsigma_*^2 K}{2}.$$

Simplifying the sum, we get

$$\mathbb{E}\left[\eta(1+\lambda_K)(f(x_K) - f(x^*)) + \frac{1}{2}\|x_{K+1} - x^* + \lambda_{K+1}m_{K+1}\|^2\right]$$
$$\leq \mathbb{E}\left[\eta(1+\lambda_0)(f(x_0) - f(x^*)) + \frac{1}{2}\|x_1 - x^* + \lambda_k m_1\|^2\right] + \frac{d\eta^2\varsigma_*^2 K}{2}.$$

dropping the second term in the first expectation (positive), we get (using also Eq. (194))

$$\mathbb{E}\left[f(x_K) - f(x^*)\right]$$
$$\leq \frac{\mathbb{E}\left[\eta(1+\lambda_0)(f(x_0) - f(x^*)) + \frac{1}{2}\|x_1 - x^* + \lambda_k m_1\|^2\right]}{\eta(1+\lambda_K)} + \frac{d\eta^2\varsigma_*^2 K}{2\eta(1+\lambda_K)}$$
$$\leq \frac{2\eta(1+\lambda_0)(f(x_0) - f(x^*)) + \frac{1}{2}\|x_0 - x^*\|^2}{\eta(1+\lambda_K)} + \frac{d\eta^2\varsigma_*^2(1+K)}{2\eta(1+\lambda_K)}.$$

□

## B.2 PROOFS AND ADDITIONAL MATERIAL FOR SECTION 2.4

In Section 2.4 we work in $\mathbb{R}^d$ with the Euclidean norm $\|\cdot\|$. If $A$ is a matrix, $\|A\|$ denotes its supremum norm: $\|A\| = \sup_{\|y\|=1}\|Ay\|$. We denote by $\mathcal{C}^r(\mathbb{R}^n, \mathbb{R}^m)$ the family of $r$-times continuously differentiable functions from $\mathbb{R}^n$ to $\mathbb{R}^m$. We denote by $Dg \in \mathbb{R}^{m\times n}$ the Jacobian (matrix of partial derivatives) of $g : \mathbb{R}^n \to \mathbb{R}^m$; if $m = 1$ then we denote its gradient by $\nabla g$. If the dimensions are clear, we write $\mathcal{C}^r$. We seek to minimize $f : \mathbb{R}^d \to \mathbb{R}$. We list below some assumptions we will refer to.

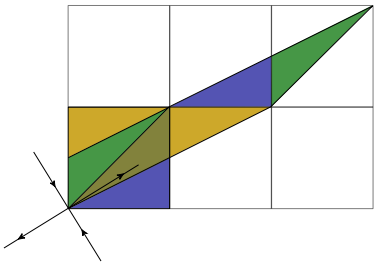

**Figure 62:** *(From Wikipedia) The cat map stretches the unit square and how its pieces are rearranged. The cat map is Anosov. Shown are the directions of the global splitting* $\mathbb{R}^n = E_s \oplus E_u$.

**(H1)** $f \in \mathcal{C}^2(\mathbb{R}^d, \mathbb{R})$ is coercive[2], bounded from below and $L$-smooth ($\forall a \in \mathbb{R}^d$, $\|\nabla^2 f(a)\| \leq L$).

**(H2)** $f$ is $\mu$-strongly-convex: for all $a \in \mathbb{R}^d$, $\|\nabla^2 f(a)\| \geq \mu$.

### B.2.1 *Hyperbolic Sets*

We state here some useful details on shadowing for the interested reader. Also, we propose a simple proof of the expansion map shadowing theorem based on Ombach, 1993.

#### B.2.1.1 *Shadowing near Hyperbolic Sets*

We first provide the definitions and results needed to state the shadowing theorem (Anosov, 1967) precisely. Our discussion is based on Lanford, 1985. Let $\Psi : \mathbb{R}^n \to \mathbb{R}^n$ be a diffeomorphism.

**Definition B.2.1.** *We say* $x \in \mathbb{R}^n$ *is an* ***hyperbolic point*** *for* $\Psi$ *if there exist a splitting* $\mathbb{R}^n = E_s(x) \oplus E_u(x)$ *in linear subspaces such that*

$$\|D(\Psi^k(x))\xi\| \leq c\lambda^k\|\xi\| \quad \text{for all } \xi \in E_s(x) \text{ and for all } k \in \mathbb{N},$$
$$\|D(\Psi^{-k}(x))\xi\| \leq c\lambda^k\|\xi\| \quad \text{for all } \xi \in E_u(x) \text{ and for all } k \in \mathbb{N},$$

*where* $\lambda$ *and* $c$ *can be taken to depend only on* $x$*, not on* $\xi \in E_u(x)$.

2 $f(x) \to \infty$ as $\|x\| \to \infty$

**Remark B.2.1.** *It can be showed that, if $x$ is hyperbolic for $\Psi$, then also $\Psi(x)$ and $\Psi^{-1}(x)$ are hyperbolic points. Moreover, using the chain rule, we have*

$$E_s(\Psi(x)) = D\Psi(x)E_s(x) \text{ and } E_u(\Psi(x)) = D\Psi(x)E_u(x).$$

**Definition B.2.2.** *A compact set $\Lambda \subset \mathbb{R}^n$ is said to be an **hyperbolic set** for $\Psi$ if it is invariant for $\Psi$ (i.e. $\Psi(\Lambda) = \Lambda$), each $x \in \Lambda$ is hyperbolic and $c, \lambda$ (see Def. B.2.1) can be taken to be independent of $x$. If the entire space is hyperbolic, then $\Psi$ is called an **Anosov diffeomorphism** (from Anosov, 1967).*

We now state the main result in the literature, originally proved in Bowen, 1975, which essentially tells us that pseudo-orbits sufficiently near an hyperbolic set are shadowed.

**Theorem B.2.1** (Shadowing theorem)**.** *Let $\Lambda$ be an hyperbolic set for $\Psi$, and let $\epsilon > 0$. Then there exists $\delta > 0$ such that, for every $\delta$-pseudo-orbit $(y_k)_{k=0}^{\infty}$ with $\|y_k - \Lambda\| \leq \delta$ for all $k$, there is $x_0$ such that*

$$\|y_k - \Psi^k(x_0)\| \leq \epsilon \text{ for all } k \in \mathbb{N}.$$

*Furthermore, if $\epsilon$ is small enough, $x_0$ is unique.*

EXAMPLE. The most famous example of an Anosov diffeomorphism is Arnold's cat map (Brin and Stuck, 2002) on the torus $\mathbb{T}^2$, which can be thought of as the quotient space $\mathbb{R}^2/\mathbb{Z}^2$. Shown in Fig. 62, details in Brin and Stuck, 2002.

#### B.2.1.2 *Expansion Map Shadowing Theorem*

First, we remind to the reader an important result in analysis.

**Theorem B.2.2** (Banach fixed-point theorem)**.** *Let $(Z, d)$ be a non-empty complete metric space with a contraction mapping $T : Z \to Z$. Then $T$ admits a unique fixed-point $z^* \in Z$ (i.e. $T(z^*) = z^*$). Furthermore, $z^*$ can be found as follows: start with an arbitrary element $z_0$ in $Z$ and iteratively apply $T$; $z^*$ is the limit of this sequence.*

We recall that $\Psi$ is said to be **uniformly expanding** if there exists $\rho > 1$ (*expansion factor*) such that for all $x_1, x_2 \in \mathbb{R}^n$, $\|\Psi(x_1) - \Psi(x_2)\| \geq \rho\|x_1 - x_2\|$. The next result is adapted from Prop. 1 in Ombach, 1993.

**Theorem B.2.3** (Expansion map shadowing theorem). *If $\Psi$ is uniformly expanding, then for every $\epsilon > 0$ there exists $\delta > 0$ such that every $\delta-$pseudo-orbit $(y_k)_{k=0}^{\infty}$ of $\Psi$ is $\epsilon-$shadowed by the orbit $(x_k)_{k=0}^{\infty}$ of $\Psi$ starting at $x_0 = \lim_{k\to\infty} \Psi^{-k}(y_k)$, that is $x_k := \Psi^k(x_0)$. Moreover,*

$$\delta \leq \left(1 - \frac{1}{\rho}\right)\epsilon. \tag{195}$$

*Proof.* Fix $\epsilon > 0$ and define $\delta = (1 - 1/\rho)\epsilon$. Let $(y_k)_{k=0}^{\infty}$ be a $\delta$-pseudo-orbit of $\Psi$ and extend it to negative iterations: $y_{-k} := \Psi^{-k}(y_0)$. We call the resulting sequence $y = (y_k)_{k\in\mathbb{Z}}$. We consider the set $Z$ of sequences which are pointwise close to $y$:

$$Z = \{z : z = (z_k)_{k\in\mathbb{Z}}, \|z_k - y_k\| \leq \epsilon \text{ for all } k \in \mathbb{Z}\}.$$

We endow $Z$ with the supremum metric $d$, defined as follows:

$$d(z, w) = \sup_{k\in\mathbb{Z}} \|z_k - w_k\|.$$

It is easy to show that $(Z, d)$ is complete. Next, we define an operator $T$ on sequences in $\mathbb{Z}$, such that

$$[T(z)]_k = \Psi^{-1}(z_{k+1}) \text{ for all } k \in \mathbb{Z}.$$

Notice that, if $T(x) = x$, then $(x_k)_{k=0}^{\infty}$ is an orbit of $\Psi$. If, in addition $x \in Z$, then by definition we have that $(x_k)_{k=0}^{\infty}$ shadows $(y_k)_{k=0}^{\infty}$. We clearly want to apply Thm. B.2.2, and we need to verify that

1. $T(Z) \subseteq Z$. This follows from the fact that $\Psi^{-1}$ is a contraction with contraction factor $1/\rho$, similarly to the contraction map shadowing theorem (see Thm. 2.4.2). Let $z \in Z$, for all $k \in \mathbb{Z}$

$$\begin{aligned}
&\|\Psi^{-1}(z_{k+1}) - y_k\| \\
&\quad\leq \quad \|\Psi^{-1}(z_{k+1}) - \Psi^{-1}(y_{k+1})\| + \|\Psi^{-1}(y_{k+1}) - y_k\| \\
&\quad\overset{\delta\text{-pseudo-orbit}}{\leq} \quad \|\Psi^{-1}(z_{k+1}) - \Psi^{-1}(y_{k+1})\| + \delta \\
&\quad\overset{\text{contraction}}{\leq} \quad \frac{1}{\rho}\|(z_{k+1}) - y_{k+1}\| + \delta \\
&\quad\overset{z\in Z}{\leq} \quad \frac{1}{\rho}\epsilon + \delta = \epsilon.
\end{aligned}$$

2. $T$ is itself a contraction in $(Z, d)$, since

$$\begin{aligned} d(T(z), T(w)) &= \sup_{k\in\mathbb{Z}} \|\Psi^{-1}(z_{k+1}) - \Psi^{-1}(w_{k+1})\| \\ &\leq \frac{1}{\rho} \sup_{k\in\mathbb{Z}} \|z_{k+1} - w_{k+1}\| \\ &\leq \frac{1}{\rho} d(z, w). \end{aligned}$$

The statement of the expansion map shadowing theorem follows then directly from the Banach fixed-point theorem applied to $T$, which is a contraction on $(Z, d)$. □

### B.2.2 *Shadowing in Optimization*

We present here the proofs of some results we claim in the thesis section.

#### B.2.2.1 *Non-expanding maps (i.e. the convex setting)*

**Proposition B.2.1.** *If $\Psi$ is uniformly non-expanding, then for $\delta-$pseudo-orbit $(y_k)_{k=0}^{\infty}$ of $\Psi$ is such that the orbit $(x_k)_{k=0}^{\infty}$ of $\Psi$ starting at $x_0 = y_0$, satisfies $\|x_k - y_k\| \leq \delta k$ for all $k$.*

*Proof.* The proposition is trivially true at $k = 0$; next, we assume the proposition holds at $k \in \mathbb{N}$ and we show validity for $k + 1$. We have

$$\begin{aligned} \|x_{k+1} - y_{k+1}\| &\overset{\text{subadditivity}}{\leq} \|\Psi(x_k) - \Psi(y_k)\| + \|\Psi(y_k) - y_{k+1}\| \\ &\overset{\delta\text{-pseudo-orbit}}{\leq} \|\Psi(x_k) - \Psi(y_k)\| + \delta \\ &\overset{\text{non-expansion}}{\leq} \|x_k - y_k\| + \delta \\ &\overset{\text{induction}}{\leq} k\delta + \delta = (k+1)\delta. \end{aligned}$$

□

The GD map on a convex function is non-expanding, hence this result bounds the GD-ODE approximation, which grows slowly as a function of the number of iterations.

#### B.2.2.2 *Perturbed Contracting Maps*

**Proposition B.2.2.** *Assume $\Psi$ is uniformly contracting with contraction factor $\rho$. Let $\tilde{\Psi}$ be the perturbed dynamical system, that is $\tilde{\Psi}(x) = \Psi(x) + \zeta$ where $\|\zeta\| \leq D$. Let $(y_k)_{k=0}^{\infty}$ be a $\delta-$pseudo-orbit of $\Psi$ and we denote by $(\tilde{x}_k)_{k=0}^{\infty}$ the orbit of of $\tilde{\Psi}$ starting at $\tilde{x}_0 = y_0$, that is $\tilde{x}_k := \tilde{\Psi}^k(\tilde{x}_0)$. We have $\epsilon$-shadowing under*

$$\delta \leq (1-\rho)\epsilon - D.$$

*Proof.* Again as in Thm. 2.4.2, we proceed by induction: the proposition is trivially true at $k = 0$, since $\|\tilde{x}_0 - y_0\| \leq \epsilon$; next, we assume the proposition holds at $k \in \mathbb{N}$ and we show validity for $k+1$. We have

$$\begin{aligned}
\|\tilde{x}_{k+1} - y_{k+1}\| &\overset{\text{subadditivity}}{\leq} \|\Psi(\tilde{x}_k) + \zeta - \Psi(y_k)\| + \|\Psi(y_k) - y_{k+1}\| \\
&\overset{\delta\text{-pseudo-orbit}}{\leq} \|\Psi(\tilde{x}_k) - \Psi(y_k)\| + D + \delta \\
&\overset{\text{contraction}}{\leq} \rho\|\tilde{x}_k - y_k\| + D + \delta \\
&\overset{\text{induction}}{\leq} \rho\epsilon + D + \delta.
\end{aligned}$$

Since $\delta \leq (1-\rho)\epsilon - D$, $\rho\epsilon + \delta + D = \epsilon$. □

In the setting of shadowing GD, $\rho = 1 - \mu h$ (Prop. 2.4.3), $\delta = \frac{\ell L h^2}{2}$, and $D = h\|\nabla f(x) - \tilde{\nabla} f(x)\| \leq Rh$ which gives the consistency equation

$$\frac{\ell L h^2}{2} \leq (1-\rho)\epsilon - D \leq \mu h\epsilon - Rh \implies h \leq \frac{2(\epsilon\mu - R)}{\ell L}.$$

#### B.2.2.3 *Hyperbolic Maps (Quadratic Saddle Setting)*

We prove the result for GD. This can be generalized to HB. A more powerful proof technique — which can tackle the effect of perturbations — can be found in the appendix of Orvieto and Lucchi, 2019b.

**Proposition B.2.3** (restated Prop. 2.4.4)**.** *Let $f$ be quadratic with Hessian $H$ which has no eigenvalues in the interval $(-\gamma, \mu)$, for some $\mu, \gamma > 0$. Assume the orbit $(y_k)_{k=0}^{\infty}$ of $\varphi_h^{GD}$ is such that **(H1)** holds up to iteration $K$. Let $\epsilon$ be*

*the desired tracking accuracy; if* $0 < h \leq \min\left\{\frac{\mu\epsilon}{L\ell}, \frac{\gamma\epsilon}{2L\ell}, \frac{1}{L}\right\}$, *then* $(y_k)_{k=0}^{\infty}$ *is* $\epsilon$*-shadowed by a orbit* $(x_k)_{k=0}^{\infty}$ *of* $\Psi_h^{GD}$ *up to iteration* $K$.

*Proof.* From Prop. 2.4.2 and thanks to the hypothesis, we know $(y_k)_{k=0}^{K}$ is a partial $\delta$-pseudo-orbit of $\Psi_h^{\mathrm{GD}}$ with $\delta = \frac{\ell L}{2}h^2$. By the triangle inequality, this property is preserved when projecting both sequences on a subspace. We project the sequences (the orbit and the pseudo-orbit) onto the stable and unstable subspaces ($E_s$ and $E_u$) of $H$. Since these are invariant (they are eigenspaces), shadowing in each space implies shadowing in the whole $\mathbb{R}^d$ (Ombach, 1993). On the stable subspace we require, by the same argument of Thm. 2.4.3, $\delta \leq \mu h\epsilon$ which gives the condition $h \leq \frac{2\epsilon\mu}{\ell L}$. On the unstable space, reversing the arrow of time (see discussion in the thesis) and by Thm. B.2.3, we require $\delta \leq \left(1 - \frac{1}{1+\gamma h}\right)\epsilon = \frac{\gamma h}{1+\gamma h}\epsilon$. Since $\gamma \leq L$ and $h \leq \frac{1}{L}$, we have $\gamma h \leq 1$; which implies[3] $\frac{\gamma h}{2}\epsilon \leq \frac{\gamma h}{1+\gamma h}\epsilon$. An easier but stronger sufficient condition is therefore $\delta \leq \frac{\gamma h}{2}\epsilon$. Therefore, shadowing in the unstable subspace requires $\frac{\ell L}{2}h^2 \leq \frac{\gamma h}{2}\epsilon \implies h \leq \frac{\gamma\epsilon}{\ell L}$. All in all, since we have to consider both manifold together, by subadditivity of the norm we actually need to require a radius of $\epsilon/2$. □

### B.2.3 *Under Strong Convexity GD is a Contraction*

The result below is used in the shadowing proof.

**Proposition B.2.4** (restated Prop. 2.4.3)**.** *Assume* ***(H1)****,* ***(H2)****. For all* $h \leq \frac{1}{L}$, $\Psi_h^{GD}$ *is uniformly contracting with* $\rho = 1 - h\mu$.

*Proof.* The general proof idea comes from Carrillo et al., 2006. For any $x_1, x_2 \in \mathbb{R}^d$ we want to compute

$$\|\Psi_h^{\mathrm{GD}}(x_1) - \Psi_h^{\mathrm{GD}}(x_2)\| = \|x_1 - x_2 - h(\nabla f(x_1) - \nabla f(x_2))\|.$$

Let $w : [0,1] \to \mathbb{R}^d$ be the straight line which connects $x_2$ to $x_1$, that is $w(r) = (1-r)x_2 + rx_1$. It is clear that $w'(r) = x_1 - x_2$ and, by the fundamental theorem of calculus (FTC),

$$\nabla f(x_1) - \nabla f(x_2) = \int_0^1 \frac{d(\nabla f(w(r)))}{dr} dr = \left(\int_0^1 \nabla^2 f(w(r)) dr\right)(x_1 - x_2).$$

[3] $\frac{x}{1+x} \leq \frac{x}{2}$ for $0 \leq x \leq 1$.

*It is of chief importance to notice that, in general, we can only use the FTC if* $f \in \mathbb{C}^2(\mathbb{R}^d, \mathbb{R})$: requiring the objective to be just twice differentiable is not sufficient; indeed, if the integrand is not continuous, the integral will not in general be differentiable : as a result, the Hessian would be undefined at certain points. Next, notice that, since clearly $\bar{H} := \int_0^1 \nabla^2 f(w(r))dr$ satisfies $\mu I \leq \bar{H} \leq LI$, we get

$$\begin{aligned}\|\Psi_h^{\mathrm{GD}}(x_1) - \Psi_h^{\mathrm{GD}}(x_2)\| &= \|(I - h\bar{H})(x_1 - x_2)\| \\ &\leq \|I - h\bar{H}\|\|x_1 - x_2\| \leq (1 - h\mu)\|x_1 - x_2\|,\end{aligned}$$

where we used that $\bar{H}$ is symmetric (matrix norm is the biggest eigenvalue) and $h \leq \frac{1}{L}$. □

### B.2.4 *Heavy-ball Local Approximation Error*

We first need a lemma.

**Lemma B.2.1.** *Assume* ***(H1)****. Let* $(p, q)$ *be the solution to HB-ODE starting from* $p(0) = 0$ *and from any* $q \in \mathbb{R}^d$*, for all* $t \geq 0$*, we have*

$$\begin{aligned}\|p(t)\| &\leq \ell/\alpha, \\ \|\dot{p}(t)\| &\leq 2\ell, \\ \|\ddot{p}(t)\| &\leq 2(\alpha + L)\ell.\end{aligned}$$

*Proof.* Let $S(t) = e^{\alpha t}p(t)$, then $\dot{S} = \alpha e^{\alpha t}p(t) + e^{\alpha t}(-\alpha p(t) - \nabla f(p(t))) = -e^{\alpha t}\nabla f(p(t))$. Hence, since $p(0) = 0$, $e^{\alpha t}p(t) = \int_0^t e^{\alpha s}\nabla f(p(s))ds$. Therefore

$$\begin{aligned}\|p(t)\| = \left\|e^{-\alpha t}\int_0^t e^{\alpha s}\nabla f(p(s))ds\right\| &\leq e^{-\alpha t}\int_0^t e^{\alpha s}\|\nabla f(p(s))\|ds \\ &\leq e^{-\alpha t}\int_0^t e^{\alpha s}\ell ds = \frac{1 - e^{-\alpha t}}{\alpha}\ell \leq \frac{\ell}{\alpha}.\end{aligned}$$

Using this, we can bound the acceleration

$$\|\dot{p}(t)\| = \| - \alpha p(t) - \nabla f(p(t))\| \leq \alpha\|p(t)\| + \|\nabla f(p(t))\| \leq 2\ell$$

and the jerk

$$\|\ddot{p}(t)\| = \left\| -\alpha \dot{p}(t) - \frac{d}{dt} \nabla f(p(t)) \right\| \leq 2\alpha\ell + \|\nabla^2 f(p(t))\| \|\dot{p}(t)\| = 2(\alpha + L)\ell.$$

□

We proceed with the proof of the proposition.

**Proposition B.2.5** (restated Prop. 2.4.5)**.** *Assume **(H1)**. The discretized HB-ODE solution (starting with zero velocity) $(y_k)_{k=0}^{\infty}$ is a $\delta$-pseudo-orbit of $\Psi_{\alpha,h}^{HB}$ with $\delta = \ell(\alpha + 1 + L)h^2$:*

$$\|y_{k+1} - \Psi_{\alpha,h}^{HB}(y_k)\| \leq \delta, \quad \textit{for all } x \in \mathbb{R}^d.$$

*Proof.* Thanks to Thm. 2.4.1, since the solution $y = (p, q)$ of HB-ODE is a $\mathbb{C}^2$ curve, we can write $(p(kh+h), q(kh+h)) = y(kh+h) = \varphi_h^{\mathrm{GD}}(y(kh)) = y_{k+1}$ using Taylor's theorem with Lagrange's Remainder in Banach spaces (as in the proof of Thm. 2.4.2) around time $t = kh$:

$$\begin{aligned} p(kh+h) &= p(kh) + h\dot{p}(kh) + \mathcal{R}_p(2,h) \\ &= p(kh) + h(-\alpha p(kh) - \nabla f(q(kh))) + \mathcal{R}_p(2,h) \\ &= (1 - h\alpha)p_k - \nabla f(q_k) + \mathcal{R}_p(2,h) \end{aligned} \tag{196}$$

and

$$\begin{aligned} q(kh) &= q(hk+h) + h\dot{q}(hk+h) + \mathcal{R}_q(2,h) \\ &= q(hk+h) + hp(kh+h) + \mathcal{R}_q(2,h). \end{aligned} \tag{197}$$

where

$$\|\mathcal{R}_p(2,h)\| \leq \frac{h^2}{2} \sup_{0 \leq \lambda \leq 1} \|\ddot{p}(t + \lambda h)\| \overset{\text{Lemma B.2.1}}{\leq} h^2(\alpha + L)\ell$$

and

$$\begin{aligned} \|\mathcal{R}_q(2,h)\| &\leq \frac{h^2}{2} \sup_{0 \leq \lambda \leq 1} \|\ddot{q}(t + \lambda h)\| \\ &= \frac{h^2}{2} \sup_{0 \leq \lambda \leq 1} \|\dot{p}(t + \lambda h)\| \leq \| \overset{\text{Lemma B.2.1}}{\leq} = h^2\ell. \end{aligned}$$

Without the residuals, Eq. (196) and (197) are the exact equations of the integrator $\Psi^{\mathrm{HB}}_{\alpha,h}$ (HB-PS in thesis chapter). Hence, for all $y_k = (p_k, q_k)$ in the pseudo orbit (which we require to start at $p(0) = 0$), by the triangle inequality,

$$\|\Psi^{\mathrm{HB}}_{\alpha,h}(p_k, q_k) - \varphi^{\mathrm{HB}}_{\alpha,h}(p_k, q_k)\| \leq \|\mathcal{R}_q(2,h)\| + \|\mathcal{R}_p(2,h)\| \leq \ell(\alpha + 1 + L)h^2.$$

□

#### B.2.4.1 *Heavy-ball on a quadratic is linear hyperbolic*

We want to study hyperbolicity of the map $\Psi^{\mathrm{HB}}_{\alpha,h}$ in phase space $(v, z)$. This map was defined in HB-PS, we report it below:

$$\begin{cases} v_{k+1} = v_k + h(-\alpha v_k - \nabla f(z_k)) \\ z_{k+1} = z_k + h v_{k+1} \end{cases}, \tag{198}$$

**Theorem B.2.4.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be an $L$-smooth quadratic with Hessian $H$. Assume $H$ does not have the eigenvalue zero. If $h \leq \sqrt{\frac{2}{L}}$ and $\alpha$ is such that $(1 - h\alpha) =: \beta \in [0, 1)$, then $\Psi^{HB}_{\alpha,h}$ is linear hyperbolic in phase space.*

*Proof.* If $\nabla f(x) = H(z_k - z^*)$, by adding and subtracting $z^*$ from both sides in the second equation, we can write this as a linear system

$$\begin{pmatrix} z_{k+1} - z^* \\ v_{k+1} \end{pmatrix} = A \begin{pmatrix} z_k - z^* \\ v_k \end{pmatrix},$$

where

$$A = \begin{pmatrix} I - h^2 H & \beta h I \\ -hH & \beta I \end{pmatrix} \quad \text{and} \quad \beta = 1 - \alpha h.$$

We assume $H$ has no eigenvalue at zero, and we seek to know if $A$ has eigenvalues on the unit circle $\mathbb{S}^1$. If there are no such eigenvalues, then Heavy-ball is hyperbolic.
To find the eigenvalues of this matrix we consider solving the following eigenvalue problem: for some $\xi_v, \xi_x \in \mathbb{R}^d$ and $q \in \mathbb{R}$, we require that

$$A \begin{pmatrix} \xi_x \\ \xi_v \end{pmatrix} = q \begin{pmatrix} \xi_x \\ \xi_v \end{pmatrix}.$$

To start, notice that this implies $-hHy_k = (q-\beta)\xi_v$. Therefore, assuming $q \neq \beta$, the position and velocity part of the eigenvector are linked: $\xi_v = \frac{hH}{\beta - q} y_k$. Hence, we get $(I - h^2 H)y_k + \frac{h^2 \beta H}{\beta - q}\xi_x = q\xi_x$. Therefore, we need $((\beta - q)I - (\beta - q)h^2 H + h^2 \beta H - q(\beta - q))\xi_x = 0$. Hence, $\xi_x$ needs to be an eigenvector of $H$, relative to some eigenvalue $\lambda$ which satisfies

$$\begin{aligned} &\beta - q - (\beta - q)h^2\lambda + h^2\beta\lambda - q(\beta - q) = 0 \\ \implies &\beta - q - \cancel{\beta h^2 \lambda} + qh^2\lambda + \cancel{h^2\beta\lambda} - \beta q + q^2 = 0, \end{aligned}$$

which results in the simple quadratic equation

$$\boxed{q^2 - (\beta + 1 - h^2\lambda)q + \beta = 0.} \tag{199}$$

A similar equation was derived in Kulakova et al., 2018. This is the equation we have to study for hyperbolicity. We have to show that, under some assumptions on $h$, $\lambda$, $\beta$, the solution $p$ never has norm 1.

Case with complex roots. If the roots of Eq. (199) are complex conjugates, $p, \bar{p}$, then

$$q^2 - (\beta + 1 - h^2\lambda)q + \beta = (q - p)(q - \bar{p}).$$

Hence, $\beta = |p|^2$. Therefore, if $\beta \in [0, 1)$, so are the moduli of the complex roots.

Case with real roots. We consider the more general equation $x^2 + bx + c = 0$. It's easy to realize that, if the roots are real, then a necessary conditions for those to lie on the unit circle is $c = -1 \pm b$. Indeed,

$$\begin{aligned} |x_{sol1}| = 1 \text{ or } |x_{sol2}| = 1 &\iff |-b \pm \sqrt{b^2 - 4c}| = 2 \\ &\iff b^2 + (b^2 - 4c) \pm 2b\sqrt{b^2 - 4c} = 4 \\ &\iff b^2 - 2c - 2 = \pm b\sqrt{b^2 - 4c} \\ &\implies b^4 + 4c^2 + 4 - 4b^2 c - 4b^2 + 8c = b^2(b^2 - 4c) \\ &\iff c^2 + 2c + 1 - b^2 = 0 \\ &\iff c = \frac{-2 \pm \sqrt{4 - 4(1 - b^2)}}{2} \\ &\iff c = -1 \pm b. \end{aligned}$$

in our case, $b = -\beta - 1 + h^2\lambda$ and $c = 1$. Therefore, we have to check that the following conditions are *never* fulfilled.

1. $-1-\beta-1+h^2\lambda = \beta$. This implies $h^2 = \frac{\beta+2}{\lambda}$. If $\lambda < 0$, since $\beta \in [0,1)$, the equation is never true. Else, it is true if $h = \sqrt{\frac{\beta+2}{\lambda}}$. But, since $|\lambda| \leq L$, if we assume $h \leq \sqrt{\frac{2}{L}}$, the equation is never satisfied.

2. $-1-(-\beta-1+h^2\lambda) = \beta$. This is verified only if $\lambda = 0$, which cannot happen under our assumptions.

□

# C

## APPENDIX TO CHAPTER 3

*Truth is much too complicated to allow anything but approximations.*
– John von Neumann.

We provide here support for our analysis of Transformers in Section 3.2. Due to space constraints, certain proofs have been omitted. For a comprehensive discussion, we recommend referring to Orvieto et al., 2022b.

### C.1 PROOFS AND VALIDATIONS FOR SECTION 3.2

#### C.1.1 *Proof Theorem 3.2.1 (Neural Chains)*

*Proof.* We focus on the derivatives with respect to the parameters $w_1$ and $w_2$ since the argument can be repeated for any parameter $w_i$. Denote by $\mathbf{w}$ the vector $(w_1, w_2, \cdots, w_L)$. The derivatives of the chain loss are

$$|\nabla_{w^1}\mathcal{L}(\mathbf{w})| = |yw^L\cdots w^2x - (w^L)^2\cdots(w^2)2^2(w^1)x|$$

$$|\nabla^2_{w^1,w^2}\mathcal{L}(\mathbf{w})| = |yw^L\cdots w^3x - 2(w^L)^2\cdots(w^3)^2w^2w^1x|$$

and

$$\left|\nabla^2_{w^1,w^1}\mathcal{L}(\mathbf{w})\right| = \left|(w^L)^2...(w^2)^2x\right|$$

It can be seen that the gradient as well as the Hessian off-diagonals scale similarly in depth. Let us first consider these two. Clearly,

$$\begin{aligned}|\nabla_{w^1}\mathcal{L}(\mathbf{w})| &\leq |yw^L...w^2x| + |(w^L)^2...(w^2)^2w^1x| \\ &\leq 2\cdot\max\{|yw^L...w^2x|, |(w^L)^2...(w^2)^2w^1x|\}\end{aligned}$$

For the first term in the max, we note that $\ln\left(\prod_{k=2}^L |w^k|\right) = \sum_{k=2}^L \ln(|w^k|)$ and since $|w^k| \sim \mathcal{U}[0,\tau]$, we have $\mathbb{E}\left[\ln(|w^k|)\right] = \ln(\tau) - 1$. Thus, the strong law of large numbers yields that with probability one

$$\prod_{k=2}^{L} |w^k| \to \exp\left((L-1)\left(\ln(\tau)-1\right)\right) \tag{200}$$

For the second term, we have $\ln\left(w^1 \prod_{k=2}^{L}(w^k)^2\right) = \ln(w^1) + \sum_{k=2}^{L} \ln((w^k)^2)$ and $\mathbb{E}\left[\ln((w^k)^2)\right] = 2\left(\ln(\tau)-1\right)$. Thus, the strong law of large numbers yields that with probability one $\ln\left(w^1 \prod_{k=2}^{L}(w^k)^2\right) \to (2L-1)\left(\ln(\tau)-1\right)$. Hence

$$w^1 \prod_{k=2}^{L}(w^k)^2 \to \exp\left((2L-1)\left(\ln(\tau)-1\right)\right). \tag{201}$$

For large $L$, Eq.(200) clearly dominates Eq.(201), which proves the first statement. The second statement follows similarly to Eq. (201).

□

### C.1.2 *Proof of Theorem 3.2.3 (General MLP)*

We consider the loss

$$\mathcal{L}_{\mathbf{x},\mathbf{y}}(\mathbf{W}) = \frac{1}{2}\|\mathbf{y} - \mathbf{B}\mathbf{D}^L \mathbf{W}_\phi^{L:1}\mathbf{A}\mathbf{x}\|^2.$$

#### C.1.2.1 *Gradient Analysis*

As in Allen-Zhu et al., 2019, by noting $\tilde{\mathbf{B}} := \mathbf{B}\mathbf{D}^L$ and $\mathbf{z} = \mathbf{A}\mathbf{x}$ we get

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \mathbf{W}^k} &= \mathbf{W}_\phi^{k+1:L}\tilde{\mathbf{B}}^\top[\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:1}\mathbf{z} - \mathbf{y}]\mathbf{z}^\top \mathbf{W}^{1:k-1} \\ &= \underbrace{\mathbf{W}_\phi^{k+1:L}\tilde{\mathbf{B}}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:1}\mathbf{z}\mathbf{z}^\top\mathbf{W}^{1:k-1}}_{\partial\mathcal{L}_k^1} - \underbrace{\mathbf{W}_\phi^{k+1:L}\tilde{\mathbf{B}}^\top\mathbf{y}\mathbf{z}^\top\mathbf{W}^{1:k-1}}_{\partial\mathcal{L}_k^2},\end{aligned}$$

with $\mathbf{W}_\phi^{k+1:L} := \left(\mathbf{W}_\phi^{L:k+1}\right)^\top$. By the triangle inequality, we have

$$\left\|\frac{\partial \mathcal{L}}{\partial \mathbf{W}^k}\right\|_F \leq \|\partial\mathcal{L}_k^1\|_F + \|\partial\mathcal{L}_k^2\|_F,$$

therefore we can bound each term individually. Let $\mathbb{E}^p[\cdot]$ be the $p$-th power of $\mathbb{E}[\cdot]$.

**Proposition C.1.1** (Bounding gradients with forward passes)**.** *We have*

$$\mathbb{E}\|\partial \mathcal{L}_k^1\|_F \leq \mathbb{E}^{1/2}\left[\|\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:k+1}\|_2^2\right]\mathbb{E}^{1/4}\left[\|\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:1}\mathbf{z}\|_2^4\right]\mathbb{E}^{1/4}\left[\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\|_2^4\right];$$
$$\mathbb{E}\|\partial \mathcal{L}_k^2\|_F \leq \mathbb{E}\left[\|\mathbf{y}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:k+1}\|_2^2\right]\mathbb{E}\left[\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\|_2^2\right].$$

*Proof.* The bounds follow from submultiplicativity of Frobenius norm and Cauchy-Schwarz inequality — applied possibly twice.

$$\begin{aligned}
\mathbb{E}\|\partial \mathcal{L}_k^1\|_F &= \mathbb{E}\|\mathbf{W}_\phi^{k+1:L}\tilde{\mathbf{B}}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:1}\mathbf{z}\mathbf{z}^\top\mathbf{W}_\phi^{1:k-1}\|_F \\
&\leq \mathbb{E}\left[\|\mathbf{W}_\phi^{k+1:L}\tilde{\mathbf{B}}^\top\|_F\|\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:1}\mathbf{z}\|_F\|\mathbf{z}^\top\mathbf{W}_\phi^{1:k-1}\|_F\right] \\
&= \mathbb{E}\left[\|\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:k+1}\|_F\|\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:1}\mathbf{z}\|_F\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\|_F\right] \\
&\leq \mathbb{E}^{1/2}\left[\|\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:k+1}\|_F^2\right]\mathbb{E}^{1/2}\left[\|\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:1}\mathbf{z}\|_F^2\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\|_F^2\right] \\
&\leq \mathbb{E}^{1/2}\left[\|\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:k+1}\|_F^2\right]\mathbb{E}^{1/4}\left[\|\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:1}\mathbf{z}\|_F^4\right]\mathbb{E}^{1/4}\left[\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\|_F^4\right].
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}\|\partial \mathcal{L}_k^2\|_F &= \mathbb{E}\|\mathbf{W}_\phi^{k+1:L}\tilde{\mathbf{B}}^\top\mathbf{y}\mathbf{z}^\top\mathbf{W}_\phi^{1:k-1}\|_F \\
&\leq \mathbb{E}\left[\|\mathbf{W}_\phi^{k+1:L}\tilde{\mathbf{B}}^\top\mathbf{y}\|_F\|\mathbf{z}^\top\mathbf{W}_\phi^{1:k-1}\|_F\right] \\
&\leq \mathbb{E}^{1/2}\left[\|\mathbf{W}_\phi^{k+1:L}\tilde{\mathbf{B}}^\top\mathbf{y}\|_F^2\right]\mathbb{E}^{1/2}\left[\|\mathbf{z}^\top\mathbf{W}_\phi^{1:k-1}\|_F^2\right] \\
&= \mathbb{E}^{1/2}\left[\|\mathbf{y}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:k+1}\|_F^2\right]\mathbb{E}^{1/2}\left[\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\|_F^2\right].
\end{aligned}$$

We conclude by noting that the Frobenius norm is the 2-norm for vectors. □

**Proposition C.1.2** (Bounding gradients with forward passes, wide net)**.** *As $d \to \infty$,*

$$\mathbb{E}\|\partial \mathcal{L}_k^1\|_F \lesssim (p\sigma^2 d)^{\frac{2L-1}{2}},$$
$$\mathbb{E}\|\partial \mathcal{L}_k^2\|_F \lesssim (p\sigma^2 d)^{\frac{L-1}{2}},$$

*where "$\lesssim$" denotes "asymptotically less of equal than a multiple of" (same as $\mathcal{O}$). Hence, we have*

$$\mathbb{E}\left\|\frac{\partial \mathcal{L}}{\partial \mathbf{W}^k}\right\|_F \lesssim (p\sigma^2 d)^{\frac{L-1}{2}} \qquad \text{if} \quad (p\sigma^2 d) \leq 1 \quad \textit{(vanishing-stable regime).}$$

$$\mathbb{E}\left\|\frac{\partial \mathcal{L}}{\partial \mathbf{W}^k}\right\|_F \lesssim (p\sigma^2 d)^{\frac{2L-1}{2}} \qquad \text{if} \quad (p\sigma^2 d) \geq 1 \quad \textit{(exploding regime).}$$

*Proof.* Please check Orvieto et al., 2022b. □

#### C.1.2.2 *Hessian Analysis*

The Hessian of a linear DNN can be split into two block matrices, where each block has a Kronecker product structure. We can apply the product rule to the gradient and consider $\mathbf{W}^\ell$ and $\mathbf{W}^{\ell\top}$ ($\ell > k$) as two distinct matrices: the block $(k, \ell)$ of the Hessian matrix is:

$$\underbrace{\frac{\partial^2 \mathcal{L}}{\partial \mathbf{W}^k \partial \mathbf{W}^\ell}}_{\mathbf{H}_1^{k\ell}} + \underbrace{\frac{\partial^2 \mathcal{L}}{\partial \mathbf{W}^k \partial \mathbf{W}^{\ell\top}}}_{\mathbf{H}_2^{k\ell}} \underbrace{\frac{\partial (\mathbf{W}^{\ell\top})}{\partial \mathbf{W}^\ell}}_{\mathbb{T}},$$

where $\mathbb{T}$ is the matrix transpose tensor. Recall that

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}^k} = \mathbf{W}_\phi^{k+1:L} \tilde{\mathbf{B}}^\top [\tilde{\mathbf{B}} \mathbf{W}_\phi^{L:1} \mathbf{z} - \mathbf{y}] \mathbf{z}^\top \mathbf{W}^{1:k-1}.$$

By using the simple rule $\frac{\partial \mathbf{EWF}}{\partial \mathbf{W}} = \mathbf{F}^\top \otimes \mathbf{E}$, we get

$$\mathbf{H}_1^{k\ell} = \mathbf{W}_\phi^{k-1:1} \mathbf{z}\mathbf{z}^\top \mathbf{W}_\phi^{1:\ell-1} \otimes \mathbf{W}_\phi^{k+1:L} \tilde{\mathbf{B}}^\top \tilde{\mathbf{B}} \mathbf{W}_\phi^{L:\ell+1}$$

and

$$\begin{aligned}
\mathbf{H}_2^{k\ell} &= \mathbf{W}_\phi^{k-1:1} \mathbf{z} \left( \mathbf{z}^\top \mathbf{W}_\phi^{1:L} \tilde{\mathbf{B}}^\top - \mathbf{y}^\top \right) \tilde{\mathbf{B}} \mathbf{W}_\phi^{L:\ell+1} \otimes \mathbf{W}_\phi^{k+1:\ell-1} \\
&= \underbrace{\mathbf{W}_\phi^{k-1:1} \mathbf{z}\mathbf{z}^\top \mathbf{W}_\phi^{1:L} \tilde{\mathbf{B}}^\top \tilde{\mathbf{B}} \mathbf{W}_\phi^{L:\ell+1} \otimes \mathbf{W}_\phi^{k+1:\ell-1}}_{\mathbf{H}_{21}^{k\ell}} \\
&\quad - \underbrace{\mathbf{W}_\phi^{k-1:1} \mathbf{z}\mathbf{y}^\top \tilde{\mathbf{B}} \mathbf{W}_\phi^{L:\ell+1} \otimes \mathbf{W}_\phi^{k+1:\ell-1}}_{\mathbf{H}_{22}^{k\ell}},
\end{aligned}$$

Note that if instead $k = \ell$, $\mathbf{H}^{kk} = \mathbf{H}_1^{kk}$.

**Proposition C.1.3** (Bounding the Hessian with forward passes)**.** *It is possible to bound the Hessian with statistics only on the forward pass.*

*Proof.* We simply apply the Cauchy-Schwarz inequality twice for each term, using also the Frobenius norm formula for the Kronecker product and norm submultiplicativity.

$$
\begin{aligned}
&\mathbb{E}\|\mathbf{H}_1^{k\ell}\|_F \\
&= \mathbb{E}\left\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\mathbf{z}^\top\mathbf{W}_\phi^{1:\ell-1} \otimes \mathbf{W}_\phi^{k+1:L}\tilde{\mathbf{B}}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:\ell+1}\right\|_F \\
&\leq \mathbb{E}^{1/2}\left\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\mathbf{z}^\top\mathbf{W}_\phi^{1:\ell-1}\right\|_F^2 \mathbb{E}^{1/2}\left\|\mathbf{W}_\phi^{k+1:L}\tilde{\mathbf{B}}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:\ell+1}\right\|_F^2 \\
&\leq \mathbb{E}^{1/4}\left\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\right\|_F^4 \mathbb{E}^{1/4}\left\|\mathbf{W}_\phi^{\ell-1:1}\mathbf{z}\right\|_F^4 \mathbb{E}^{1/4}\left\|\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:\ell+1}\right\|_F^4 \mathbb{E}^{1/4}\left\|\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:k+1}\right\|_F^4.
\end{aligned}
$$

Moreover,

$$
\begin{aligned}
&\mathbb{E}\|\mathbf{H}_{22}^{k\ell}\|_F \\
&\leq \mathbb{E}^{1/2}\left\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\mathbf{y}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:\ell+1}\right\|_F^2 \mathbb{E}^{1/2}\left\|\mathbf{W}_\phi^{k+1:\ell-1}\right\|_F^2 \\
&\leq \mathbb{E}^{1/4}\left\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\right\|_F^4 \mathbb{E}^{1/4}\left\|\mathbf{y}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:\ell+1}\right\|_F^4 \mathbb{E}^{1/2}\left\|\mathbf{W}_\phi^{\ell-1:k+1}\right\|_F^2.
\end{aligned}
$$

Finally,

$$
\begin{aligned}
&\mathbb{E}\|\mathbf{H}_{21}^{k\ell}\|_F \\
&\leq \mathbb{E}^{1/2}\left\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\mathbf{z}^\top\mathbf{W}_\phi^{1:L}\tilde{\mathbf{B}}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:\ell+1}\right\|_F^2 \mathbb{E}^{1/2}\left\|\mathbf{W}_\phi^{k+1:\ell-1}\right\|_F^2 \\
&\leq \mathbb{E}^{1/4}\left\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\mathbf{z}^\top\mathbf{W}_\phi^{1:\ell}\mathbf{W}_\phi^{\ell+1:L}\tilde{\mathbf{B}}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:\ell+1}\right\|_F^2 \mathbb{E}^{1/4}\left\|\mathbf{W}_\phi^{\ell-1:k+1}\right\|_F^2.
\end{aligned}
$$

Note that the last term is not simplified completely, but unfortunately a simple iterated Cauchy-Schwarz splitting would lead to quantities with high exponents (eighth moment). Hence, we need to take a more complex approach. First, we split between terms which do not share weights.

$$
\mathbb{E}\left[\left\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\mathbf{z}^\top\mathbf{W}_\phi^{1:\ell}\right\|_F^2 \left\|\mathbf{W}_\phi^{\ell+1:L}\tilde{\mathbf{B}}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:\ell+1}\right\|_F^2\right].
$$

Using the law of total expectation, the last expression becomes

$$
\mathbb{E}\left[\ \mathbb{E}\left[\ \left\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\mathbf{z}^\top\mathbf{W}_\phi^{1:\ell}\right\|_F \left\|\mathbf{W}_\phi^{\ell+1:L}\tilde{\mathbf{B}}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:\ell+1}\right\|_F^2 \ \middle|\ \mathcal{F}_\ell\ \right]\ \right],
$$

where $\mathcal{F}_\ell$ is the information until layer $\ell$. It is easy to realize (check Orvieto et al., 2022b) that fixing the preactivation a separate integration of the second term in the product. In particular, the expression becomes

$$\mathbb{E}\left[\left\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\mathbf{z}^\top\mathbf{W}_\phi^{1:\ell}\right\|_F\right]\cdot\mathbb{E}\left[\left\|\mathbf{W}_\phi^{\ell+1:L}\tilde{\mathbf{B}}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:\ell+1}\right\|_F^2\;\middle|\;\mathcal{F}_\ell\right].$$

As usual, we drop the filtration notation and plug this back into the expression for $\mathbb{E}\|\mathbf{H}_{21}^{k\ell}\|_F$:

$$\begin{aligned}
&\mathbb{E}\|\mathbf{H}_{21}^{k\ell}\|_F\\
&\leq \mathbb{E}^{1/2}\left\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\mathbf{z}^\top\mathbf{W}_\phi^{1:\ell}\right\|_F^2\mathbb{E}^{1/2}\left\|\mathbf{W}_\phi^{\ell+1:L}\tilde{\mathbf{B}}^\top\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:\ell+1}\right\|_F^2\mathbb{E}^{1/2}\left\|\mathbf{W}_\phi^{\ell-1:k+1}\right\|_F^2\\
&\leq \mathbb{E}^{1/4}\left\|\mathbf{W}_\phi^{k-1:1}\mathbf{z}\right\|_F^4\mathbb{E}^{1/4}\left\|\mathbf{W}_\phi^{\ell:1}\mathbf{z}\right\|_F^4+\mathbb{E}^{1/2}\left\|\tilde{\mathbf{B}}\mathbf{W}_\phi^{L:\ell+1}\right\|_F^4\mathbb{E}^{1/2}\left\|\mathbf{W}_\phi^{\ell-1:k+1}\right\|_F^2.
\end{aligned}$$

□

**Proposition C.1.4** (Bounding Hessians with forward passes, wide net)**.** *As $d\to\infty$, for $k\neq\ell$*

$$\begin{aligned}
\mathbb{E}\|\mathbf{H}^{k\ell}\|_F &\lesssim (p\sigma^2 d)^{\frac{L-2}{2}}+(p\sigma^2 d)^{L-1},\\
\mathbb{E}\|\mathbf{H}^{kk}\|_F &\lesssim (p\sigma^2 d)^{L-1},
\end{aligned}$$

*where "$\lesssim$" denotes "asymptotically less of equal than a multiple of" (same as $\mathcal{O}$). Hence, we have*

$$\begin{aligned}
&\mathbb{E}\|\mathbf{H}^{k\ell}\|_F\lesssim(p\sigma^2 d)^{\frac{L-2}{2}},\quad \mathbb{E}\|\mathbf{H}^{kk}\|_F\lesssim(p\sigma^2 d)^{L-1} \qquad \text{if}\quad (p\sigma^2 d)\leq 1.\\
&\mathbb{E}\|\mathbf{H}^{k\ell}\|_F\lesssim(p\sigma^2 d)^{L-1},\quad \mathbb{E}\|\mathbf{H}^{kk}\|_F\lesssim(p\sigma^2 d)^{L-1} \qquad \text{if}\quad (p\sigma^2 d)\geq 1.
\end{aligned}$$

*Proof.* Follows from the triangle inequality. The only element left to bound is the transpose tensor $\mathbb{T}$, which however has only polynomial Frobenius norm in $d$ and $L$. □

### C.1.3 *Behaviour of RMSprop on Neural Chains*

We empirically study the behavior of RMSprop on the chain loss with one data point $(x,y)=(1,1)$:

$$\mathcal{L}_{\text{chain}}(\mathbf{w})=\frac{1}{2}(1-w_L w_{L-1}\cdots w_1)^2.$$

While this cost function is very simple, it showcases the adaptiveness of RMSprop in an extremely clean and concrete way and gave us the opportunity for an in-depth analysis in Chapter 3. In this appendix, we consider $L = 10$ and initialize each weight to $w_i(0) \sim \mathcal{U}[-0.2, 0.2]$, to induce vanishing gradients (order $10^{-8}$) and curvature (order order $10^{-7}$), as it can be seen from Figure 63 and is predicted by Theorem 3.2.1 and Corollary 3.2.1. This initialization is close to $\mathbf{w} = \mathbf{0}$, which is clearly a saddle point because all partial derivatives vanish, but there exist directions of both increases and decreases: increasing all $w_i$ simultaneously to $\epsilon > 0$ makes the loss decrease, while increasing half (i.e. five) of them to $\epsilon$ and decreasing the other half to $-\epsilon$ makes the loss increase.
We show results comparing gradient descent (GD) and RMSprop ((Tieleman and Hinton, 2012) for 3 different noise injection levels. Results are shown in Figure 63: while perturbed GD is slow to escape the saddle for any noise injection level and any stepsize, RMSprop is able to adapt to curvature and quickly escapes the saddle. This result validates the claim presented in Prop. 3.2.4 of this thesis. In addition, Figure 64 shows that, if the initialization is such that the gradients at initialization are bigger, i.e. $w_i(0) \sim \mathcal{U}[-1, 1]$, then the performance of perturbed gradient descent get can get closer to the one of RMSprop.

**Notation.** We denote by $\mathbf{v}(t)$ the exponential moving average (with parameter $\beta_2 = 0.9$) of the squared gradients at iteration $t$ and by $\Lambda(t)$ the maximum Hessian eigenvalue at iteration $t$. Recall that RMSprop updates the weights as

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \frac{\eta}{\sqrt{\mathbf{v}(t)}} \nabla L(\mathbf{w}(t)),$$
$$\mathbf{v}(t+1) = \beta_2 \mathbf{v}(t) + (1-\beta_2) \nabla L(\mathbf{w}(t))^{\odot 2},$$

where $L(\mathbf{w}(t))^{\odot 2}$ is the elementwise square of $L(\mathbf{w}(t))$.

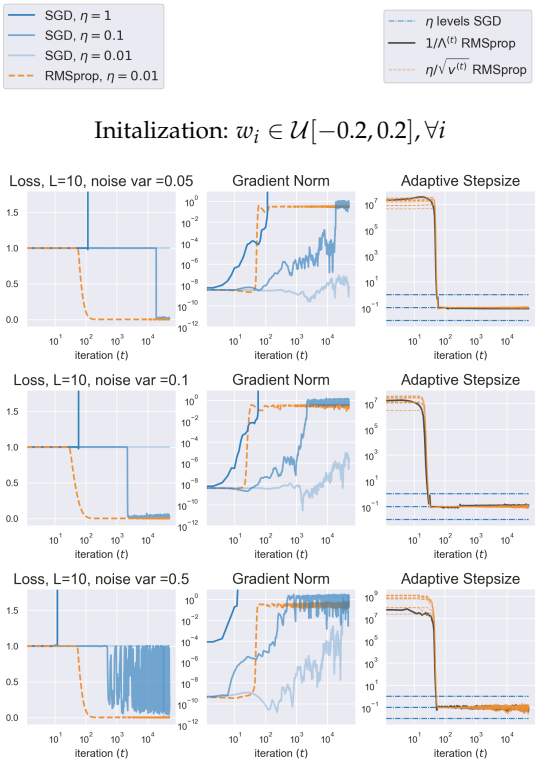


**Figure 63:** Optimization of a ten-dimensional chain, $w_i(0) \sim \mathcal{U}[-0.2, 0.2]$. Injected is an isotropic Gaussian noise of standard deviations $0.05, 0.1, 0.5$. While moderate noise helps GD (blue), no choice of stepsize is able to provide a performance comparable to RMSprop, which escapes after less than 100 iterations. Notably, the effective RMSprop stepsize matches the inverse curvature and is robust to noise. For additional runs, please check the appendix of Orvieto et al., 2022b.

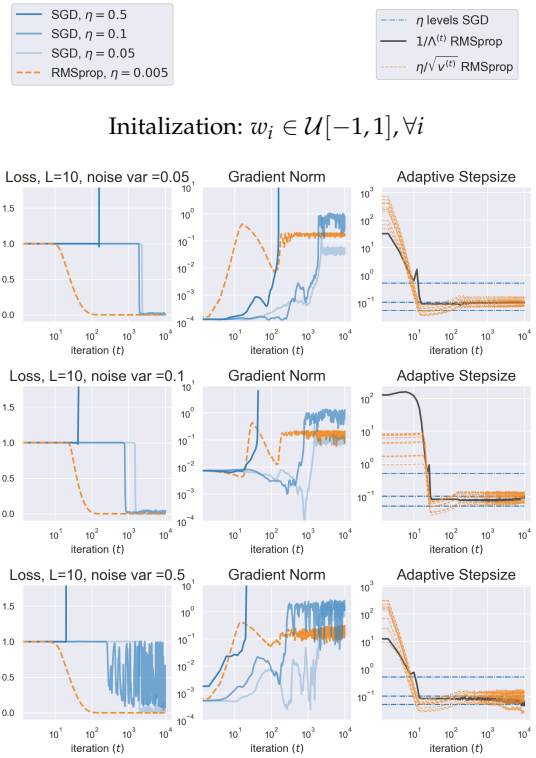


**Figure 64:** Here we instead consider $w_i(0) \sim \mathcal{U}[-1, 1]$, which induces less curvature-gradient vanishing compared to Fig. 63. If the initial gradient norm is high, Perturbed gradient descent gets now closer in performance to RMSprop. However, the gradient/Hessian norm at initialization has a considerable variance, which makes the performance less predictable compared to Fig. 63. This fact also makes the curvature adaptation feature of RMSprop less apparent. For additional runs, please check the appendix of Orvieto et al., 2022b.

### C.1.4 *Numerical Verification for Proposition 3.2.5*

In Figure 65 we provide a numerical verification for Proposition 3.2.5. We do not present the proof for this proposition in this thesis since it would require several pages of notation and calculations. The interested reader can find the complete discussion in the appendix of Orvieto et al., 2022b. We recall the statement:
Let $\mathbf{z} = \mathbf{A}\mathbf{x}$. Let $\kappa = \mu_4/\sigma_4$ be the kurtosis (fourth standardized moment) of the initialization distribution for each weight entry (see Assumption 3.2.1). Let $p = 1$ in the linear case and $p = 1/2$ in the ReLU case. Then we have

$$\mathbb{E}\|\mathbf{W}_\phi^{k:1}\mathbf{z}\|_2^2 = (d\sigma^2 p)^k \mathbb{E}\|\mathbf{z}\|_2^2.$$

$$\begin{pmatrix} \mathbb{E}\|\mathbf{W}_\phi^{k:1}\mathbf{z}\|_2^4 \\ \mathbb{E}\|\mathbf{W}_\phi^{k:1}\mathbf{z}\|_4^4 \end{pmatrix} = \left(p^2 d\sigma^4\right)^k \mathbf{Q}^k \begin{pmatrix} \mathbb{E}\|\mathbf{z}\|_2^4 \\ \mathbb{E}\|\mathbf{z}\|_4^4 \end{pmatrix},$$

$$\text{with} \quad \mathbf{Q} := \begin{pmatrix} d+2 & \frac{\kappa-3+(1-p)(d+2)}{p} \\ 3 & \frac{\kappa-3p}{p} \end{pmatrix}.$$

In Figure 65 we provide an empirical verification of the formulas above.

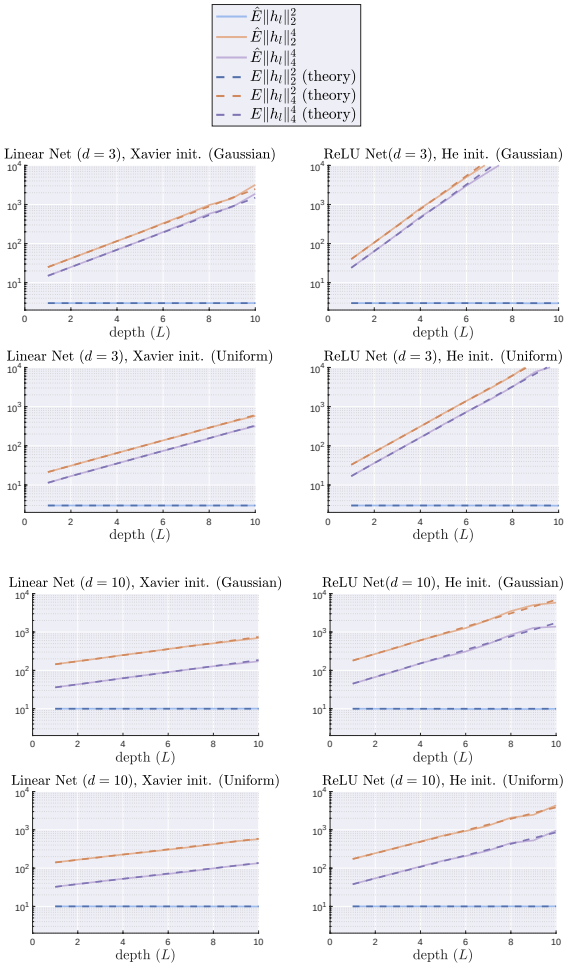


**Figure 65:** Numerical validation of Proposition 3.2.5 using the classical stabilizing initializations (Glorot and Bengio, 2010; He et al., 2015). The variance of the weights is set to $1/\sqrt{d}$ for linear nets and $2/\sqrt{d}$ for ReLU nets (here, we used $d = 3, 10$). We consider a random Gaussian input and initialization of the weights with either Gaussian or uniform distribution. The theory matches the experiment (empirical mean denoted as $\hat{\mathbb{E}}$ — $1e5$ runs for $d = 10$, $1e7$ runs for $d = 3$). The results for the two initializations are similar, yet Gaussian case explodes a bit faster due to the effect of the kurtosis, which for the Gaussian is 3 while for the uniform is $3 - \frac{6}{5}$. The formula in Prop. 3.2.5 also perfectly predicts this tiny shift in the population quantities, *confirming the correctness of our calculations*.

## C.2 PROOFS AND VALIDATIONS FOR SECTION 3.3

We provide here support for our analysis of Transformers in Section 3.3. Due to space constraints, certain proofs have been omitted. For a comprehensive discussion, we recommend referring to (Noci et al., 2022).

### C.2.1 *Attention is Uniform at Initialization*

Here, we empirically test the accuracy and limitations of the uniform-attention assumption.
For the empirical verification of Assumption 3.3.1 in the forward pass analysis, we plot the density of the norm of the representations for only-encoder Transformers of increasing depth. The results are shown in Fig 66. Note that when the standard deviation of the input is set to $1/\sqrt{d}$, then the uniform-attention assumption provide an excellent approximation to the common Xavier-initialization. On the contrary, we observe a deviation when the standard deviation of the input is increased. Also, note how as the depth increases, the distribution becomes more heavy-tailed. This heavy-tailedness was recently formally shown for standard MLPs with and without ReLU activation (Noci et al., 2021; Zavatone-Veth and Pehlevan, 2021).
For the verification of the assumption in the backward pass, we additionally show in Fig. 67 how the norm of the gradients w.r.t queries and keys depends on the hidden dimension, the sequence length, the input correlation and the input variance. *Ground-truth* gradients are calculated with automatic differentiation, and they are compared with our theoretical results based on Assumption 3.3.1. As shown in Fig.67, our theoretical predictions show a very good agreement with the true gradients. Again, we notice that the smaller the values of the input standard deviation the tighter the agreement of the theory with the simulations. Intuitively, a higher input variance causes the argument of the softmax to have a large range of values. This in turn causes a deviation from the uniform distribution (i.e. maximum entropy), towards the distribution of minimum entropy (a Delta Dirac, corresponding to attending to only one token).

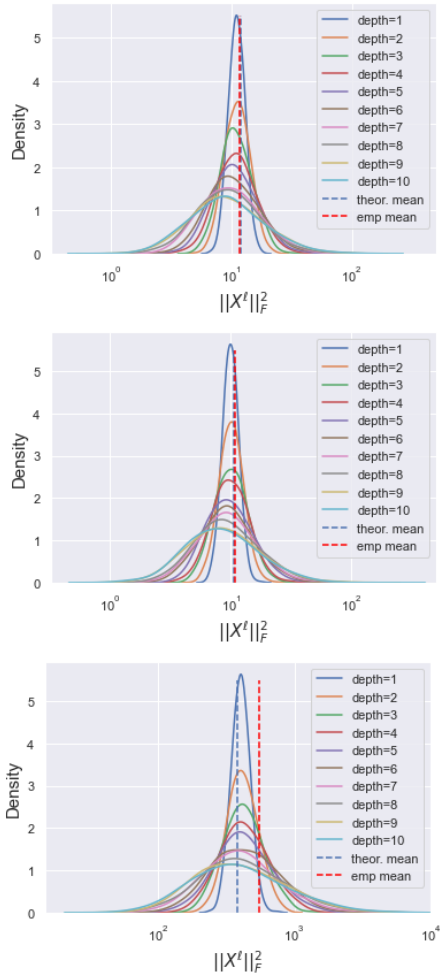


**Figure 66:** *Density plots for* $\|\mathbf{X}^\ell\|_F^2$ *for Transformers of depths* $L$ *from* 1 *to* 10. *The input* $\mathbf{X}$ *contains i.i.d Gaussian entries, simulating an embedding layer. We set* $d := d_v = d_q = 30$. *The empirical mean at* $L = 10$ *is highlighted in a vertical dashed red line, while the theoretical mean (Lemma 3.3.3) is a dashed blue line. The densities are estimated by sampling 1000 times the weights of the network.* ***(Top)****: we adopt the uniform-attention. The standard deviation of the input is set to* $1/\sqrt{d}$. ***(Center)****: Same, but removing the uniform-attention assumption.* ***(Bottom)****: We remove the uniform-attention assumption, and set the standard deviation of the input to* 1.

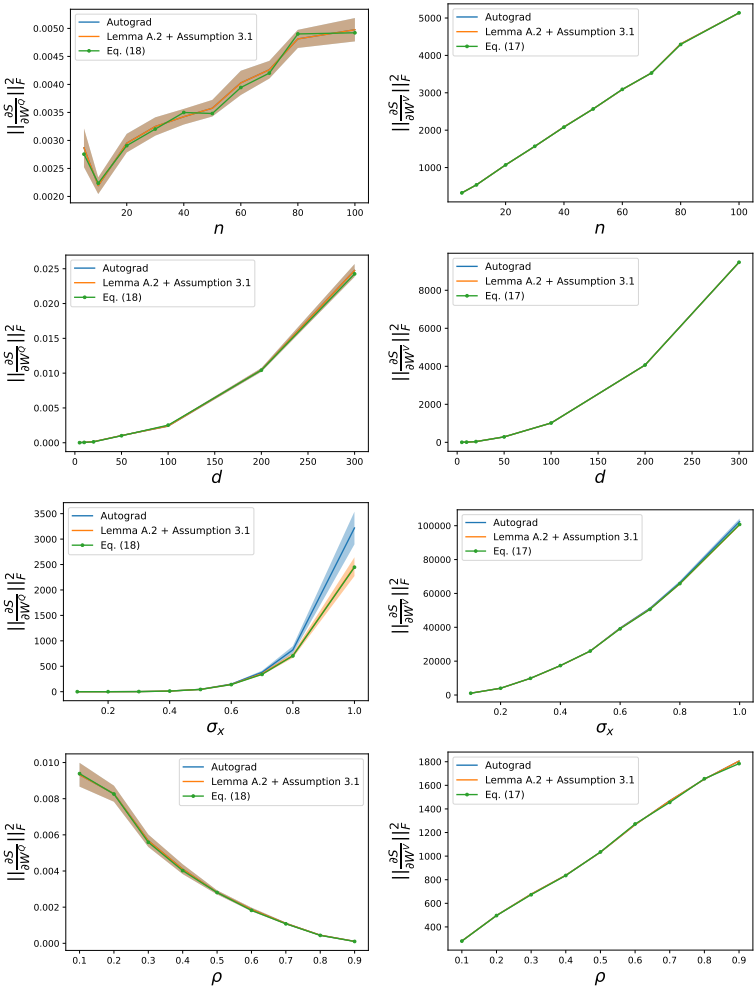


**Figure 67:** *Empirical comparison of our theoretical findings. We sample, as aforementioned, the tokens according to a zero-mean Gaussian distribution, while varying the hidden dimension, sequence length, input correlation and input variance. Results are averaged over 20 runs.*

### C.2.2 *Verification of the Gradient Analysis*

Finally, in Figures 68 we show the dependence of the norm of the gradients for the keys and values based on the parameters of the architecture and the task-specific parameters. The bottom panel illustrates the true dependence and the top panel the one expected by the theory based on our assumptions. In short, the main takeaways are the following.

- As the correlation between the tokens increases ($x$-axis in the global plot), the norm of the gradients of the queries quickly diminishes compared to the one of the values.
- The dependence on the variance of the input $\sigma_x^2$ is different ($y$-axis in the global plot), being linear for the values and cubic for the queries. This highlights the importance of a stabilized forward pass and provides another explanation regarding the successful use of layer norm in Transformers.
- The dependence on $n$ ($x$-axis in each subplot) and $d$ ($y$-axis in each subplot) is more complicated, also being a function of the correlation $\rho$ (compare the first column where $\rho = 0$ to the rest).

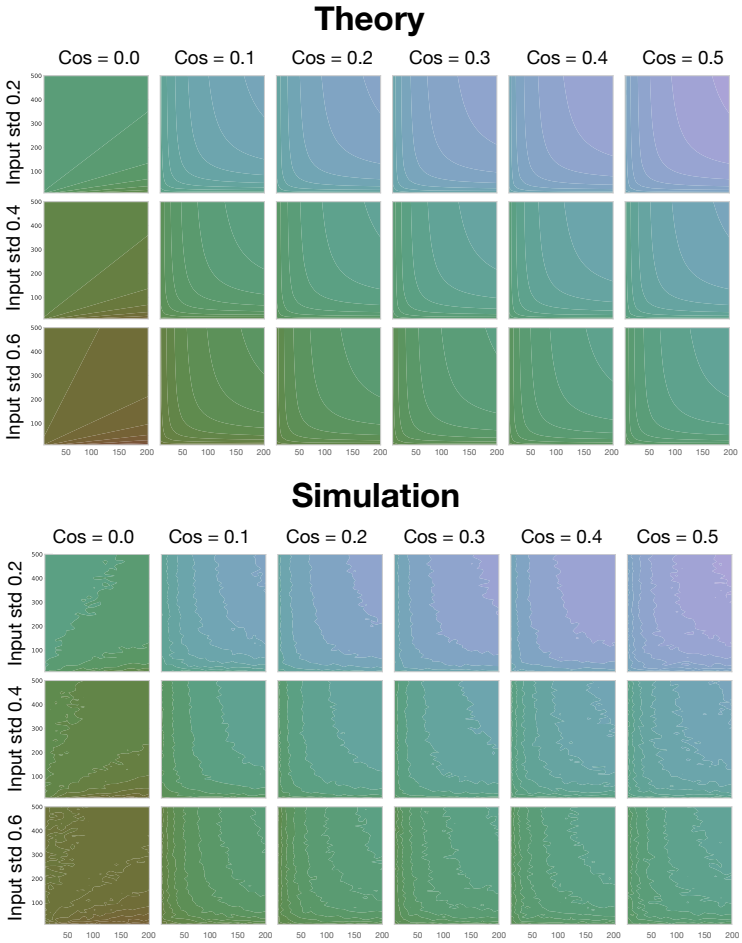


**Figure 68:** *Validation of Equations (84) and (85). Plotted is the Log ratio of the norm of the gradients for the queries compared to those of the values for varying values of embedding dimension, sequence length, cosine of the tokens angle and standard deviation.*

### C.2.3 *Proof of Theorem 3.3.1 (Vanishing Gradients)*

Before starting the proof, it is interesting to note that, even though the gradients of queries and keys vanish in the rank collapse regime (i.e. $\left\|\mathbf{X}^\top\mathbf{X} - n\bar{\mathbf{x}}\bar{\mathbf{x}}^\top\right\| = 0$), the gradient with respect to the values and the input does not (see Theorem 3.3.1). From this simple remark, we can conclude that, even in the rank collapse regime, information still propagates in the backward pass. In Section 3.3.3, we show that even if gradients effectively propagate, the phenomenon studied in this theorem still greatly affects training.

*Proof.* By using the chain rule and the fact that for two matrixes $\mathbf{A}, \mathbf{B}$ we have that $\|\mathbf{AB}\|_F^2 \leq \|\mathbf{A}\|_F^2 \|\mathbf{B}\|_F^2$, we can upper bound the gradient as:

$$
\begin{aligned}
&\left\| \frac{\partial \mathcal{L}}{\partial \mathbf{W}^{Q,\ell}} \right\|_F^2 \\
&\leq \prod_{i=\ell+1}^{L-1} \left\| \frac{\partial \mathbf{X}^{i+1}}{\partial \mathbf{X}^i} \right\|_F^2 \left\| \frac{\partial \mathcal{L}}{\partial \mathbf{X}^L} \right\|_F^2 \left\| \frac{\partial \mathbf{X}^{\ell+1}}{\partial \mathbf{W}^{Q,\ell}} \right\|_F^2 \\
&\leq \prod_{i=\ell+1}^{L-1} \left\| \frac{\partial \mathbf{X}^{i+1}}{\partial \mathbf{X}^i} \right\|_F^2 \left\| \frac{\partial \mathcal{L}}{\partial \mathbf{X}^L} \right\|_F^2 \left\| \frac{\partial \mathbf{X}^{\ell+1}}{\partial \mathbf{Z}^\ell} \right\|_F^2 \left\| \frac{\partial \mathbf{Z}^\ell}{\partial \mathbf{W}^{Q,\ell}} \right\|_F^2 \\
&\leq \prod_{i=\ell+1}^{L-1} \left\| \frac{\partial \mathbf{X}^{i+1}}{\partial \mathbf{X}^i} \right\|_F^2 \left\| \frac{\partial \mathcal{L}}{\partial \mathbf{X}^L} \right\|_F^2 \left\| \frac{\partial \mathbf{X}^{\ell+1}}{\partial \mathbf{Z}^\ell} \right\|_F^2 \left( \left\| \frac{\partial \alpha_1 \mathbf{S}^\ell}{\partial \mathbf{W}^{Q,\ell}} \right\|_F^2 + \underbrace{\left\| \frac{\partial \mathbf{X}^\ell}{\partial \mathbf{W}^{Q,\ell}} \right\|_F^2}_{=0} \right),
\end{aligned}
$$

where we recall that $\mathbf{Z}^\ell = \alpha_1 \mathbf{S}^\ell + \mathbf{X}^\ell$ and in the last step we have used that $\mathbf{X}^\ell$ does not depend on $\mathbf{W}^{Q,\ell}$, hence the gradient vanishes. By taking expectation and using the tower property, we have that:

$$
\mathbb{E}\left\| \frac{\partial \mathcal{L}}{\partial \mathbf{W}^{Q,\ell}} \right\|_F^2 \leq \mathbb{E}\left[ \underbrace{\mathbb{E}\left[ \prod_{i=\ell+1}^{L-1} \left\| \frac{\partial \mathbf{X}^{i+1}}{\partial \mathbf{X}^i} \right\|_F^2 \left\| \frac{\partial \mathcal{L}}{\partial \mathbf{X}^L} \right\|_F^2 \left\| \frac{\partial \mathbf{X}^{\ell+1}}{\partial \mathbf{Z}^\ell} \right\|_F^2 \right]}_{=:G(\mathbf{X}^\ell)} \left\| \frac{\partial \alpha_1 \mathbf{S}^\ell}{\partial \mathbf{W}^{Q,\ell}} \right\|_F^2 \right],
$$

where the expectations are taken with respect to $\mathbf{X}^\ell$ for the outer one and conditioning on $\mathbf{X}^\ell$ for inner one. Indeed, the first three terms only depend on the network values after $\mathbf{X}^\ell$. Now, a repeated application of the tower property in $G(\mathbf{X}^\ell)$, together with the results on the gradients of Lemma 3.3.1, easily shows that $G(\mathbf{X}^\ell)$ stays bounded under our hypothesis. To see this one can also simply note that, since the softmax and its derivatives are almost surely bounded, the boundedness of $G(\mathbf{X}^\ell)$ is implied by an analogous statement for a vanilla linear MLP (i.e removing the softmax). In this setting, the random variable inside the expectation in $G(\mathbf{X}^\ell)$ is a finite linear combination of Gaussian products — which has bounded expectation.
All in all, we have that

$$\mathbb{E}\left\|\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{Q,\ell}}\right\|_F^2 \leq \mathbb{E}\left[B_{\mathbf{X}^\ell}\left\|\frac{\partial \alpha_1 \mathbf{S}^\ell}{\partial \mathbf{W}^{Q,\ell}}\right\|_F^2\right],$$

where $B_{\mathbf{X}^\ell}$ is an almost-surely-bounded function of $\mathbf{X}^\ell$. Hence, to show that $\mathbb{E}\left\|\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{Q,\ell}}\right\|_F^2 = 0$, we now just need to show that:

$$\mathbb{E}\left\|\frac{\partial \alpha_1 \mathbf{S}^\ell}{\partial \mathbf{W}^{Q,\ell}}\right\|_F^2 = 0$$

under the rank-1 hypothesis for $\mathbf{X}^\ell$. Let $\mathbf{X}_1^\ell, \dots \mathbf{X}_n^\ell \in \mathbb{R}^{d_v}$ be the representations for the $n$ tokens. Under the rank-1 assumption, each token can be written as a multiple of a single vector $\mathbf{x} \in \mathbb{R}^{d_v}$, and hence there exists $a_1, \dots, a_n \in \mathbb{R}$ such that $\mathbf{X}_1 = a_1\mathbf{x}, \dots, \mathbf{X}_n = a_n\mathbf{x}$. From Lemma 3.3.1, we know that:

$$\mathbb{E}\left\|\frac{\partial \mathbf{S}^\ell}{\partial \mathbf{W}^{Q}}\right\|_F^2 = \frac{\sigma_v^2\sigma_k^2 d^2}{n^2} \cdot \mathbb{E}\left[\|\mathbf{X}^\ell\|_F^2 \cdot \|(\mathbf{X}^\ell)^\top \mathbf{X}^\ell - n\bar{\mathbf{x}}^\ell(\bar{\mathbf{x}}^\ell)^\top\|_F^2\right].$$

The mean token simplifies to $\bar{\mathbf{x}}^l = \frac{\mathbf{x}}{n}\sum_k a_k$ and hence $\left(\bar{\mathbf{x}}^\ell(\bar{\mathbf{x}}^\ell)^\top\right)_{ij} = \frac{1}{n^2}(\sum_k a_k)^2 x_i x_j$. Similarly, $\left((\mathbf{X}^\ell)^\top \mathbf{X}^\ell\right)_{ij} = \sum_k a_k^2 x_i x_j$. If furthermore all the coefficients $a_i$ are the same (which corresponds to the rank collapse assumption $\mathbf{X}^\ell = \mathbf{1}_n \mathbf{x}^T$ analyzed here), then it is easy to see that $\left((\mathbf{X}^\ell)^\top \mathbf{X}^\ell\right)_{ij} - n\left(\bar{\mathbf{x}}^\ell(\bar{\mathbf{x}}^\ell)^\top\right)_{ij} = 0\ \forall i, j$ and hence $\|(\mathbf{X}^\ell)^\top \mathbf{X}^\ell - n\bar{\mathbf{x}}^\ell(\bar{\mathbf{x}}^\ell)^\top\|_F^2 = 0$.

□

### C.2.4 *Making SGD trainable via Softmax Tempering*

The theory devised in Section 3.3.2.3 postulates that a different magnitude between the gradients of the queries/keys and the values should likely be observed. Here, we propose a simple remedy that consists in introducing an inverse temperature scaling $\tau$ inside the softmax that modifies the attention operation to

$$\mathbf{S}_{\tau}^{\ell} := \text{softmax}\left(\frac{\tau}{\sqrt{d_k}}\mathbf{X}^{\ell}\mathbf{W}^{Q}\left(\mathbf{X}^{\ell}\mathbf{W}^{K}\right)^{\top}\right)\mathbf{X}^{\ell}\mathbf{W}^{V}.$$

By computing the gradients of the queries/keys as in Section 3.3.2.3, Equation 85 becomes:

$$\mathbb{E}\left\|\frac{\partial \mathbf{S}_{\tau}}{\partial \mathbf{W}^{Q}}\right\|_{F}^{2} = \tau^{2}\sigma_{x}^{6}\frac{(n-1)}{n}(1-\rho)^{2}d(n+d). \tag{202}$$

Hence, the norm of gradients of the queries/keys scales linearly with $\tau$. On the contrary, the temperature scaling inside the Softmax will not affect the gradients with respect to the values due to the fact that the Softmax normalizes the activations (see Lemma A.1 in Wang et al., 2022a). Hence $\tau$ can be heuristically chosen such that the magnitude of the gradients approximately matches:

$$\mathbb{E}\left\|\frac{\partial \mathbf{S}_{\tau}}{\partial \mathbf{W}^{Q}}\right\|_{F}^{2} \overset{!}{=} \mathbb{E}\left\|\frac{\partial \mathbf{S}_{\tau}}{\partial \mathbf{W}^{V}}\right\|_{F}^{2} \iff \tau^{2} \approx \frac{dn(1+\rho(n-1))}{\sigma_{x}^{4}(1-\rho)^{2}(n+d)(n-1)}.$$

We stress that this requires a constant correlation $\rho$. In practice, this can be estimated as the mean correlation across all pairs of tokens (as we do in the computation of the correlations in Figure 27). Furthermore, both $\rho$ and the variance $\sigma_x^2$ change across layers as our analysis in Section 3.3.2.2 predicts. Hence in practice a different temperature per layer should be adopted. Finally, note that in practice both $\rho$ and $\sigma_x^2$ change during training, and is hard to study their the dynamics under SGD. We leave the time evolution of $\rho$ and $\tau$ as an exciting future direction. Also, the value of $n$ is set to be the average number of tokens per sentence. In this work, we set $\tau$ to a fixed value according to our analysis at initialization.

### C.2.5 *Experimental Details*

TRANSLATION TASK. We now describe the experimental setup regarding the translation task on the IWSLT'14 De-En dataset. Using the ideas detailed in the previous section, we choose a temperature value of $\tau_{\text{final}} = 8.5$ to match the gradient norms of the values and queries as in Equations. (84) and (85). Doing so, we assume a constant small correlation between tokens (also empirically verified in Fig. 69) and set the sequence length $n$ to the average found in our training dataset. Due to instabilities in training, we use warm-up on this temperature value. In short:

$$\tau = \tau_{\text{final}} \cdot \max(1, \frac{\text{step}}{\text{steps}_{\text{warmup}}}),$$

with '$\text{steps}_{\text{warmup}} = 1000$' and 'step' the current training step.
We base our implementation on fairseq (Ott et al., 2019). For the hyperparameter configuration, we mostly rely on the extensive search already done in fairseq (Ott et al., 2019) and Liu et al., 2020a. The final used parameters are exhibited in Table 7. For the final evaluation, we use the best-performing model on the left-out validation set. We apply weight decay as in Loshchilov and Hutter, 2017 for both SGD and Adam.
Finally, in Figure 69 we display the evolution of correlations, residual scaling, and norm of the activations, with depth, for our best trained model. The residual scaling $\alpha_1, \alpha_2$ are trainable parameters. This enables them to weight differently the residual branches if deemed necessary. Although these values increase during training, the correlation between the tokens does not significantly increase, which as implied by our main results, allows efficient propagation of the gradients. The norm of the propagated forward signal tends to slightly increase with depth.

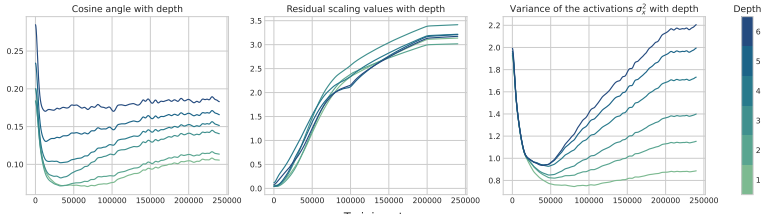


**Figure 69:** *Evolution of the cosine of the angles, the trained residual $\alpha_1, \alpha_2$ and the activation norm throughout our training.*

| | Hyperparameters | Value |
|---|---|---|
| | Max tokens | 4096 |
| | Label smoothing | 0.1 |
| | clip-norm | 0.0 |
| | General Dropout | 0.3 |
| | Attention Dropout | 0.1 |
| | ReLU Dropout | 0.1 |
| | Hidden size | 512 |
| | FFN inner hidden size | 2048 |
| | Attention Heads | 4 |
| Adam | Learning rate | $7\epsilon^{-4}$ |
| | Learning rate scheduler | inverse sqrt |
| | Warm-up updates | 6000 |
| | Warm-up init learning rate | 1e-7 |
| | Adam $(\beta_1, \beta_2)$ | (0.9, 0.98) |
| | Training updates | 100K |
| | Weight decay | 0.0001 |
| SGD | Learning rate | $2\epsilon^{-2}$ |
| | Learning rate scheduler | step |
| | Step scheduler $\gamma$ | 0.1 |
| | Step scheduler update steps | [100K, 200K] |
| | Training updates | 250K |
| | Weight decay | 0.001 |

**Table 7:** *Hyperparameters for the IWSLT'14 De-En translation task (Figure 27, 31).*

# D

## APPENDIX TO CHAPTER 4

*The precision of naming takes away from the uniqueness of seeing.*
– Pierre Bonnard.

### D.1 PROOFS FOR SECTION 4.2

Here we study Decreasing SPS (DecSPS), which combines stepsize decrease with the adaptiveness of SPS.

$$\gamma_k := \frac{1}{c_k} \min \left\{ \frac{f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}}{\|\nabla f_{\mathcal{S}_k}(x^k)\|^2},\ c_{k-1}\gamma_{k-1} \right\}, \tag{DecSPS}$$

for $k \geq 0$, where we set $c_{-1} = c_0$ and $\gamma_{-1} = \gamma_b$ (stepsize bound, similar to Loizou et al., 2021), to get

$$\gamma_0 := \frac{1}{c_0} \cdot \min \left\{ \frac{f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}}{\|\nabla f_{\mathcal{S}_k}(x^k)\|^2},\quad c_0\gamma_b \right\}.$$

#### D.1.1 *Proof of Lemma 4.2.1 (DecSPS Bounds)*

*Proof.* First, note that $\gamma_k$ is trivially *non-increasing* since $\gamma_k \leq c_{k-1}\gamma_{k-1}/c_k$.
Next, we prove the bounds on $\gamma_k$.
For $k = 0$, we can directly use Lemma 4.0.2:

$$\gamma_b \geq \gamma_0 = \frac{1}{c_0} \cdot \min \left\{ \frac{f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}}{\|\nabla f_{\mathcal{S}_k}(x^k)\|^2},\quad c_0\gamma_b \right\} \geq \min \left\{ \frac{1}{2c_0L}, \gamma_b \right\}.$$

Next, we proceed by induction: we assume the proposition holds for $\gamma_k$:

$$\min \left\{ \frac{1}{2c_kL}, \frac{c_0\gamma_b}{c_k} \right\} \leq \gamma_k \leq \frac{c_0\gamma_b}{c_k}.$$

Then, we have : $\gamma_{k+1} = \frac{1}{c_{k+1}} \min\left\{\frac{f_{\mathcal{S}_{k+1}}(x^{k+1}) - f^*_{\mathcal{S}_{k+1}}}{\|\nabla f_{\mathcal{S}_{k+1}}(x^{k+1})\|^2}, \iota\right\}$, where

$$\iota := c_k\gamma_k \in \left[\min\left\{\frac{1}{2L}, c_0\gamma_b\right\}, c_0\gamma_b\right]$$

by the induction hypothesis. This bound directly implies that the proposition holds true for $\gamma_{k+1}$, since again by Lemma 4.0.2 we have

$$\frac{f_{\mathcal{S}_{k+1}}(x^{k+1}) - f^*_{\mathcal{S}_{k+1}}}{\|\nabla f_{\mathcal{S}_{k+1}}(x^{k+1})\|^2} \geq \frac{1}{2L}.$$

This concludes the induction step. □

### D.1.2 *Proof of Theorem 4.2.1 (DecSPS, Convex)*

**Remark D.1.1** (Why was this challenging?)**.** *The fundamental problem towards a proof for DecSPS is that the error to control due to gradient stochasticity does not come from the term $\gamma_k^2\|\nabla f(x^k)\|^2$ in the expansion of $\|x^k - x^*\|^2$, as instead is usual for SGD with decreasing stepsizes. Instead, the error comes from the inner product term $\gamma_k\langle\nabla f(x^k), x^k - x^*\rangle$. Hence, the error is proportional to $\gamma_k$, and not $\gamma^2$. As a result, the usual Robbins-Monro conditions (Robbins and Monro, 1951) do not yield convergence. A similar problem is discussed for AdaGrad in Ward et al., 2019.*

*Proof.* Note that from the definition we have that $\gamma_k \leq \frac{1}{c_k} \cdot \frac{f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}}{\|\nabla f_{\mathcal{S}_k}(x^k)\|^2}$. Multiplying by $\gamma_k$ and rearranging terms we get the fundamental inequality

$$\gamma_k^2\|\nabla f_{\mathcal{S}_k}(x^k)\|^2 \leq \frac{\gamma_k}{c_k}[f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}]. \tag{203}$$

Using the definition of DecSPS and convexity we get

$$\begin{aligned}
&\|x^{k+1} - x^*\|^2 \\
&= \|x^k - \gamma_k\nabla f_{\mathcal{S}_k}(x^k) - x^*\|^2 \\
&\overset{(203)}{\leq} \|x^k - x^*\|^2 - 2\gamma_k\langle\nabla f_{\mathcal{S}_k}(x^k), x^k - x^*\rangle + \frac{\gamma_k}{c_k}(f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}).
\end{aligned}$$

Next, using convexity,

$$\begin{aligned}
&\|x^{k+1}-x^*\|^2 \\
&\leq \|x^k-x^*\|^2 - 2\gamma_k[f_{\mathcal{S}_k}(x^k)-f_{\mathcal{S}_k}(x^*)] + \frac{\gamma_k}{c_k}[f_{\mathcal{S}_k}(x^k)-f_{\mathcal{S}_k}(x^*)+f_{\mathcal{S}_k}(x^*)-\ell^*_{\mathcal{S}_k}] \\
&= \|x^k-x^*\|^2 - 2\gamma_k[f_{\mathcal{S}_k}(x^k)-f_{\mathcal{S}_k}(x^*)] + \frac{\gamma_k}{c_k}[f_{\mathcal{S}_k}(x^k)-f_{\mathcal{S}_k}(x^*)] + \frac{\gamma_k}{c_k}[f_{\mathcal{S}_k}(x^*)-\ell^*_{\mathcal{S}_k}] \\
&= \|x^k-x^*\|^2 - \left(2-\frac{1}{c_k}\right)\gamma_k[f_{\mathcal{S}_k}(x^k)-f_{\mathcal{S}_k}(x^*)] + \frac{\gamma_k}{c_k}[f_{\mathcal{S}_k}(x^*)-\ell^*_{\mathcal{S}_k}].
\end{aligned}$$

Let us divide everything by $\gamma_k > 0$.

$$\frac{\|x^{k+1}-x^*\|^2}{\gamma_k} \leq \frac{\|x^k-x^*\|^2}{\gamma_k} - \left(2-\frac{1}{c_k}\right)[f_{\mathcal{S}_k}(x^k)-f_{\mathcal{S}_k}(x^*)] + \frac{1}{c_k}[f_{\mathcal{S}_k}(x^*)-\ell^*_{\mathcal{S}_k}].$$

Since by hypothesis $c_k \geq 1$ for all $k \in \mathbb{N}$, we have $\left(2-\frac{1}{c_k}\right) \geq 1$ and therefore

$$f_{\mathcal{S}_k}(x^k)-f_{\mathcal{S}_k}(x^*) \leq \frac{\|x^k-x^*\|^2}{\gamma_k} - \frac{\|x^{k+1}-x^*\|^2}{\gamma_k} + \frac{1}{c_k}[f_{\mathcal{S}_k}(x^*)-\ell^*_{\mathcal{S}_k}].$$

Next, summing from $k=0$ to $K-1$:

$$\begin{aligned}
&\sum_{k=0}^{K-1} f_{\mathcal{S}_k}(x^k)-f_{\mathcal{S}_k}(x^*) \\
&\leq \sum_{k=0}^{K-1}\frac{\|x^k-x^*\|^2}{\gamma_k} - \sum_{k=0}^{K-1}\frac{\|x^{k+1}-x^*\|^2}{\gamma_k} + \sum_{k=0}^{K-1}\frac{1}{c_k}[f_{\mathcal{S}_k}(x^*)-\ell^*_{\mathcal{S}_k}].
\end{aligned}$$

And therefore

$$
\begin{aligned}
&\sum_{k=0}^{K-1} f_{\mathcal{S}_k}(x^k) - f_{\mathcal{S}_k}(x^*) \\
&\leq \sum_{k=0}^{K-1} \frac{\|x^k - x^*\|^2}{\gamma_k} - \sum_{k=0}^{K-1} \frac{\|x^{k+1} - x^*\|^2}{\gamma_k} + \sum_{k=0}^{K-1} \frac{1}{c_k}[f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}] \\
&\leq \frac{\|x^0 - x^*\|^2}{\gamma_0} + \sum_{k=1}^{K-1} \frac{\|x^k - x^*\|^2}{\gamma_k} - \sum_{k=0}^{K-2} \frac{\|x^{k+1} - x^*\|^2}{\gamma_k} - \frac{\|x^K - x^*\|^2}{\gamma_{K-2}} + \sum_{k=0}^{K-1} \frac{1}{c_k}[f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}] \\
&\leq \frac{\|x^0 - x^*\|^2}{\gamma_0} + \sum_{k=0}^{K-2} \frac{\|x^{k+1} - x^*\|^2}{\gamma_{k+1}} - \sum_{k=0}^{K-2} \frac{\|x^{k+1} - x^*\|^2}{\gamma_k} + \sum_{k=0}^{K-1} \frac{1}{c_k}[f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}] \\
&\leq \frac{\|x^0 - x^*\|^2}{\gamma_0} + \sum_{k=0}^{K-2} \left(\frac{1}{\gamma_{k+1}} - \frac{1}{\gamma_k}\right) \|x^{k+1} - x^*\|^2 + \sum_{k=0}^{K-1} \frac{1}{c_k}[f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}] \\
&\leq D^2 \left[\frac{1}{\gamma_0} + \sum_{k=0}^{K-2} \left(\frac{1}{\gamma_{k+1}} - \frac{1}{\gamma_k}\right)\right] + \sum_{k=0}^{K-1} \frac{1}{c_k}[f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}] \\
&\leq \frac{D^2}{\gamma_{K-1}} + \sum_{k=0}^{K-1} \frac{1}{c_k}[f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}].
\end{aligned}
$$

**Remark D.1.2** (Where did we use the modified SPS definition?)**.** *In the last steps above, we are able to collect $D^2$ because $\left(\frac{1}{\gamma_{k+1}} - \frac{1}{\gamma_k}\right) \geq 0$. This is guaranteed by the new SPS definition (DecSPS), along with the fact that $c_k$ is increasing. Note that one could not perform this step under the original SPS update rule of Loizou et al., 2021.*

Thanks to Lemma 4.2.1, we have:

$$\gamma_k \geq \min\left\{\frac{1}{2c_k L}, \frac{c_0 \gamma_b}{c_k}\right\}.$$

Hence,

$$\frac{1}{\gamma_k} \leq c_k \cdot \max\left\{2L, \frac{1}{c_0 \gamma_b}\right\}. \tag{204}$$

Let us call $\tilde{L} = \max\left\{L, \frac{1}{2c_0\gamma_b}\right\}$. By combining Eq. (204) with the bound above and dividing by $K$ we get:

$$\frac{1}{K}\sum_{k=0}^{K-1} f_{\mathcal{S}_k}(x^k) - f_{\mathcal{S}_k}(x^*) \leq \frac{2c_{K-1}\tilde{L}D^2}{K} + \frac{1}{K}\sum_{k=0}^{K-1} \frac{[f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}]}{c_k},$$

We conclude by taking the expectation and using Jensen's inequality as follows:

$$\mathbb{E}\left[f(\bar{x}^K) - f(x^*)\right] \overset{\text{Jensen}}{\leq} \frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left[f(x^k) - f(x^*)\right] \leq \frac{2c_{K-1}\tilde{L}D^2}{K} + \frac{1}{K}\sum_{k=0}^{K-1}\frac{\hat{\sigma}^2_{B,\max}}{c_k}.$$

where $\hat{\sigma}^2_B$ is as defined in Eq. (103). □

**Remark D.1.3** (Second term does not depend on $\gamma_b$)**.** *Note that, in the convergence rate, the second term does not depend on $\gamma_b$ while the first does. This is different from the original SPS result (Loizou et al., 2021), and due to the different proof technique: specifically, we divide by $\gamma_k$ early in the proof — and not at the end. To point to the exact source of this difference, we invite the reader to inspect Equation 24 in the appendix[1] of Loizou et al., 2021: the last term there is proportional to $\gamma_b/\alpha$, where $\alpha$ is a lower bound on the SPS and $\gamma_b$ is an upper bound. In our proof approach, these terms — which bound the same quantity — effectively cancel out (because we divide by $\gamma_k$ earlier in the proof), at the price of having $D^2$ in the first term.*

### D.1.3 *Proof of Proposition 4.2.1 (Domain Bound)*

We need the following lemma. An illustration of the result can be found in Fig. 70.

**Lemma D.1.1.** *Let $z^{k+1} = A_k z^k + \varepsilon_k$ with $A_k = (1 - a/\sqrt{k+1})$ and $\varepsilon_k = b/\sqrt{k+1}$. If $z^0 > 0$, $0 < a \leq 1$, $b > 0$, then $z^k \leq \max\{z^0, b/a\}$ for all $k \geq 0$.*

*Proof.* Simple to prove by induction. The base case is trivial, since $z^0 \leq \max\{z^0, b/a\}$. Let us now assume the proposition holds true for $z^k$ (that is, $z^k \leq \max\{z^0, b/a\}$) , we want to show it holds true for $k+1$. We have

$$z^{k+1} = \left(1 - \frac{a}{\sqrt{k+1}}\right) z^k + \frac{b}{\sqrt{k+1}}.$$

If $b/a = \max\{z^0, b/a\}$, then we get, by induction

$$z^{k+1} \leq \left(1 - \frac{a}{\sqrt{k+1}}\right)\frac{b}{a} + \frac{b}{\sqrt{k+1}} = \frac{b}{a} = \max\{z^0, b/a\}.$$

1 http://proceedings.mlr.press/v130/loizou21a/loizou21a-supp.pdf

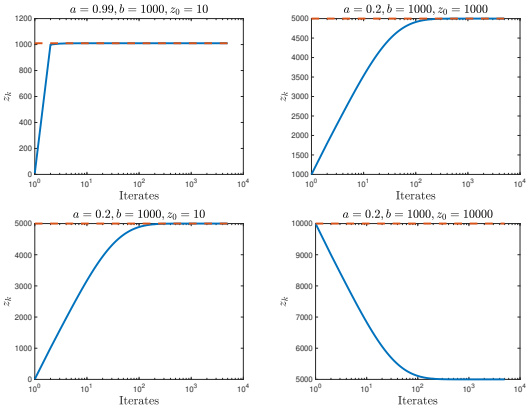


**Figure 70:** *Numerical Verification of Lemma D.1.1. Bound in the lemma is indicated with dashed line.*

Else, if $z^0 = \max\{z^0, b/a\}$, then by induction

$$z^{k+1} \leq \left(1 - \frac{a}{\sqrt{k+1}}\right) z^0 + \frac{b}{\sqrt{k+1}} = z^0 - \frac{az^0 - b}{\sqrt{k+1}} \leq z^0 = \max\{z^0, b/a\},$$

where the last inequality holds because $az^0 - b > 0$ and $a$ is positive. This completes the proof. □

*Proof.* Using the SPS definition we directly get

$$\begin{aligned}\|x^{k+1} - x^*\|^2 &= \|x^k - \gamma_k \nabla f_{\mathcal{S}_k}(x^k) - x^*\|^2 \\ &= \|x^k - x^*\|^2 - 2\gamma_k \langle \nabla f_{\mathcal{S}_k}(x^k), x^k - x^* \rangle + \gamma_k^2 \|\nabla f_{\mathcal{S}_k}(x^k)\|^2 \\ &\leq \|x^k - x^*\|^2 - 2\gamma_k \langle \nabla f_{\mathcal{S}_k}(x^k), x^k - x^* \rangle + \frac{\gamma_k}{c_k}(f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}),\end{aligned} \tag{205}$$

where (as always) we used the fact that since from the definition $\gamma_k := \frac{1}{c_k} \cdot \min\left\{\frac{f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}}{\|\nabla f_{\mathcal{S}_k}(x^k)\|^2}, \quad c_{k-1}\gamma_{k-1}\right\}$, then $\gamma_k \leq \frac{1}{c_k} \cdot \frac{f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}}{\|\nabla f_{\mathcal{S}_k}(x^k)\|^2}$ and we have

$$\gamma_k^2 \|\nabla f_{\mathcal{S}_k}(x^k)\|^2 \leq \frac{1}{c_k}[f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}].$$

Now recall that, if each $f_i$ is $\mu_i$-strongly convex then for any $x, y \in \mathbb{R}^d$ we have, setting $\mu = \min_i \mu_i$

$$-\langle \nabla f_{\mathcal{S}_k}(x), x - y\rangle \leq -\frac{\mu}{2}\|x - y\|^2 - f_{\mathcal{S}_k}(x) + f_{\mathcal{S}_k}(y).$$

For $y = x^*$ and $x = x^k$, this implies

$$-\langle \nabla f_{\mathcal{S}_k}(x^k), x^k - x^*\rangle \leq -\frac{\mu}{2}\|x^k - x^*\|^2 - f_{\mathcal{S}_k}(x^k) + f_{\mathcal{S}_k}(x^*).$$

Adding and subtracting $\ell^*_{\mathcal{S}_k}$ to the RHS of the inequality above, we get

$$-\langle \nabla f_{\mathcal{S}_k}(x^k), x^k - x^*\rangle \leq -\frac{\mu}{2}\|x^k - x^*\|^2 - (f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}) + (f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}).$$

Since $\gamma_k > 0$, we can substitute this inequality in Equation (205) and get

$$\begin{aligned}
&\|x^{k+1} - x^*\|^2 \\
&\leq \|x^k - x^*\|^2 + \frac{\gamma_k}{c_k}(f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}) \\
&\underbrace{-\mu\gamma_k\|x^k - x^*\|^2 - 2\gamma_k(f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}) + 2\gamma_k(f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k})}_{\leq -2\gamma_k\langle \nabla f_{\mathcal{S}_k}(x^k), x^k - x^*\rangle}.
\end{aligned}$$

Rearranging a few terms we get

$$\begin{aligned}
&\|x^{k+1} - x^*\|^2 \\
&\leq (1 - \mu\gamma_k)\|x^k - x^*\|^2 + \frac{\gamma_k}{c_k}(f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}) \\
&\quad - 2\gamma_k(f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}) + 2\gamma_k(f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}) \\
&\leq (1 - \mu\gamma_k)\|x^k - x^*\|^2 - \left(2 - \frac{1}{c_k}\right)\gamma_k(f_{\mathcal{S}_k}(x^k) - \ell^*_{\mathcal{S}_k}) + 2\gamma_k(f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}).
\end{aligned}$$

Since we assumed $c_k \geq 1/2$ for all $k \in \mathbb{N}$, we can drop the term $-\left(2 - \frac{1}{c_k}\right)\gamma_k[f_{\mathcal{S}_k}(x^k) - f^*_{\mathcal{S}_k}]$, since also $f^*_{\mathcal{S}_k} \geq \ell^*_{\mathcal{S}_k}$. Hence, we get the following bound:

$$\begin{aligned}
\|x^{k+1} - x^*\|^2 &\leq (1 - \mu\gamma_k)\|x^k - x^*\|^2 + 2\gamma_k(f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}) \\
&\leq (1 - \mu\gamma_k)\,\|x^k - x^*\|^2 + \frac{2c_0\gamma_b}{c_k}(f_{\mathcal{S}_k}(x^*) - \ell^*_{\mathcal{S}_k}) \\
&\leq (1 - \mu\gamma_k)\,\|x^k - x^*\|^2 + \frac{2c_0\gamma_b\hat{\sigma}^2_{B,\max}}{c_k},
\end{aligned}$$

where we used the inequality $\min\left\{\frac{1}{2c_kL}, \frac{c_0\gamma_b}{c_k}\right\} \leq \gamma_k \leq \frac{c_0\gamma_b}{c_k}$ (Lemma 4.2.1). Now we seek an upper bound for the contraction factor. Under $c_k = \sqrt{k+1}$, using again Lemma 4.2.1 we have, since $c_0 = 1$,

$$1 - \mu\gamma_k \geq 1 - \frac{\min\left\{\frac{\mu}{2L}, \mu\gamma_b\right\}}{\sqrt{k+1}}.$$

Now have all ingredients to bound the iterates: the result follows from Lemma D.1.1 using $a = \min\left\{\frac{\mu}{2L}, \mu\gamma_b\right\}$ and $b = 2c_0\gamma_b\hat{\sigma}^2_{B,\max}$. So, we get

$$\|x^{k+1} - x^*\|^2 \leq \max\left\{\|x^0 - x^*\|^2, \frac{2c_0\gamma_b\hat{\sigma}^2_{B,\max}}{\min\left\{\frac{\mu}{2L}, \mu\gamma_b\right\}}\right\}, \text{for all } k \geq 0.$$

This completes the proof. □

## D.2 PROOFS FOR SECTION 4.3

Let $f = \frac{1}{N}\sum_{i=1}^N f_i$, where each $f_i : \mathbb{R}^d \to \mathbb{R}$ is non-negative, $L_i$-smooth and convex. Let $L = \max_{i\in[N]} L_i$. Consider stochastic gradient descent with batch size 1 and let $i_k$ be the data index sampled at iteration $k$. For any value of $\sigma > 0$, the stepsize we consider is

$$\gamma_k = \frac{\sigma}{1 + \frac{\sigma}{2f_{i_k}(x^k)}\|\nabla f_{i_k}(x^k)\|^2} \qquad \text{(NGN-stochastic)}$$

### D.2.1 *Fondamental NGN Properties*

We are ready to prove the first useful property of the NGN stepsize.

**Lemma D.2.1** (Stepsize bounds). *Let each $f_i : \mathbb{R}^d \to \mathbb{R}$ be non-negative, differentiable and L-smooth. Consider $\gamma_k$ as in Eq. (NGN-stochastic), we have*

$$\gamma_k \in \left[\frac{\sigma}{1+\sigma L}, \sigma\right] = \left[\frac{1}{L+\sigma^{-1}}, \sigma\right].$$

*Proof.* Recall again that for a smooth function $2L(f_{i_k}(x) - f_{i_k}^*) \geq \|\nabla f_{i_k}(x)\|^2$ (Lemma 4.0.2). Since we assumed each $f_{i_k}$ is non-negative we also have $2Lf_{i_k}(x) \geq \|\nabla f_{i_k}(x)\|^2$. All in all,

$$0 \leq \frac{\|\nabla f_{i_k}(x)\|^2}{2f_{i_k}(x)} \leq L.$$

This directly implies a stepsize range above. □

Further, the NGN stepsize definition provides a useful equality.

**Lemma D.2.2** (Fundamental Equality)**.** *Consider $\gamma_k$ as in Eq.* (NGN-stochastic)*. One has*

$$\gamma_k \|\nabla f_{i_k}(x)\|^2 = 2\left(\frac{\sigma - \gamma_k}{\sigma}\right) f_{i_k}(x).$$

*Proof.* The definition of our stepsize implies

$$\left(1 + \frac{\sigma}{2f_{i_k}(x)}\|\nabla f_{i_k}(x)\|^2\right)\gamma_k = \sigma.$$

Which one can write as

$$\frac{\sigma}{2f_{i_k}(x)}\|\nabla f_{i_k}(x)\|^2\gamma_k = \sigma - \gamma_k.$$

This proves the result. □

**Lemma D.2.3** (Fundamental inequality)**.** *Let each $f_i : \mathbb{R}^d \to \mathbb{R}$ be non-negative, L-smooth and convex. Consider $\gamma_k$ as in Eq.* (NGN-stochastic)*, we have*

$$\begin{aligned}&\gamma_k^2\|\nabla f_{i_k}(x^k)\|^2\\ &\leq \left(\frac{4\sigma L}{1+2\sigma L}\right)\gamma_k[f_{i_k}(x^k) - f_{i_k}^*] + \frac{2\sigma^2 L}{1+\sigma L}\cdot\max\left\{0, \frac{2\sigma L - 1}{2\sigma L + 1}\right\}\cdot f_{i_k}^*.\end{aligned}$$

*For $\sigma \leq 1/(2L)$ or $f_{i_k}^* = 0$ for all $k$ we have no interpolation error.*

*Proof.* Lemma 4.0.2&D.2.2 imply the following relations:

$$\begin{aligned}\gamma_k^2\|\nabla f_{i_k}(x^k)\|^2 &\leq 2L\gamma_k^2(f_{i_k}(x) - f_{i_k}^*)\\ \gamma_k^2\|\nabla f_{i_k}(x^k)\|^2 &= 2\gamma_k\left(\frac{\sigma - \gamma_k}{\sigma}\right) f_{i_k}(x^k)\end{aligned}$$

This implies that for any $\delta \in (-\infty, 1]$ we have

$$(1-\delta)\gamma_k^2 \|\nabla f_{i_k}(x^k)\|^2 \leq 2(1-\delta)L\gamma_k^2(f_{i_k}(x) - f_{i_k}^*)$$
$$\delta\gamma_k^2 \|\nabla f_{i_k}(x^k)\|^2 = 2\delta\gamma_k \left(\frac{\sigma - \gamma_k}{\sigma}\right) f_{i_k}(x^k).$$

Summing the two inequalities above, we get

$$\gamma_k^2 \|\nabla f_{i_k}(x^k)\|^2 \leq 2\delta\gamma_k \left(\frac{\sigma - \gamma_k}{\sigma}\right) f_{i_k}(x^k) + 2(1-\delta)L\gamma_k^2(f_{i_k}(x^k) - f_{i_k}^*).$$

After collecting a few terms,

$$\gamma_k^2 \|\nabla f_{i_k}(x^k)\|^2$$
$$\leq 2\gamma_k \underbrace{\left[\delta\left(\frac{\sigma - \gamma_k}{\sigma}\right) + (1-\delta)L\gamma_k\right]}_{A(\delta)} [f_{i_k}(x^k) - f_{i_k}^*] + \underbrace{2\gamma_k\delta\left(\frac{\sigma - \gamma_k}{\sigma}\right)}_{B(\delta)} f_{i_k}^*.$$

We would now want to choose $\delta$, which minimizes $A(\delta)$ and $B(\delta)$ simultaneously. It turns out that a convenient choice is

$$\delta = \frac{2\sigma L - 1}{2\sigma L + 1}$$

Note that since $2\sigma L > 0$, $\delta$ is a real number in the range $[-1, 1]$ (which is a valid subset of $[-\infty, 1]$). We provide a paragraph after the proof justifying this choice. We have

$$\begin{aligned} A(\delta) &= \frac{2\sigma L - 1}{2\sigma L + 1}\left(1 - \frac{\gamma_k}{\sigma}\right) + \frac{2L\gamma_k}{2\sigma L + 1} \\ &= \frac{2\sigma L - 1}{2\sigma L + 1} - \frac{2\sigma L - 1}{2\sigma L + 1}\frac{\gamma_k}{\sigma} + \frac{2L\gamma_k}{2\sigma L + 1} \\ &= \frac{2\sigma L - 1}{2\sigma L + 1} + \frac{\gamma_k}{(2\sigma L + 1)\sigma} \\ &\leq \frac{2\sigma L}{2\sigma L + 1}, \end{aligned}$$

where in the last line we used the property $\gamma_k \leq \sigma$. The bound becomes:

$$\gamma_k^2 \|\nabla f_{i_k}(x^k)\|^2$$
$$\leq \left(\frac{4\sigma L}{1 + 2\sigma L}\right)\gamma_k[f_{i_k}(x^k) - f_{i_k}^*] + 2\gamma_k\left(\frac{2\sigma L - 1}{2\sigma L + 1}\right)\left(\frac{\sigma - \gamma_k}{\sigma}\right) f_{i_k}^*.$$

To bound the second term, the error term, we have two cases:

- If $\sigma \leq 1/(2L)$ then $\delta = \frac{2\sigma L - 1}{2\sigma L + 1}$ is negative, and we have

$$2\gamma_k \left(\frac{2\sigma L - 1}{2\sigma L + 1}\right)\left(\frac{\sigma - \gamma_k}{\sigma}\right) f_{i_k}^* \leq 0,$$

since the worst case is $\gamma_k = \sigma$.

- Otherwise, $\delta > 0$ and we can proceed as follows, using the fact that $\gamma_k \in \left[\frac{\sigma}{1+\sigma L}, \sigma\right]$

$$\begin{aligned} 2\gamma_k \left(\frac{2\sigma L - 1}{2\sigma L + 1}\right)\left(\frac{\sigma - \gamma_k}{\sigma}\right) f_{i_k}^* &\leq 2\sigma \left(\frac{2\sigma L - 1}{2\sigma L + 1}\right)\left(\frac{\sigma - \frac{\sigma}{1+\sigma L}}{\sigma}\right) f_{i_k}^* \\ &= 2\sigma \left(\frac{2\sigma L - 1}{2\sigma L + 1}\right)\left(\frac{\sigma L}{1+\sigma L}\right) f_{i_k}^* \\ &= \frac{2\sigma^2 L}{1+\sigma L}\left(\frac{2\sigma L - 1}{2\sigma L + 1}\right) f_{i_k}^*, \end{aligned}$$

All in all, considering both cases, we get the following upper bound for the error term:

$$\frac{2\sigma^2 L}{1+\sigma L} \cdot \max\left\{0, \frac{2\sigma L - 1}{2\sigma L + 1}\right\} \cdot f_{i_k}^*.$$

This concludes the proof. □

**Remark D.2.1** (On the choice of $\delta$ in Lemma D.2.3)**.** *In the context of the proof of Lemma D.2.3, let us assume we want to find $\delta$ such that $A(\delta) \leq \alpha$. That implies*

$$\begin{aligned} &\delta\left(\frac{\sigma - \gamma_k}{\sigma}\right) + (1-\delta)L\gamma_k \leq \alpha \\ \iff &\delta\sigma + [(1-\delta)L\sigma - \delta]\,\gamma_k \leq \alpha\sigma \\ \iff &[(1-\delta)L\sigma - \delta]\,\gamma_k \leq (\alpha - \delta)\sigma \\ \iff &[(1-\delta)L\sigma - \delta] \leq (\alpha - \delta)\frac{\sigma}{\gamma_k}. \end{aligned}$$

*For $\alpha \geq \delta$, the right-hand side is positive. Further, note that $\frac{\sigma}{\gamma_k} \in [1, 1+\sigma L]$. So if we like the inequality to hold for every value of $\gamma$ we need (worst case analysis)*

$$[(1-\delta)L\sigma - \delta] \leq (\alpha - \delta) \iff (1-\delta)L\sigma \leq \alpha \iff \delta \geq 1 - \frac{\alpha}{L\sigma}.$$

*Since $\alpha \geq \delta$ we also need*

$$\alpha \geq 1 - \frac{\alpha}{L\sigma} \iff L\sigma\alpha \geq L\sigma - \alpha \iff \alpha \geq \frac{\sigma L}{1+\sigma L} = 1 - \frac{1}{1+\sigma L}$$

*The bound becomes*

$$\gamma_k^2 \|\nabla f_{i_k}(x^k)\|^2 \leq 2\alpha\gamma_k[f_{i_k}(x^k) - f_{i_k}^*] + \underbrace{2\gamma_k\delta\left(\frac{\sigma-\gamma_k}{\sigma}\right)}_{B(\delta)} f_{i_k}^*.$$

*For the sake of minimizing the first term in the bound, it would make sense to use $\alpha = \frac{\sigma L}{1+\sigma L}$. However, under this condition we get that $\delta \geq \frac{\sigma L}{1+\sigma L}$ as well. This is not ideal since we want $B(\delta)$, the error factor, to vanish for small $\gamma$. To do this, we need to slightly increase the value of $\alpha$ to allow $\delta$ to get negative for small values of $y = L\sigma$. Note that for $\delta$ to be (potentially) negative we need*

$$1 - \frac{\alpha}{\sigma L} \leq 0 \iff \alpha \geq L\sigma.$$

*Hence, if we want to keep $\alpha < 1$ (needed for proofs), we can only have this condition for $\sigma \leq 1/L$ — i.e. the case where we know convergence holds. To this end, let us consider*

$$1 > \alpha = \frac{2\sigma L}{1+2\sigma L} = \frac{\sigma L}{1/2+\sigma L} \geq \frac{\sigma L}{1+\sigma L}.$$

*Using this $\alpha$, we get*

$$\delta \geq 1 - \frac{\alpha}{L\sigma} = 1 - \frac{2}{1+2\sigma L} = \frac{2\sigma L - 1}{2\sigma L + 1}.$$

*Note that if $\sigma \leq \frac{1}{2L}$ then the minimum allowed $\delta$ is negative. Let us pick for $\delta$ its minimum allowed value.*

### D.2.2 *Proof of Theorem 4.3.1 (NGN, Convex)*

By assumption, each $f_i$ is convex and $L$-smooth. Expanding the squared distance and using convexity, we get:

$$\begin{aligned}
\|x^{k+1} - x^*\|^2 &= \|(x^{k+1} - x^k) + (x^k - x^*)\|^2 \\
&= \|x^k - x^*\|^2 + 2\langle x^k - x^*, x^{k+1} - x^k\rangle + \|x^{k+1} - x^k\|^2 \\
&= \|x^k - x^*\|^2 - 2\gamma_k\langle x^k - x^*, \nabla f_{i_k}(x^k)\rangle + \gamma_k^2\|\nabla f_{i_k}(x^k)\|^2 \\
&\leq \|x^k - x^*\|^2 - 2\gamma_k[f_{i_k}(x^k) - f_{i_k}(x^*)] + \gamma_k^2\|\nabla f_{i_k}(x^k)\|^2.
\end{aligned}$$

We now make use of Lemma D.2.3:

$$\begin{aligned}
&\|x^{k+1} - x^*\|^2 \\
&\leq \|x^k - x^*\|^2 - 2\gamma_k[f_{i_k}(x^k) - f_{i_k}(x^*)] \\
&\quad + \left(\frac{4\sigma L}{1+2\sigma L}\right)\gamma_k[f_{i_k}(x^k) - f_{i_k}^*] + \frac{2\sigma^2 L}{1+\sigma L}\cdot\max\left\{0, \frac{2\sigma L - 1}{2\sigma L + 1}\right\} f_{i_k}^*.
\end{aligned}$$

Note that taking the expectation conditional on $x^k$ at this point (as done in the classical SGD proof) is not feasible: indeed, the variable $f_{i_k}(x^k) - f_{i_k}(x^*)$, which would have expectation $f(x^k) - f(x^*)$, is correlated with $\gamma_k$ — meaning that we would need to consider the expectation of the product $\gamma_k(f_{i_k}(x^k) - f_{i_k}(x^*))$.

The analysis of Loizou (Loizou et al., 2021) in the context of SPS consists in writing $f_{i_k}(x^k) - f_{i_k}(x^*) = [f_{i_k}(x^k) - f_{i_k}^*] - [f_{i_k}(x^*) - f_{i_k}^*]$, where both terms within brackets are positive and therefore one can use the step-size bounds before taking the expectation. This is a smart approach for a quick computation; however it introduces a bias term $\mathbf{E}[\gamma_k(f_{i_k}(x^*) - f_{i_k}^*)] \leq \sigma[f(x^*) - \mathbf{E}_i f_i^*] = O(\sigma)$. It is instead desirable, if the method allows, to have error terms only of the kind $O(\sigma^2)$, so that one can guarantee later in the proof that, as $\sigma \to 0$, the method converges to the problem solution.

To this end, we write $\gamma_k = \rho + \epsilon_k$, where both $\gamma$ and $\epsilon_k$ are non-negative and $\alpha$ is deterministic. For intuition, the reader can think of $\rho$ as a step-size lower bound such that ideally $\rho = O(\sigma)$ and $\xi_k = O(\sigma^2)$ for all realizations of $\gamma_k$— we will be more precise in the next lines:

$$\begin{aligned}
&\|x^{k+1} - x^*\|^2 - \|x^k - x^*\|^2 \\
&\leq -2\rho[f_{i_k}(x^k) - f_{i_k}(x^*)] - 2\epsilon_k[f_{i_k}(x^k) - f_{i_k}(x^*)] \\
&\quad + \left(\frac{4\sigma L}{1+2\sigma L}\right)\gamma_k[f_{i_k}(x^k) - f_{i_k}^*] + O(\sigma^2) f_{i_k}^*,
\end{aligned}$$

where we wrote the last factor simply as $O(\sigma^2)$ for brevity but will plug in the correct expression at the end of the proof.

At this point, we write $f_{i_k}(x^k) - f_{i_k}(x^*) = [f_{i_k}(x^k) - f_{i_k}^*] - [f_{i_k}(x^*) - f_{i_k}^*]$ *only for the second term in the bound above*. Our purpose, is to make the resulting term $-2\epsilon_k[f_{i_k}(x^k) - f_{i_k}^*]$ (negative) dominant compared to the

third term in the bound above (positive). In formulas, since the bound on the distance update $\|x^{k+1} - x^*\|^2 - \|x^k - x^*\|^2$ becomes

$$\begin{aligned}&-2\rho[f_{i_k}(x^k) - f_{i_k}(x^*)]\\&\quad - 2\left(\epsilon_k - \frac{2\sigma L\gamma_k}{1+2\sigma L}\right)[f_{i_k}(x^k) - f_{i_k}^*] + 2\epsilon_k[f_{i_k}(x^*) - f_{i_k}^*] + O(\sigma^2)f_{i_k}^*,\end{aligned}$$

we require $\epsilon_k - \frac{2\sigma L}{1+2\sigma L}\gamma_k \geq 0$. Note that $\epsilon_k = \gamma_k - \rho$. Therefore, we need

$$\left(1 - \frac{2\sigma L}{1+2\sigma L}\right)\gamma_k \geq \rho \quad \implies \gamma_k \geq (1+2\sigma L)\rho.$$

Since $\gamma_k \geq \frac{\sigma}{1+\sigma L}$ thanks to Lemma D.2.1, we have that a sufficient condition is

$$\rho \leq \frac{\sigma}{(1+2\sigma L)(1+\sigma L)}.$$

Let us then pick $\rho$ equal to this upper bound. Our bound on the distance update simplifies as

$$-\frac{2\sigma}{(1+2\sigma L)(1+\sigma L)}[f_{i_k}(x^k) - f_{i_k}(x^*)] + 2\epsilon_k[f_{i_k}(x^*) - f_{i_k}^*] + O(\sigma^2)f_{i_k}^*.$$

We now get to the interesting part: what is the order of $\epsilon_k$ under our choice for $\rho$?

$$\epsilon_k = \gamma_k - \rho \leq \sigma - \frac{\sigma}{(1+2\sigma L)(1+\sigma L)} = \frac{\sigma(1+2\sigma L)(1+\sigma L) - \sigma}{(1+2\sigma L)(1+\sigma L)}.$$

Simplifying the bound, we get the desired result: $\epsilon_k = O(\sigma^2)$. Indeed,

$$\epsilon_k \leq \frac{3L\sigma^2 + 2L^2\sigma^3}{(1+2\sigma L)(1+\sigma L)} = L\sigma^2\frac{3+2L\sigma}{(1+2\sigma L)(1+\sigma L)} \leq \frac{3L\sigma^2}{1+2\sigma L}.$$

All in all, our bound becomes

$$\begin{aligned}&\|x^{k+1} - x^*\|^2 - \|x^k - x^*\|^2\\&\leq -T_0(\sigma)[f_{i_k}(x^k) - f_{i_k}(x^*)] + T_1(\sigma^2)[f_{i_k}(x^*) - f_{i_k}^*] + T_2(\sigma^2)f_{i_k}^*, \quad (206)\end{aligned}$$

with $T_0(\sigma) = \frac{2\sigma}{(1+2\sigma L)(1+\sigma L)}$, $T_1(\sigma^2) = \frac{6L\sigma^2}{1+2\sigma L}$, and finally $T_2(\sigma^2) = \frac{2\sigma^2 L}{1+\sigma L} \cdot \max\left\{0, \frac{2\sigma L - 1}{2\sigma L + 1}\right\}$.

We are finally ready to take the conditional expectation with respect to $k$:

$$\begin{aligned}\mathbf{E}_k\|x^{k+1}-x^*\|^2 \leq \|x^k-x^*\|^2 - T_0(\sigma)[f(x^k)-f(x^*)] \\ + T_1(\sigma^2)\mathbf{E}_k[f_{i_k}(x^*)-f_{i_k}^*] + T_2(\sigma^2)\mathbf{E}_k[f_{i_k}^*].\end{aligned}$$

Let us call $E_k$ the expected error at step $k$:

$$\begin{aligned}E(\sigma^2) &:= T_1(\sigma^2)\mathbf{E}_k[f_{i_k}(x^*)-f_{i_k}^*] + T_2(\sigma^2)\mathbf{E}_k[f_{i_k}^*] \\ &\leq T_1(\sigma^2)\Delta_{\text{int}} + T_2(\sigma^2)\Delta_{\text{pos}},\end{aligned}$$

where $\Delta_{\text{int}} := \mathbf{E}_k[f_{i_k}(x^*)-f_{i_k}^*]$ and $\Delta_{\text{pos}} := \mathbf{E}_k[f_{i_k}^*]$. Therefore we get the compact formula

$$\mathbf{E}_k\|x^{k+1}-x^*\|^2 - \|x^k-x^*\|^2 \leq -T_0(\sigma)[f(x^k)-f(x^*)] + E(\sigma^2)$$

Taking the expectation again and using the tower property

$$\mathbf{E}\|x^{k+1}-x^*\|^2 - \mathbf{E}\|x^k-x^*\|^2 \leq -T_0(\sigma)\mathbf{E}[f(x^k)-f(x^*)] + E(\sigma^2)$$

Finally, averaging over iterations,

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbf{E}\|x^{k+1}-x^*\|^2 - \mathbf{E}\|x^k-x^*\|^2 \leq -\frac{T_0(\sigma)}{K}\sum_{k=0}^{K-1}\mathbf{E}[f(x^k)-f(x^*)] + E(\sigma^2).$$

Using linearity of expectation

$$\frac{1}{K}\mathbf{E}\|x^K-x^*\|^2 - \frac{1}{K}\mathbf{E}\|x^0-x^*\|^2 \leq -T_0(\sigma)\mathbf{E}\left[\sum_{k=0}^{K-1}\frac{1}{K}f(x^k)-f(x^*)\right] + E(\sigma^2).$$

and therefore

$$\mathbf{E}\left[\frac{1}{K}\sum_{k=0}^{K-1}f(x^k)-f(x^*)\right] \leq \frac{1}{T_0(\sigma)K}\mathbf{E}\|x^0-x^*\|^2 + \frac{E(\sigma^2)}{T_0(\sigma)}.$$

Finally, by Jensen's inequality,

$$\mathbf{E}\left[f(\bar{x}^k)-f(x^*)\right] \leq \frac{1}{T_0(\sigma)K}\mathbf{E}\|x^0-x^*\|^2 + \frac{E(\sigma^2)}{T_0(\sigma)},$$

where $\bar{x}^K = \frac{1}{K}\sum_{k=0}^{K-1} x^k$. Finally, recall that $E(\sigma^2) = T_1(\sigma^2)\Delta_{\text{int}} + T_2(\sigma^2)\Delta_{\text{pos}}$ with $T_0(\sigma) = \frac{2\sigma}{(1+2\sigma L)(1+\sigma L)}$, $T_1(\sigma^2) = \frac{6L\sigma^2}{1+2\sigma L}$, and finally $T_2(\sigma^2) = \frac{2\sigma^2 L}{1+\sigma L} \cdot \max\left\{0, \frac{2\sigma L - 1}{2\sigma L + 1}\right\}$. Therefore we can write

$$\begin{aligned}\mathbf{E}\left[f(\bar{x}^k) - f(x^*)\right] \leq \frac{(1+2\sigma L)^2}{2\sigma K}\mathbf{E}\|x^0 - x^*\|^2 \\ + 3\sigma L(1+\sigma L)\Delta_{\text{int}} + \sigma L \cdot \max\{0, 2\sigma L - 1\}\,\Delta_{\text{pos}}.\end{aligned}$$

This concludes the proof.

### D.2.3 *Proof of Theorem 4.3.2 (NGN, Nonconvex)*

The proof deviates significantly from the one for stochastic Polyak step-sizes by Loizou et al., 2021. We start with the classic expansion based on gradient smoothness

$$\begin{aligned}f(x^{k+1}) - f(x^k) &\leq \langle \nabla f(x^k), x^{k+1} - x^k\rangle + \frac{L}{2}\|x^{k+1} - x^k\|^2 \\ &= -\gamma_k \langle \nabla f(x^k), \nabla f_{i_k}(x^k)\rangle + \frac{L\gamma_k^2}{2}\|\nabla f_{i_k}(x^k)\|^2 \\ &\leq -\gamma_k \langle \nabla f(x^k), \nabla f_{i_k}(x^k)\rangle + \frac{L\sigma^2}{2}\|\nabla f_{i_k}(x^k)\|^2.\end{aligned}$$

We would like to take the conditional expectation with respect to $x^k$. Yet, this is not easy since $\gamma_k$ and $\nabla f_{i_k}(x^k)$ are correlated. Note however that we can write, given the bound in Lemma D.2.2,

$$\gamma_k = \frac{\sigma}{\sigma L + 1} + \frac{\sigma^2 L}{\sigma L + 1}\xi_{i_k},$$

where $\xi_{i_k} \in [0,1]$ is a random variable. When $\xi_{i_k} = 1$ then $\gamma_k = \sigma$, when $\xi_{i_k} = 1$ then $\gamma_k = \frac{\sigma}{\sigma L + 1}$. This model for $\gamma_k$ covers its complete range and makes one property explicit : $\gamma_k = O(\sigma)$ with variation range $O(\sigma^2)$.

As such, as $\sigma \to 0$ the stepsize becomes deterministic, and the update reduces to SGD with constant stepsize. Leveraging this representation:

$$\begin{aligned}
&-\gamma_k \langle \nabla f(x^k), \nabla f_{i_k}(x^k) \\
&= -\frac{\sigma}{\sigma L + 1} \langle \nabla f(x^k), \nabla f_{i_k}(x^k) \rangle - \frac{\sigma^2 L}{\sigma L + 1} \xi_{i_k} \langle \nabla f(x^k), \nabla f_{i_k}(x^k) \rangle \\
&\leq -\frac{\sigma}{\sigma L + 1} \langle \nabla f(x^k), \nabla f_{i_k}(x^k) \rangle + \frac{\sigma^2 L}{\sigma L + 1} \left| \xi_{i_k} \right| \cdot \left| \langle \nabla f(x^k), \nabla f_{i_k}(x^k) \rangle \right| \\
&\leq -\frac{\sigma}{\sigma L + 1} \langle \nabla f(x^k), \nabla f_{i_k}(x^k) \rangle + \frac{\sigma^2 L}{\sigma L + 1} \left| \langle \nabla f(x^k), \nabla f_{i_k}(x^k) \rangle \right|.
\end{aligned}$$

Therefore

$$\begin{aligned}
&-\mathbf{E}_k[\gamma_k \langle \nabla f(x^k), \nabla f_{i_k}(x^k) \rangle] \\
&\leq -\frac{\sigma}{\sigma L + 1} \|\nabla f(x^k)\|^2 + \frac{\sigma^2 L}{\sigma L + 1} \mathbf{E}_k \left| \langle \nabla f(x^k), \nabla f_{i_k}(x^k) \rangle \right|.
\end{aligned}$$

The first term in the bound is $O(\sigma)$ and directly helps convergence, while the last term is an error $O(\sigma^2)$. Next, recall the basic inequality:

$$|\langle a, b \rangle| \leq \frac{1}{2}\|a\|^2 + \frac{1}{2}\|b\|^2 + \frac{1}{2}\|a - b\|^2.$$

Applied to our setting, using the assumption $\zeta^2 = \sup_{x \in \mathbb{R}^d} \mathbf{E}_i \|\nabla f(x) - \nabla f_i(x)\|^2 < \infty$, we get

$$\begin{aligned}
&2\mathbf{E}_k \left| \langle \nabla f(x^k), \nabla f_{i_k}(x^k) \rangle \right| \\
&\leq \|\nabla f(x^k)\|^2 + \mathbf{E}_k \|\nabla f_{i_k}(x^k)\|^2 + \mathbf{E}_k \|\nabla f(x^k) - \nabla f_{i_k}(x^k)\|^2 \\
&\leq \|\nabla f(x^k)\|^2 + \mathbf{E}_k \|\nabla f_{i_k}(x^k)\|^2 + \zeta^2 \\
&\leq 2\|\nabla f(x^k)\|^2 + 2\zeta^2
\end{aligned}$$

Therefore, we get the compact inequality

$$\begin{aligned}
&-\mathbf{E}_k[\gamma_k \langle \nabla f(x^k), \nabla f_{i_k}(x^k) \rangle] \\
&\leq -\frac{\sigma}{\sigma L + 1} \|\nabla f(x^k)\|^2 + \frac{\sigma^2 L}{\sigma L + 1} \left( \|\nabla f(x^k)\|^2 + \zeta^2 \right) \\
&\leq -\sigma \left( \frac{1 - \sigma L}{1 + \sigma L} \right) \|\nabla f(x^k)\|^2 + \frac{\sigma^2 L}{\sigma L + 1} \zeta^2,
\end{aligned}$$

that we can insert back in the original expansion to get

$$\begin{aligned}
&\mathbf{E}_k[f(x^{k+1})] - f(x^k) \\
&\leq -\mathbf{E}_k[\gamma_k \langle \nabla f(x^k), \nabla f_{i_k}(x^k)\rangle] + \frac{L\sigma^2}{2}\mathbf{E}_k\|\nabla f_{i_k}(x^k)\|^2 \\
&\leq \left[-\sigma\left(\frac{1-\sigma L}{1+\sigma L}\right) + \frac{L\sigma^2}{2}\right]\|\nabla f(x^k)\|^2 + \left(\frac{\sigma^2 L}{\sigma L + 1} + \frac{\sigma^2 L}{2}\right)\zeta^2.
\end{aligned}$$

We therefore need

$$-\sigma\left(\frac{1-\sigma L}{1+\sigma L}\right) + \frac{L\sigma^2}{2} = -\sigma\left(\frac{1-\sigma L}{1+\sigma L} - \frac{\sigma L}{2}\right) \leq 0$$

The function $\frac{1-\sigma L}{1+\sigma L} - \frac{\sigma L}{2}$ is monotonically decreasing as $\sigma L > 0$ increases. For $\sigma L = 0$ it is 1 and reaches value zero at $-3/2 + \sqrt{17}/2 \approxeq 0.56$. For $\sigma L = 0.5$, one gets $\frac{1-\sigma L}{1+\sigma L} - \frac{\sigma L}{2} = \frac{0.5}{1.5} - 0.25 = 1/12$. Therefore, for $\sigma \leq \frac{1}{2L}$, we get $-\sigma\left(\frac{1-\sigma L}{1+\sigma L}\right) + \frac{L\sigma^2}{2} \leq -\frac{\sigma}{12}$.

Next, for the noise term, note that $\frac{\sigma^2 L}{\sigma L+1} + \frac{\sigma^2 L}{2} \leq \frac{3\sigma^2 L}{2}$. All in all, for $\sigma \leq \frac{1}{2L}$, we get:

$$\mathbf{E}_k[f(x^{k+1})] - f(x^k) \leq -\frac{\sigma}{12}\|\nabla f(x^k)\|^2 + \frac{3\sigma^2 L}{2}\zeta^2,$$

Or, more conveniently:

$$\|\nabla f(x^k)\|^2 \leq -\frac{12}{\sigma}[\mathbf{E}_k[f(x^{k+1})] - f(x^k)] + 18\sigma L\zeta^2,$$

After taking the expectation using the tower property, we get

$$\mathbf{E}\|\nabla f(x^k)\|^2 \leq -\frac{12}{\sigma}[\mathbf{E}[f(x^{k+1})] - \mathbf{E}[f(x^k)]] + 18\sigma L\zeta^2,$$

Summing over iterations and telescoping the sum, after adding and subtracting $f^*$ we get

$$\begin{aligned}\frac{1}{K}\sum_{k=0}^{K-1}\mathbf{E}\|\nabla f(x^k)\|^2 &\leq -\frac{12}{\sigma K}\sum_{k=0}^{K-1}\mathbf{E}[f(x^{k+1})] + \frac{12}{\sigma K}\sum_{k=0}^{K-1}\mathbf{E}[f(x^k)] + 18\sigma L\zeta^2 \\ &\leq -\frac{12}{\sigma K}\mathbf{E}[f(x^K)] + \frac{12}{\sigma K}\mathbf{E}[f(x^0)] + 18\sigma L\zeta^2 \\ &\leq -\frac{12}{\sigma K}\mathbf{E}[f(x^K) - f^*] + \frac{12}{\sigma K}\mathbf{E}[f(x^0) - f^*] + 18\sigma L\zeta^2 \\ &\leq \frac{12}{\sigma K}\mathbf{E}[f(x^0) - f^*] + 18\sigma L\zeta^2.\end{aligned}$$

This concludes the proof.

### D.2.4 *Increasing regularization for convergence*

Let us now consider the setting $\sigma_k \to 0$, which can be considered increasing regularization (i.e. getting more similar to a vanilla SGD update). Recall that indeed that the algorithm we propose stems from the following approximation:

$$\tilde{f}_\sigma(x+p) := \frac{1}{n}\sum_{i=1}^{n}\left(r_i(x) + \nabla r_i(x)^\top p\right)^2 + \frac{1}{2\sigma}|p\|^2.$$

In this section, we analyse NGN with decreasing $\sigma_k$, i.e.

$$\gamma_k = \frac{\sigma_k}{1 + \frac{\sigma_k}{2f_{i_k}(x^k)}\|\nabla f_{i_k}(x^k)\|^2} \qquad \text{(NGN-stochastic-dec)}$$

For a general time-dependent $\sigma$, all local inequalities hold. In particular, the following three lemmata hold with no additional proof required.

**Lemma D.2.4** (Stepsize bounds, time dependent). *Let each $f_i : \mathbb{R}^d \to \mathbb{R}$ be non-negative, differentiable and $L$-smooth.*
*Consider $\gamma_k$ as in NGN-stochastic-dec, we have*

$$\gamma_k \in \left[\frac{\sigma_k}{1+\sigma_k L}, \sigma\right] = \left[\frac{1}{L + \sigma_k^{-1}}, \sigma_k\right].$$

**Lemma D.2.5** (Fundamental Equality, time dependent). *Consider $\gamma_k$ as in NGN-stochastic-dec. One has*

$$\gamma_k \|\nabla f_{i_k}(x)\|^2 = 2\left(\frac{\sigma_k - \gamma_k}{\sigma_k}\right) f_{i_k}(x).$$

**Lemma D.2.6** (Fundamental inequality, time dependent). *Let each $f_i : \mathbb{R}^d \to \mathbb{R}$ be non-negative, L-smooth and convex.*
*Consider $\gamma_k$ as in NGN-stochastic-dec, we have*

$$\begin{aligned}&\gamma_k^2 \|\nabla f_{i_k}(x^k)\|^2 \\ &\leq \left(\frac{4\sigma_k L}{1+2\sigma_k L}\right)\gamma_k[f_{i_k}(x^k) - f_{i_k}^*] + \frac{2\sigma_k^2 L}{1+\sigma_k L}\cdot \max\left\{0, \frac{2\sigma_k L - 1}{2\sigma_k L + 1}\right\}\cdot f_{i_k}^*.\end{aligned}$$

We present proof for the convex setting. The nonconvex is very similar and uses the same techniques; therefore, we omit it.

**Theorem D.2.1** (NGN, convex, decreasing $\sigma$). *Let $f = \frac{1}{N}\sum_{i=1}^N f_i$, where each $f_i : \mathbb{R}^d \to \mathbb{R}$ is non-negative, $L_i$-smooth and convex. Let $L = \max_{i\in[N]} L_i$. Consider stochastic gradient descent with batch size 1 and let $i_k$ be the data index sampled at iteration k. For any value of $\sigma_0 > 0$, NGN-stochastic-dec with $\sigma_k = \sigma_0/\sqrt{k+1}$ leads to the following rate: for $K \geq 2$,*

$$\mathbf{E}[f(\bar{x}^K) - f(x^*)] \leq \frac{C_1\|x^0 - x^*\|^2}{\sqrt{K}-1} + \frac{C_1 C_2 \ln(K+1)}{\sqrt{K}-1} = \mathcal{O}\left(\frac{\ln(K)}{\sqrt{K}}\right).$$

*where $\bar{x}^K = \sum_{k=0}^{K-1} p_{k,K} x^k$ with $p_{k,K} = \frac{\sigma_k}{\sum_{k=0}^{K-1}\sigma_k}$ and*

$$C_1 = \frac{(1+2\sigma_0 L)(1+\sigma_0 L)}{4\sigma_0}, \quad C_2 = \left[6\Delta_{int} + 2\max\{0, 2\sigma_0 L - 1\}\Delta_{pos}\right] L\sigma_0^2.$$

*Proof.* Using Lemma D.2.4,D.2.5&D.2.6 and following the same exact steps as in Theorem 4.3.1, we arrive at

$$\begin{aligned}&\|x^{k+1} - x^*\|^2 - \|x^k - x^*\|^2 \\ &\leq -T_0(\sigma_k)[f_{i_k}(x^k) - f_{i_k}(x^*)] + T_1(\sigma_k^2)[f_{i_k}(x^*) - f_{i_k}^*] + T_2(\sigma_k^2) f_{i_k}^*,\end{aligned}$$

with

$$T_0(\sigma_k) = \frac{2\sigma_k}{(1+2\sigma_k L)(1+\sigma_k L)},$$
$$T_1(\sigma_k^2) = \frac{6L\sigma_k^2}{1+2\sigma_k L},$$
$$T_2(\sigma_k^2) = \frac{2\sigma_k^2 L}{1+\sigma_k L} \cdot \max\left\{0, \frac{2\sigma_k L - 1}{2\sigma_k L + 1}\right\}.$$

Taking the expectation, we get the compact formula

$$T_0(\sigma_k)\mathbf{E}[f(x^k) - f(x^*)] \leq -\mathbf{E}\left[\|x^{k+1} - x^*\|^2 - \|x^k - x^*\|^2\right] + E(\sigma_k^2),$$

where $E_k$ is the expected error at step $k$:

$$\begin{aligned} E(\sigma_k^2) &:= T_1(\sigma_k^2)\mathbf{E}[f_{i_k}(x^*) - f_{i_k}^*] + T_2(\sigma_k^2)\mathbf{E}[f_{i_k}^*] \\ &\leq T_1(\sigma_k^2)\Delta_{\text{int}} + T_2(\sigma_k^2)\Delta_{\text{pos}}, \end{aligned}$$

where $\Delta_{\text{int}} := \mathbf{E}[f_i(x^*) - f_i^*]$ and $\Delta_{\text{pos}} := \mathbf{E}[f_i^*]$.
Following standard techniques (Garrigos and Gower, 2023), summing over $k$ and using telescopic cancellation gives

$$\sum_{k=0}^{K-1} T_0(\sigma_k)\mathbf{E}[f(x^k) - f(x^*)] \leq \|x^0 - x^*\|^2 + \sum_{k=0}^{K-1} E(\sigma_k^2),$$

Let us now construct a pointwise lower bound $\tilde{T}_0(\sigma_k)$ to $T_0(\sigma_k)$, using the fact that $\sigma$ is decreasing:

$$T_0(\sigma_k) = \frac{2\sigma_k}{(1+2\sigma_k L)(1+\sigma_k L)} \geq \frac{2\sigma_k}{(1+2\sigma_0 L)(1+\sigma_0 L)} =: \tilde{T}_0(\sigma_k).$$

We have

$$\sum_{k=0}^{K-1} \tilde{T}_0(\sigma_k)\mathbf{E}[f(x^k) - f(x^*)] \leq \|x^0 - x^*\|^2 + \sum_{k=0}^{K-1} E(\sigma_k^2),$$

Let us divide everything by $\sum_{k=0}^{K-1} \tilde{T}_0(\sigma_k)$:

$$\sum_{k=0}^{K-1} \left(\frac{\tilde{T}_0(\sigma_k)}{\sum_{k=0}^{K-1} \tilde{T}_0(\sigma_k)}\right) \mathbf{E}[f(x^k) - f(x^*)] \leq \frac{\|x^0 - x^*\|^2}{\sum_{k=0}^{K-1} \tilde{T}_0(\sigma_k)} + \frac{\sum_{k=0}^{K-1} E(\sigma_k^2)}{\sum_{k=0}^{K-1} \tilde{T}_0(\sigma_k)}.$$

Using an integral bound, we have, for $K \geq 2$

$$\sum_{k=0}^{K-1} \frac{1}{\sqrt{k+1}} \geq \int_1^K \frac{1}{\sqrt{t}} dt = 2(\sqrt{K}-1).$$

Therefore, we have that

$$\begin{aligned}\sum_{k=0}^{K-1} \tilde{T}_0(\sigma_k) &= \sum_{k=0}^{K-1} \frac{2\sigma_k}{(1+2\sigma_0 L)(1+\sigma_0 L)} \\ &= \frac{2\sigma_0}{(1+2\sigma_0 L)(1+\sigma_0 L)} \sum_{k=0}^{K-1} \frac{1}{\sqrt{k+1}} \\ &\geq \frac{4\sigma_0(\sqrt{K}-1)}{(1+2\sigma_0 L)(1+\sigma_0 L)}.\end{aligned}$$

Note also that

$$\left( \frac{\tilde{T}_0(\sigma_k)}{\sum_{k=0}^{K-1} \tilde{T}_0(\sigma_k)} \right) = \frac{\sigma_k}{\sum_{k=0}^{K-1} \sigma_k} =: p_{k,K},$$

where $p_{k,K}$ as a function of $k$ is a probability distribution over the interval $[0, K-1]$.
We are left with bounding $\sum_{k=0}^{K-1} E_k$. Since $\sigma_k$ is decreasing, we have the following bounds:

$$T_1(\sigma_k^2) = \frac{6L\sigma_k^2}{1+2\sigma_k L} \leq 6L\sigma_k^2,$$

$$T_2(\sigma_k^2) = \frac{2\sigma_k^2 L}{1+\sigma_k L} \cdot \max\left\{0, \frac{2\sigma_k L - 1}{2\sigma_k L + 1}\right\} \leq 2\sigma_k^2 L \cdot \max\left\{0, 2\sigma_0 L - 1\right\}.$$

We next use a standard bound[2] on the $K$-th Harmonic number $H_K$.

$$\sum_{k=0}^{K-1} \frac{1}{k+1} = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{K} = H_K \leq \ln(K+1).$$

Therefore,

$$\sum_{k=0}^{K-1} T_1(\sigma_k^2) \leq 6L \sum_{k=0}^{K-1} \sigma_k^2 = 6L\sigma_0^2 \sum_{k=0}^{K-1} \frac{1}{k+1} \leq 6L\sigma_0^2 \ln(K+1).$$

2 See e.g. https://www.cs.umd.edu/class/spring2016/cmsc351-0101/harmonic.pdf

and

$$\begin{aligned}\sum_{k=0}^{K-1} T_2(\sigma_k^2) &\leq 2L \cdot \max\{0, 2\sigma_0 L - 1\} \sum_{k=0}^{K-1} \sigma_k^2 \\ &\leq 2L \cdot \max\{0, 2\sigma_0 L - 1\} \sigma_0^2 \ln(K+1).\end{aligned}$$

All in all, we have

$$\begin{aligned}\sum_{k=0}^{K-1} E(\sigma_k^2) &\leq \Delta_{\text{int}} \sum_{k=0}^{K-1} T_1(\sigma_k^2) + \Delta_{\text{pos}} \sum_{k=0}^{K-1} T_2(\sigma_k^2) \\ &\leq \left[6\Delta_{\text{int}} + 2\max\{0, 2\sigma_0 L - 1\} \Delta_{\text{pos}}\right] L\sigma_0^2 \ln(K+1).\end{aligned}$$

Plugging everything back into the bound, we have, for $K \geq 2$

$$\sum_{k=0}^{K-1} p_{k,K} \mathbf{E}[f(x^k) - f(x^*)] \leq \frac{C_1 \|x^0 - x^*\|^2}{\sqrt{K} - 1} + \frac{C_1 C_2 \ln(K+1)}{\sqrt{K} - 1}.$$

with

$$C_1 = \frac{(1 + 2\sigma_0 L)(1 + \sigma_0 L)}{4\sigma_0}, \quad C_2 = \left[6\Delta_{\text{int}} + 2\max\{0, 2\sigma_0 L - 1\} \Delta_{\text{pos}}\right] L\sigma_0^2.$$

To conclude, let $\bar{x}^K = \sum_{k=0}^{K-1} p_{k,K} x^k$. Jensen's inequality implies

$$f(\bar{x}^K) = f\left(\sum_{k=0}^{K-1} p_{k,K} x^k\right) \leq \sum_{k=0}^{K-1} p_{k,K} f(x^k).$$

This concludes the proof. □

# E

## APPENDIX TO CHAPTER 5

*After all, time is not a temporal space, but an ordering.*

– Ludwig Wittgenstein.

### E.1 TRAINING SPEEDUPS

In Tb.8, we show training speed comparisons of the LRU with a regular RNN with *tanh* activations, as well as with the S4D and S5 models. As we elaborate in Orvieto et al., 2023c, for the LRU, we closely followed the optimal model sizes of the S5 model. Consequently, we also see similar training speeds as the S5 model on all tasks.

| Model | sCIFAR | ListOps | Text | Retrieval | Pathfinder | PathX |
|---|---|---|---|---|---|---|
| Tanh RNN | 2.0 | 1.1 | 0.5 | 0.5 | 2.1 | 0.14 |
| LRU | 15.9 (8x) | 2.1 (1.9x) | 14.7 (29x) | 5.7 (11.4x) | 15.5 (7.4x) | 2.4 (17x) |
| S5 (our repr.) | 15.9 | 2.2 | 14.4 | 5.7 | 15.6 | 2.3 |
| S4D (our repr.) | 13.5 | 2.2 | 10.6 | 3.0 | 24.5 | 2.6 |

**Table 8:** *Speeds (steps/sec) during training on a A100 GPU. We also show the speedup of the LRU over the tanh RNN for each task. The batch size used for each task is specified in Orvieto et al., 2023c. We compare with our reproduction of S4D/S5.*

### E.2 PARALLEL UNROLLING OF LINEAR RECURRENCES

We provide here a quick example to give an idea of the inner workings of parallel scans. The example is similar to the one presented in Smith et al., 2022. Consider the computation of 4th hidden state of the linear

recurrent model $x_k = Ax_{k-1} + Bu_k$, i.e. $x_3$. We have, starting as usual from $x_{-1} = 0$,

$$
\begin{aligned}
x_0 &= Bu_0, \\
x_1 &= ABu_0 + Bu_1, \\
x_2 &= A^2Bu_0 + ABu_1 + Bu_2, \\
x_3 &= A^3Bu_0 + A^2Bu_1 + ABu_2 + Bu_3.
\end{aligned}
$$

Note that, starting from the sequence of projected inputs $Bu_0$, $Bu_1$, $Bu_2$, to compute $x_3$ sequentially we need three consecutive steps: $x_0 \to x_1, x_1 \to x_2$ and $x_2 \to x_3$. If however we have access to parallel compute, we can reduce the number of steps — in this setting, we can make it in just two steps.

Consider tuples $(A, Bu_0), (A, Bu_1), (A, Bu_2), (A, Bu_3)$ and the operation $\odot$ such that $(a, b) \odot (a', b') = (aa', a'b + b')$. Below, in Figure 71, we illustrate how we can reduce the number of steps from 3 to 2 using the $\odot$ operation. Of course, for longer sequences, the number of operations decreases drastically. In particular, we note that this type of parallelization is particularly effective in the case where $A$ is diagonal (as proposed in this work as well as in Smith et al., 2022), so that no dense matrix multiplication is needed.

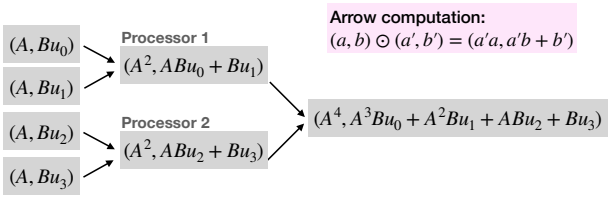


**Figure 71:** *Example of parallel scans on a short sequence.*

### E.3 EXPERIMENTAL DETAILS

We consider the standard S4 architecture of Gu et al., 2021 and replace the S4 layers with RNN layers or with S5 (Smith et al., 2022) or S4D (Gu et al., 2022a) layers for our baselines. We give an overview of the architecture used in Fig.43. The input is first encoded into $H$ features, followed

by a stack of residual blocks. For all our experiments, we use networks with a depth of 6 residual blocks. Each residual block consists of identity skip connection, and the residual path containing a normalization layer (in our case, we always use batch normalization in our experiments), followed by the RNN/SSM block. While using the "post-norm" option of adding the normalization layer after the skip and residual branches typically improves performance, we stick to this design due to this architecture being more scalable in general (De and Smith, 2020).

Each RNN/SSM block first contains the recurrent layer as described in Eqs.(120) and (122) in Section 5.2. This is followed by a mixing layer. For all experiments except PathX, we use the GLU activation function (Dauphin et al., 2017) with dropout as the mixing layer, similar to Gu et al., 2021. For PathX, we instead use a GLU activation function without one additional linear transform; the same as used by Smith et al., 2022 for their experiments.

We use bidirectional models for our experiments on PathFinder and PathX, using a similar setup as Gu et al., 2021, and use unidirectional models for the rest of our experiments. We use AdamW as our optimizer (Loshchilov and Hutter, 2017). We use warmup for the learning rate, where we start from a value of $10^{-7}$ and increase the learning rate linearly up a specified value for the first 10% of training. This is followed by cosine annealing for the rest of training down to a value of $10^{-7}$. We used a smaller learning rate for the RNN/SSM parameters in the recurrent computation (i.e. $A$ (or $\Lambda$) and $B$). When using normalization in our RNNs, we also used a smaller learning rate on the normalization parameter $\gamma$. For our S5 and S4D baselines, we used a smaller learning rate for the discretization step size $\Delta$. This smaller learning rate was determined by multiplying the base learning rate by a factor $< 1$ (See Tb.9 for the learning rate factor used for each task). We use weight decay for all parameters except the RNN/SSM parameters $A$ and $B$ (and $\gamma$ and $\Delta$ when applicable).

All experiments were carried out on accelerated hardware A100 GPUs.

HYPERPARAMETERS. For our S5 (Smith et al., 2022) baseline, we tuned the model dimension $H$ and state dimension $N$. These values were then used for our final LRU configuration without re-tuning. For the S5 baseline, we additionally tuned the number of blocks (denoted $P$ in Smith

| Task | Depth | $H$ | $N$ | Iterations | Batch size | LR factor | Dropout |
|---|---|---|---|---|---|---|---|
| sCIFAR | 6 | 512 | 384 | 180k | 50 | 0.25 | 0.1 |
| ListOps | 6 | 128 | 256 | 80k | 32 | 0.5 | 0.0 |
| Text | 6 | 256 | 192 | 50k | 32 | 0.1 | 0.1 |
| Retrieval | 6 | 128 | 256 | 100k | 64 | 0.5 | 0.1 |
| PathFinder | 6 | 192 | 256 | 500k | 64 | 0.25 | 0.0 |
| PathX | 6 | 128 | 256 | 250k | 32 | 0.25 | 0.0 |

**Table 9:** *List of all the hyperparameters used for each task for the LRU model. Weight decay is set to* 0.05 *for all tasks.*

et al., 2022) as well as the initialization scheme. For our S4D Gu et al., 2022a baseline, we used the same model dimension $H$ as used for our S5 baseline, and additionally tuned the state dimension $N$ and the initialization scheme. For the LRU, we tuned the initialization scheme (set by $r_{\min}$ and $r_{\max}$). Thus, our S4D and S5 baselines were afforded a similar tuning budget as the LRU (arguably, a larger tuning budget than the LRU). This explains why some of the numbers for our baselines are superior to the values reported in the original papers on the same tasks. For all our experiments, we tuned the base learning rate on a logarithmic grid of 2 to choose the optimal learning rate. We present the hyperparameters we used for each LRU experiment in Tb.9. We also note that the number of parameters in the architectures above is similar.

STABILITY. Common hyperparameters such as model width, model depth, and the learning rate, were no more sensitive for the LRU than for our S4/S4D/S5 baselines. The main hyperparameters specific to the LRU are the $r_{\min}$ and $r_{\max}$ hyperparameters governing the initialization scheme. We found that these could simply be set to default values of $r_{\min} = 0$ and $r_{\max} = 1$ for most tasks, but needed to be tuned to achieve strong performance on Pathfinder and PathX, the two most challenging tasks in the LRA benchmark.

# F

## APPENDIX TO CHAPTER 6

*Everything should be made as simple as possible, but not simpler.*
– Albert Einstein.

In this appendix, we provide the full proof concerning the behavior of Anti-PGD on widening valleys — our **toy landscape**. This is perhaps the most **unconventional** proof of this thesis. In the interest of space, all the other proofs for this chapter can be found in the corresponding papers (Lucchi et al., 2022; Orvieto et al., 2022a; Orvieto et al., 2023b).

### F.1 PROOF OF THEOREM 6.3.2

To prove Theorem 6.3.2, we first need some preliminary preparation; in doing so, we will also provide some intuition for the reader. The main proof follows afterwards, in Section F.1.2.
Consider the problem of minimizing the cost function

$$L(u,v) = \frac{1}{2}v^2\|u\|^2,$$

where $\|\cdot\|$ is the Euclidean norm, $v \in \mathbb{R}$, and $u \in \mathbb{R}^d$. To minimize the loss, we use PGD or Anti-PGD (see Table 6). Note that any point where $v = 0$ or $\|u\|^2 = 0$ minimizes the loss (see Fig. 56). By the considerations in the section above, we want to find a solution $(u,v)$ where $\|u\|$ is small (i.e. a solution with low curvature). We show that, while standard noise injection does not necessarily induce this bias on the dynamics, injection of anticorrelated noise does. This show that anticorrelated noise effectively minimizes the trace of the Hessian:

$$Tr(\nabla^2 L(u,v)) = dv^2 + \|u\|^2.$$

PRELIMINARY CONSIDERATIONS. Let us start by writing down the update in discrete-time. Recall that the gradient is $(v^2 u, \|u\|^2 v)$, hence:

$$u_{k+1} = (1 - \eta v_k^2) \cdot u_k + \varepsilon_k^u \tag{207}$$

$$v_{k+1} = (1 - \eta \|u_k\|^2) \cdot v_k + \varepsilon_k^v \tag{208}$$

where $\varepsilon_k^u \in \mathbb{R}^d$ and $\varepsilon_k^v \in \mathbb{R}$ are the noise variables.

1. For stability (in the noiseless setting), we need $\eta \leq \frac{2}{\max\{v_k^2, \|u_k\|^2\}}$.

2. Starting from a big $\|u\|$ and any $v$, under noiseless GD, since $d \gg 1$, we converge to $(u_0, 0)$, with $\|u_0\| := D \gg 1$.

The key to the proof of effectiveness of anticorrelated noise, compared to uncorrelated noise, relies on the following observation:

**Empirical Observation:** for the widening valley $L(u, v) = \frac{1}{2} v^2 \|u\|^2$, if we only perturb the $v$ coordinate with *any noise* then we get to a wide minimum.

**Why? Intuition behind the proof.** Well, of course this is the case! If one knows is advance which direction to move in order to pick up a signal, then *les jeux sont faits* (since I did this while interning with Francis Bach in France). The problem is that in order to perturb this direction — we need to perturb *all directions*, and this leads to "getting lost" if the noise is not controlled (i.e. does not have an attraction force to the origin). This also motivates why the effect gets more intense as the dimension $d$ increases: there is a lot of bias added, which drives us away from good minima.

**Plan:** To show the result, we follow the following procedure:

1. Starting from $(u_0, 0)$, we start injecting noise and want to reach $(\tilde{u}, \tilde{v})$ such that $\|\tilde{u}\|^2 = \alpha D$ and $0 < \alpha < 1$. We want to show here a difference in behavior under different noise correlation.

2. We proceed by contradiction: starting from $(\tilde{u}, \tilde{v})$, we assume that $\|u_k\|^2 \geq \alpha D$ for all $k \geq 0$ ($\alpha \in (0, 1)$). Under injection of anticorrelated noise, we show that *this leads to a contradiction* — i.e.

that the dynamics substantially decreases the trace of the Hessian: $\|u_\infty\|^2 < \alpha D$ (worst-case upper bound). Crucially, we also show that the hypothesis does not lead to any contradiction under standard noise injection — i.e. without anticorrelation we do not significantly decrease the trace of the Hessian. More specifically, we show that $\lim_{n\to\infty} \mathbb{E}\left[\|u_n\|^2\right] \geq D/\alpha$ under uncorrelated noise injection (worst-case lower bound).

3. To simplify the computations, assume that coordinate-wise the noise is a result of a Bernoulli$(1/2)$ perturbation $(\xi_k)_i \in \{-\sigma, \sigma\}$. The injected noise is then either $\varepsilon_k = \xi_k$ or $\varepsilon_k = \xi_k - \xi_{k-1}$, for PGD and Anti-PGD respectively.

### F.1.1 *Some Useful Lemmata*

This section is pretty technical, hence the reader can skip the proof on a first read. *The meaning behind the bounds we derive and a numerical verification can be found in Figure 72.*
We start by recalling the variation of constants formula, which we will heavily use along the proof.

**Lemma F.1.1** (Variations of constants formula)**.** *Let $w \in \mathbb{R}^d$ evolve with time-varying linear dynamics $w_{k+1} = A_k w_k + \varepsilon_k$, where $A_k \in \mathbb{R}^{d\times d}$ and $\varepsilon_k \in \mathbb{R}^d$ for all $k$. Then, with the convention that $\prod_{j=k+1}^{k} A_j = 1$,*

$$w_{k+1} = \left(\prod_{j=0}^{k} A_j\right) w_0 + \sum_{i=0}^{k} \left(\prod_{j=i+1}^{k} A_j\right) \varepsilon_i.$$

*Proof.* For $k = 1$ we get $w_1 = A_0 w_0 + \varepsilon_0$. The induction step yields

$$\begin{aligned} w_{k+1} &= A_k \left(\left(\prod_{j=0}^{k-1} A_j\right) w_0 + \sum_{i=0}^{k-1} \left(\prod_{j=i+1}^{k-1} A_j\right) \varepsilon_i\right) + \varepsilon_k \\ &= \left(\prod_{j=0}^{k} A_j\right) w_0 + \sum_{i=0}^{k-1} \left(\prod_{j=i+1}^{k} A_j\right) \varepsilon_i + \left(\prod_{j=k+1}^{k} A_j\right) \varepsilon_k \\ &= \left(\prod_{j=0}^{k} A_j\right) w_0 + \sum_{i=0}^{k} \left(\prod_{j=i+1}^{k} A_j\right) \varepsilon_i. \end{aligned}$$

This completes the proof of the variations of constants formula. □

We extend this formula to the anticorrelated case, where $\varepsilon_k$ has some additional structure.

**Corollary F.1.1** (Anticorrelated variations of constants formula)**.** *Under the same setting of Lemma F.1.1, if there exist a family of vectors $\{\xi_k\}$ such that $\varepsilon_0 = \xi_0$ and $\varepsilon_k = \xi_k - \xi_{k-1}$, then*

$$w_{k+1} = \left(\prod_{j=0}^{k} A_j\right) w_0 + \xi_k + \sum_{i=0}^{k-1} (A_{i+1} - I) \left(\prod_{j=i+2}^{k} A_j\right) \xi_i.$$

*Proof.* We have, by direct computation:

$$\begin{aligned}
w_{k+1} &= \left(\prod_{j=0}^{k} A_j\right) w_0 + \left(\prod_{j=1}^{k} A_j\right) \xi_0 + \sum_{i=1}^{k} \left(\prod_{j=i+1}^{k} A_j\right) \xi_i - \sum_{i=1}^{k} \left(\prod_{j=i+1}^{k} A_j\right) \xi_{i-1} \\
&= \left(\prod_{j=0}^{k} A_j\right) w_0 + \left(\prod_{j=1}^{k} A_j\right) \xi_0 + \sum_{i=1}^{k-1} \left(\prod_{j=i+1}^{k} A_j\right) \xi_i + \xi_k - \sum_{i=0}^{k-1} \left(\prod_{j=i+2}^{k} A_j\right) \xi_i \\
&= \left(\prod_{j=0}^{k} A_j\right) w_0 + \xi_k + \sum_{i=0}^{k-1} \left(\prod_{j=i+1}^{k} A_j\right) \xi_i - \sum_{i=0}^{k-1} \left(\prod_{j=i+2}^{k} A_j\right) \xi_i \\
&= \left(\prod_{j=0}^{k} A_j\right) w_0 + \xi_k + \sum_{i=0}^{k-1} \left[\left(\prod_{j=i+1}^{k} A_j\right) - \left(\prod_{j=i+2}^{k} A_j\right)\right] \xi_i.
\end{aligned}$$

This concludes the proof. □

**Remark F.1.1.** *If $A_i = I$ for all $i$, then the last summand is zero. This showcases the effect of anticorrelation: noise cancellation under noise accumulation.*

EXPECTATION QUANTITIES UNDER DETERMINISTIC $\rho_k$ Using the variation of constants formula, we can write the dynamics of the second moment of stochastic linear time-varying dynamical systems, with either standard or anticorrelated noise.

**Proposition F.1.1** (An Itô-like formula)**.** *Let $w \in \mathbb{R}^d$ evolve with time-varying linear dynamics $w_{k+1} = A_k w_k + \varepsilon_k$, where $A_k \in \mathbb{R}^{d\times d}$ and $\varepsilon_k \in \mathbb{R}^d$ for all $k$. Let $\{\xi_k\}$ be a family of uncorrelated zero-mean $d$-dimensional random variables with variance $\mathbb{E}[\|\xi_k\|^2] = d\sigma^2$ (dependency on the dimension because*

*additivity of squared norm). Consider* $\varepsilon_0 = \xi_0$ *and* $\varepsilon_k = \xi_k - \xi_{k-1}$ *for all* $k \geq 1$. *Further, assume that* $A_k = \rho_k I$ *for all* $k$ *(i.e.* $A_k$ *is a multiple of the identity), with* $\rho_k \in \mathbb{R}$ *a deterministic quantity. Then, with the convention that* $\prod_{j=k+1}^{k} A_j = 1$, *we have*

$$\mathbb{E}[\|w_{k+1}\|^2] = \left(\prod_{j=0}^{k} \rho_j^2\right) \|w_0\|^2 + \left(1 + \sum_{i=0}^{k-1} \left[(1-\rho_{i+1})^2 \prod_{j=i+2}^{k} \rho_j^2\right]\right) d\sigma^2.$$

*Instead, if* $\varepsilon_k = \xi_k$ *for all* $k$ *(standard noise injection) we have*

$$\mathbb{E}[\|w_{k+1}\|^2] = \left(\prod_{j=0}^{k} \rho_j^2\right) \|w_0\|^2 + \sum_{i=0}^{k} \left(\prod_{j=i+1}^{k} \rho_j^2\right) d\sigma^2.$$

*Proof.* Using independence of the $\{\xi_k\}$ family, we obtain for the anticorrelated case:

$$\begin{aligned}
\mathbb{E}[w_{k+1}^\top w_{k+1}] &= \left(\prod_{j=0}^{k} \rho_j\right)^2 \|w_0\|^2 + \mathbb{E}[\|\xi_k\|^2] \\
&\quad + \sum_{i=0}^{k-1} (\rho_{i+1} - 1)^2 \left(\prod_{j=i+2}^{k} \rho_j\right)^2 \mathbb{E}[\|\xi_i\|^2] \\
&= \left(\prod_{j=0}^{k} \rho_j\right)^2 \|w_0\|^2 + \sigma^2 + \sum_{i=0}^{k-1} (\rho_{i+1} - 1)^2 \left(\prod_{j=i+2}^{k} \rho_j^2\right) \sigma^2,
\end{aligned}$$

where we used the fact that the $\xi_k$ are not correlated. The case $\varepsilon_k = \xi_k$ is similar and therefore left to the reader. □

**Corollary F.1.2.** *In the setting of Proposition F.1.1, assume* $\rho_j = \rho \in (0,1)$ *is constant for all* $j$. *If* $\varepsilon_0 = \xi_0$ *and* $\varepsilon_k = \xi_k - \xi_{k-1}$ *for all* $k \geq 1$ *then*

$$\begin{aligned}
\mathbb{E}[\|w_{k+1}\|^2] &= \rho^{2(k+1)} \|w_0\|^2 + \left(1 + \frac{(1-\rho)^2(1-\rho^{2(k+1)})}{1-\rho^2}\right) d\sigma^2 \\
&\xrightarrow{\infty} \quad \frac{2}{1+\rho} d\sigma^2.
\end{aligned}$$

*Instead, if $\varepsilon_k = \xi_k$ for all $k$ (standard noise injection) we have*

$$\mathbb{E}[\|w_{k+1}\|^2] = \rho^{2(k+1)}\|w_0\|^2 + \frac{1-\rho^{2(k+1)}}{1-\rho^2} d\sigma^2 \quad \xrightarrow{\infty} \quad \frac{1}{1-\rho^2} d\sigma^2.$$

*Proof.* Simple application of the formula for geometric series. Numerical verification in Figure 72. □

**Remark F.1.2.** *Note that the corollary has a clear interpretation: if $\rho$ is between zero and one, we experience striking difference between uncorrelated and anticorrelated noise. If $\rho$ increases, the total accumulated anticorrelated noise decreases.[1] This trend is reversed for normal noise injection: as $\rho \to 1$ the total accumulated variance explodes. Numerical verification can be found in Figure 72.*

DEALING WITH POTENTIAL STOCHASTICITY IN THE $\rho_k$. For the proof in the next subsection, we must deal with stochastic $\rho_k$, which are only specified up to an interval.

**Proposition F.1.2** (Limit bound on second moment for anticorrelated noise)**.** *Let $w \in \mathbb{R}^d$ evolve with time-varying linear dynamics $w_{k+1} = A_k w_k + \varepsilon_k$, where $A_k \in \mathbb{R}^{d\times d}$ and $\varepsilon_k \in \mathbb{R}^d$ for all $k$. Let $\{\xi_k\}$ be a family of uncorrelated zero-mean $d$-dimensional random variables with variance $\mathbb{E}[\|\xi_k\|^2] = d\sigma^2$ (dependency on the dimension because additivity of squared norm). Consider $\varepsilon_0 = \xi_0$ and $\varepsilon_k = \xi_k - \xi_{k-1}$ for all $k \geq 1$. Further, assume that $A_k = \rho_k I$ for all $k$ (i.e. $A_k$ is a multiple of the identity) and that $\rho_k \in [0,1]$ for all $k$. Assume that the probability of $\rho_k < 1$ is non-zero, i.e. that $\rho_k \neq 1$ with non-vanishing probability. Then, we have*

$$\lim_{k\to\infty} \mathbb{E}[\|w_{k+1}\|^2] \leq 2d\sigma^2.$$

*Proof.* The proof is based on an induction argument, starting from the equation in Proposition F.1.1:

$$\mathbb{E}[\|w_{k+1}\|^2] = \left(\prod_{j=0}^{k} \rho_j^2\right)\|w_0\|^2 + \left(1 + \sum_{i=0}^{k-1}\left[(1-\rho_{i+1})^2 \prod_{j=i+2}^{k} \rho_j^2\right]\right) d\sigma^2.$$

1 $\frac{2}{1+\rho}$ is a decreasing function or $\rho$, while $1/(1-\rho^2)$ is increasing.

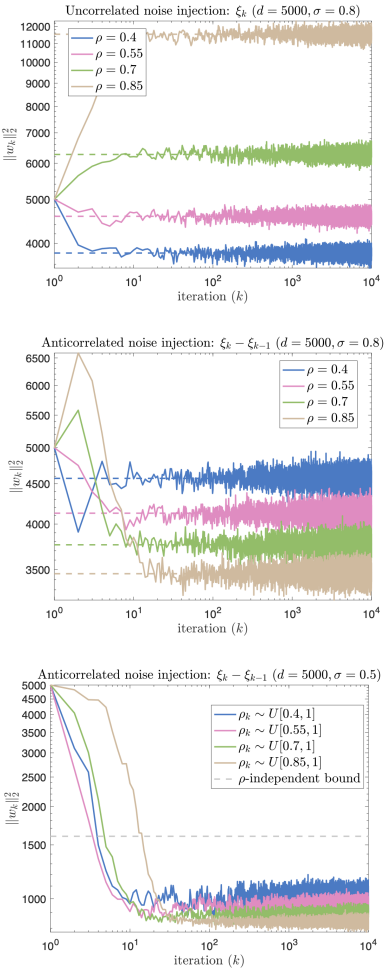


**Figure 72:** *Numerical verification of our final result that we will use in the proof, i.e. Corollary F.1.2 (first and second panel) and Proposition F.1.2 (last panel). Dashed lines indicate our predicted value (in expectation) by the theory. In the right-most plot, we sample $\rho_k$ at each iteration uniformly on an interval.*

First, note that by assumption on $\rho_k$ the first term vanishes as $k \to \infty$. We just have to deal with the second term. Specifically, we want to show that for whatever sequence $\rho_k \in (0,1)$ we have

$$\nu_k = \sum_{i=0}^{k-1} (1-\rho_{i+1})^2 \left( \prod_{j=i+2}^{k} \rho_j^2 \right) \leq 1, \quad \forall k \geq 0.$$

A fundamental observation, is that the term can be written in a recursive form. Indeed,

$$\begin{aligned}
\nu_k &= \sum_{i=0}^{k-1} \left[ (1-\rho_{i+1})^2 \left( \prod_{j=i+2}^{k} \rho_j^2 \right) \right] \\
&= \sum_{i=0}^{k-2} \left[ (1-\rho_{i+1})^2 \left( \prod_{j=i+2}^{k} \rho_j^2 \right) \right] + (1-\rho_k)^2 \\
&= \rho_k \sum_{i=0}^{k-2} \left[ (1-\rho_{i+1})^2 \left( \prod_{j=i+2}^{k-1} \rho_j^2 \right) \right] + (1-\rho_k)^2 \\
&= \rho_k^2 \nu_{k-1} + (1-\rho_k)^2,
\end{aligned}$$

where the second equality follows from the fact that, as previously noted, our notation implies $\prod_{j=k+1}^{k} \rho_k^2 = 1$, for all $k$. Let us now proceed again by induction to show that $v_k \in (0,1)$ for all $k \geq 0$. Note that trivially $v_0 = 0$. Let's proceed with the inductive step:

$$\nu_k = \rho_k^2 \nu_{k-1} + (1-\rho_k)^2 = (\nu_{k-1}+1)\rho_k^2 - 2\rho_k + 1.$$

This quantity is less then one if and only if

$$(\nu_{k-1}+1)\rho_k^2 \leq 2\rho_k.$$

Note that this is satisfied since $\nu_{k-1}+1 \leq 2$, and $\rho_k^2 \leq \rho_k$ since $\rho_k \in (0,1)$. The result follows. □

A numerical verification of this result can be found in Figure 72.

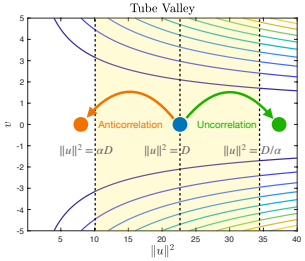


**Figure 73:** *The sketch illustrates the intuition behind the result in Theorem 6.3.2.*

### F.1.2 *Proof of the Main Result*

Using the results from the last subsection, we are now ready to show the main theorem for optimization of the widening valley.

*Proof of Theorem 6.3.2.* As above, we denote the perturbations by $\varepsilon_k$; i.e., $\varepsilon_k = \xi_k$ for PGD and $\varepsilon_k = \xi_{k+1} - \xi_k$ for Anti-PGD. Let us start by inspecting the equation

$$u_{k+1} = (1 - \eta v_k^2) \cdot u_k + \varepsilon_k^u,$$

where $\varepsilon_k^u \in \mathbb{R}^d$ is the projection of the noise $\varepsilon_k$ to the first $d$ coordinates. It is clear that the optimal strategy of making $\|w\|$ small is to increase $|v|$, so to sample nearby points and pick up the gradient. The greater $v$ is in norm, the better. We can increase the norm of $v$ by heavy noise injection (second equation). However, too much noise also increases $\varepsilon_k^u$, which acts adversarially to the decrease of $\|w\|$ (error accumulation increases the Euclidean norm in expectation).

CHOICE OF STEPSIZE AND OPERATING REGION. We start by motivating the choice of stepsize $\eta = \frac{\alpha}{2D}$. Starting from the point $(u_0, 0)$ with $\|u_0\|^2 = D > 0$, we consider the operating landscape region $\alpha D < \|u_k\|^2 < D/\alpha$, with $\alpha \in (0,1)$. We want to show that while standard noise injection makes the process exit the region from the right ($D/\alpha$ side, see Figure 73), anticorrelated noise injection makes the process exit the region from the left ($\alpha D$ side). In this region, named $\mathcal{D}_\alpha$, the maximal allowed learning rate is $\eta \leq \frac{2}{\max_{\mathcal{D}_\alpha}\{v^2, \|u\|^2\}}$. Since $v$ stays small (we are going to check this later in great detail), we select the stepsize $\eta \leq$

$\frac{1}{2\max_{\mathcal{D}_\alpha}\|u\|^2} = \frac{\alpha}{2D}$ — which guarantees stability in expectaction, i.e. without noise injection (even allowing for some slack).

LOWER BOUND FOR UNCORRELATED NOISE ($\varepsilon_k = \xi_k$). For this case, we have to show that standard noise injection cannot possibly work for reaching $\alpha D$, therefore we have to put ourselves in *the best case scenario* for PGD: that is, we have to provide an uniform upper bound for $v_k$ under the second equation (i.e. the equation for $v$) and show that this is not enough for a substantial decrease in $\|u\|$. In the next paragraph (anticorrelated noise), we instead have to put ourselves in the *worst case scenario* — i.e. a lower bound for $|v|$— and show that this is still enough for anticorrelated noise to yield a substantial decrease in $\|u\|$.
To start, let us then look at the second equation:

$$v_{k+1} = (1 - \eta\|u_k\|^2) \cdot v_k + \varepsilon_k^v,$$

where $\varepsilon_k^v$ is the $(d+1)$-th component of $\varepsilon_k$. Since we start from $v_0 = 0$, the equation is completely dominated by noise, and is strongly mean reverting (i.e. $v$ is effectively bounded). Indeed, since $\eta = \frac{\alpha}{2D}$ and $\|u_k\|^2 \in (\alpha D, D/\alpha)$ by assumption, we have

$$|v_{k+1}| \leq \max\left\{1 - \frac{\alpha^2}{2}, \frac{1}{2}\right\} \cdot |v_k| + \sigma = \left(1 - \frac{\alpha^2}{2}\right)|v_k| + \sigma.$$

where we used the fact that $|\varepsilon_k^v| = \sigma$ and that $\alpha^2 \in (0,1)$. By induction, the last inequality yields that, starting from $v_0 = 0$, we have

$$|v_k| \leq v_{\max} := \frac{2\sigma}{\alpha^2}, \quad \forall k \geq 0.$$

Hence, we found the "best case scenario" for the $w$ equation: $|v_k| = \frac{2\sigma}{\alpha^2}$, for all $k$. This gives $w$ the best decrease rate possible.[2] However, we need to check this value $v_{\max}$ is such that the equation for $w$ is indeed stable (we promised this to the reader in the last paragraph). We recall that this equation is $w_{k+1} = (1 - \eta v_k^2) \cdot w_k + \varepsilon_k^u$. Let us require $(1 - \eta v_k^2) \in (0,1)$, for this we need $1/v_{\max}^2 > \eta = \frac{\alpha}{2D}$. Therefore, we need

$$\frac{1}{v_{\max}^2} = \frac{\alpha^4}{4\sigma^2} \geq \eta = \frac{\alpha}{2D} \quad \Longrightarrow \quad \sigma^2 \leq \alpha^3 D/2.$$

2 Note that noise injection in $u$ is independent of $v$, therefore to minimize $\|u\|$ we need the shrinking factor to be as large as possible. We note that using this bound is precise: with probability one we are in the best scenario (we are finding a lower bound).

This is guaranteed by assumption. To proceed, we substitute $v_{\max}$ into the first equation to get

$$u_{k+1} = (1 - \eta \frac{4\sigma^2}{\alpha^4}) \cdot u_k + \varepsilon_k^u = \frac{\alpha^3 D - 2\sigma^2}{\alpha^3 D} u_k + \varepsilon_k^u.$$

Let us call $\rho := \left(\frac{\alpha^3 D - 2\sigma^2}{\alpha^3 D}\right) \in (0, 1)$ the (best case) shrinking factor. Since $\varepsilon_i^u$ is zero-mean, computing the expected value of $\|u_k\|^2$ leads to the following limit by Corollary F.1.2:

$$\lim_{k\to\infty} \mathbb{E}[\|u_k\|^2] = \frac{d\sigma^2}{1-\rho^2} = \frac{dD^2\alpha^6}{2D\alpha^3 - 2\sigma^2}.$$

where we assumed $\sigma^2$ strictly positive. Note that this limit is a monotonically increasing function of $\sigma^2 \in (0, D\alpha^3/2)$. Hence, we get

$$\lim_{k\to\infty} \mathbb{E}[\|u_k\|^2] \in \left(\frac{dD\alpha^3}{2}, dD\alpha^3\right). \tag{209}$$

**Remark F.1.3** (Phase transition)**.** *Note that, for $\sigma$ exactly 0, the limit is instead $\|u_0\|^2 = D$. Instead, for any small noise the process will grow up until at least $dD\alpha^2/2$. This might seem weird at first — but recall that there is an interaction between noise scale and our best-case scenario bound for $v$: they both depend on $\sigma$. This causes a cancellation effect and a transition in behavior at $\sigma = 0$.*

Last, we need to show that this lower bound on $\lim_{k\to\infty} \mathbb{E}[\|u_k\|^2]$ coincides with (or is bigger than) the right boundary of the operating region in Figure 73. To do this we set:

$$\frac{dD\alpha^3}{2} \geq \frac{D}{\alpha} \quad \Longrightarrow \quad d \geq \frac{2}{\alpha^4}.$$

This concludes the proof of Eq. (159).

UPPER BOUND FOR ANTICORRELATED NOISE ($\varepsilon_k = \xi_k - \xi_{k-1}$). We consider anticorrelated noise injection $(\xi_k)_i \in \{-\sigma, \sigma\}$, and $\varepsilon_k = (\varepsilon_k^u, \varepsilon_k^v) = \xi_k - \xi_{k-1}$. Again, let us first look at the second equation:

$$v_{k+1} = (1 - \eta\|u_k\|^2) \cdot v_k + \varepsilon_k^v.$$

Since by hypothesis $\eta = \frac{\alpha}{2D}$ and $\alpha D \leq \|u_k\|^2 \leq D/\alpha$, we have $(1 - \eta\|u_k\|^2) \in \left(1 - \frac{\alpha^2}{2}, \frac{1}{2}\right)$. Clearly, we have that, for any $k \geq 1$ $v_k^2$, is non-zero with a non-vanishing probability. For (noiseless) stability, we also need an upper bound on $|v_k|$. An easy (yet absolutely not tight) upper bound is the following:

$$|v_{k+1}| \leq \frac{1}{2}|v_k| + 2\sigma\varepsilon_k^v. \tag{210}$$

where we simply used the absolute value subadditivity and the fact that $|\varepsilon_k^v| \leq |\xi_k| + |\xi_{k-1}| = 2\sigma$. Note that the equation directly yields by induction $|v_k| \leq 4\sigma$ for all $k \geq 0$.
Let us now deal with the equation for $w$.

$$u_{k+1} = (1 - \eta v_k^2) \cdot u_k + \varepsilon_k^u$$

For this equation, we would want all the coefficients $\rho_k := 1 - \eta v_k^2$ to be between 0 and 1 — i.e. we need to check that $v$ is indeed not too big. Since $|v_k| \leq 4\sigma$ for all $k \geq 0$, we have the requirement $1 - \frac{\alpha}{2D}16\sigma^2 > 0$, which implies $\sigma^2 \leq \frac{D}{8\alpha}$ — that satisfies our hypothesis.
So, to sum it up, we are in operating regime of Proposition F.1.2: anti-correlated noise, $\rho_k < 1$ with non-vanishing probability and $\rho_k$ always between 0 and 1. Hence, we get that

$$\lim_{k\to\infty} \mathbb{E}[\|u_k\|^2] \leq 2\sigma^2 d.$$

Hence, for $\sigma^2$ small enough, the value $\alpha D$ is reached. This directly implies the missing Eq. (160). The proof is thereby complete. □

### F.1.3 *Proof of Corollary 6.3.1*

*Proof.* Recall from Eq. (158) that $Tr(\nabla^2 L(u, v)) = dv^2 + \|u\|^2$.
For the first two inequalities in the corollary, recall from Eq. (210) of the proof of Theorem 6.3.2 that $|v_n| \leq 4\sigma$ for all $n$, almost surely. The first two inequalities in the Corollary follow now by Eq. (160).
For the last two inequalities in the Corollary, we lower bound the trace by $\|u\|^2$ and make use of Eq. (159). □

# BIBLIOGRAPHY


Ahn, Kwangjun and Suvrit Sra (2022). "Understanding Nesterov's acceleration via proximal point method." In: *Symposium on Simplicity in Algorithms (SOSA)*. SIAM, pp. 117–130.

Alimisis, Foivos, Antonio Orvieto, Gary Bécigneul, and Aurelien Lucchi (2020). "A continuous-time perspective for modeling acceleration in Riemannian optimization." In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 1297–1307.

Alimisis, Foivos, Antonio Orvieto, Gary Becigneul, and Aurelien Lucchi (2021). "Momentum improves optimization on Riemannian manifolds." In: *International conference on artificial intelligence and statistics*. PMLR, pp. 1351–1359.

Allen-Zhu, Zeyuan (2017). "Katyusha: The first direct acceleration of stochastic gradient methods." In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, pp. 1200–1205.

Allen-Zhu, Zeyuan, Yuanzhi Li, and Zhao Song (2019). "A convergence theory for deep learning via over-parameterization." In: *International Conference on Machine Learning*. PMLR, pp. 242–252.

Allen-Zhu, Zeyuan and Lorenzo Orecchia (2017). "Linear coupling: An ultimate unification of gradient and mirror descent." In: *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Amari, Shun-Ichi (1998). "Natural gradient works efficiently in learning." In: *Neural computation* 10.2, pp. 251–276.

Amari, Shun-ichi (2016). *Information geometry and its applications*. Vol. 194. Springer.

Andriushchenko, Maksym, Francesco Croce, Maximilian Müller, Matthias Hein, and Nicolas Flammarion (2023). "A modern look at the relationship between sharpness and generalization." In: *arXiv preprint arXiv:2302.07011*.

Anosov, Dmitry Victorovich (1967). "Geodesic flows on closed Riemannian manifolds of negative curvature." In: *Trudy Matematicheskogo Instituta Imeni VA Steklova* 90, pp. 3–210.

Araújo, Vitor and Marcelo Viana (2009). *Hyperbolic dynamical systems*. Springer.

Arjevani, Yossi, Yair Carmon, John C Duchi, Dylan J Foster, Nathan Srebro, and Blake Woodworth (2023). "Lower bounds for non-convex stochastic optimization." In: *Mathematical Programming* 199.1-2, pp. 165–214.

Arjovsky, Martin, Amar Shah, and Yoshua Bengio (2016). "Unitary evolution recurrent neural networks." In: *International Conference on Machine Learning*. PMLR.

Arnol'd, Vladimir Igorevich (2013). *Mathematical methods of classical mechanics*. Vol. 60. Springer Science & Business Media.

Arora, Sanjeev, Noah Golowich, Nadav Cohen, and Wei Hu (2019). "A convergence analysis of gradient descent for deep linear neural networks." In: *International Conference on Learning Representations*.

Arpit, Devansh, Víctor Campos, and Yoshua Bengio (2019). "How to initialize your network? robust initialization for weightnorm & resnets." In: *Advances in Neural Information Processing Systems* 32.

Arpit, Devansh, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. (2017). "A closer look at memorization in deep networks." In: *International Conference on Machine Learning*. PMLR, pp. 233–242.

Asimov, Daniel (1993). *Notes on the topology of vector fields and flows*. Tech. rep. Technical report, NASA Ames Research Center, 1993. RNR-93-003.

Aulbach, Bernd and Fritz Colonius (1996). *Six lectures on dynamical systems*. World Scientific.

Axler, Sheldon (1997). *Linear algebra done right*. Springer Science & Business Media.

Ayache, Antoine and Yimin Xiao (2005). "Asymptotic properties and Hausdorff dimensions of fractional Brownian sheets." In: *Journal of Fourier Analysis and Applications* 11, pp. 407–439.

Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E Hinton (2016). "Layer normalization." In: *arXiv preprint arXiv:1607.06450*.

Bachlechner, Thomas, Bodhisattwa Prasad Majumder, Henry Mao, Gary Cottrell, and Julian McAuley (2021). "Rezero is all you need: Fast convergence at large depth." In: *Uncertainty in Artificial Intelligence*. PMLR, pp. 1352–1361.

Bachmann, Gregor, Sotiris Anagnostidis, and Thomas Hofmann (2023). "Scaling MLPs: A tale of inductive bias." In: *arXiv preprint arXiv:2306.13575*.

Baevski, Alexei, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli (2020). "wav2vec 2.0: A framework for self-supervised learning of speech representations." In: *Advances in Neural Information Processing Systems* 33.

Baldi, Pierre F. and Kurt Hornik (1995). "Learning in linear neural networks: A survey." In: *IEEE Transactions on Neural Networks* 6.4, pp. 837–858.

Balles, Lukas and Philipp Hennig (2018). "Dissecting Adam: The sign, magnitude and variance of stochastic gradients." In: *International Conference on Machine Learning*. PMLR, pp. 404–413.

Barbaro, Vanessa, Antonio Orvieto, Gualtiero Alvisi, Marina Bertolin, Filippo Bonelli, Thomas Liehr, Tigran Harutyunyan, Stefanie Kankel, Gordana Joksic, Stefano Ferrari, et al. (2022). "Analysis and pharmacological modulation of senescence in human epithelial stem cells." In: *Journal of Cellular and Molecular Medicine* 26.14, pp. 3977–3994.

Benettin, Giancarlo and Antonio Giorgilli (1994). "On the Hamiltonian interpolation of near-to-the identity symplectic mappings with application to symplectic integration algorithms." In: *Journal of Statistical Physics* 74, pp. 1117–1143.

Bengio, Y, Patrice Simard, and Paolo Frasconi (1994). "Learning long-term dependencies with gradient descent is difficult." In: *IEEE transactions on neural networks*.

Beyn, W-J (1987). "On the numerical approximation of phase portraits near stationary points." In: *SIAM Journal on Numerical Analysis* 24.5, pp. 1095–1113.

Biggio, Luca, Tommaso Bendinelli, Alexander Neitz, Aurelien Lucchi, and Giambattista Parascandolo (2021). "Neural symbolic regression that scales." In: *International Conference on Machine Learning*. PMLR.

Bisla, Devansh, Jing Wang, and Anna Choromanska (2022). "Low-pass filtering SGD for recovering flat optima in the deep learning optimization landscape." In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 8299–8339.

Blanc, Guy, Neha Gupta, Gregory Valiant, and Paul Valiant (2020). "Implicit regularization for deep neural networks driven by an Ornstein-

Uhlenbeck like process." In: *Conference on learning theory*. PMLR, pp. 483–513.

Borovkov, Konstantin, Yuliya Mishura, Alexander Novikov, and Mikhail Zhitlukhin (2017). "Bounds for expected maxima of Gaussian processes and their discrete approximations." In: *Stochastics* 89.1, pp. 21–37.

Bottou, Léon, Frank E Curtis, and Jorge Nocedal (2018). "Optimization methods for large-scale machine learning." In: *Siam Review* 60.2, pp. 223–311.

Bowen, Rufus (1975). "$\omega$-limit sets for axiom A diffeomorphisms." In: *Journal of differential equations* 18.2, pp. 333–339.

Braides, Andrea et al. (2002). *Gamma-convergence for beginners*. Vol. 22. Clarendon Press.

Bregman, Lev M (1967). "The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming." In: *USSR computational mathematics and mathematical physics* 7.3, pp. 200–217.

Brin, Michael and Garrett Stuck (2002). *Introduction to dynamical systems*. Cambridge university press.

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. (2020). "Language models are few-shot learners." In: *Advances in Neural Information Processing Systems* 33, pp. 1877–1901.

Cabot, Alexandre, Hans Engler, and Sébastien Gadat (2009). "On the long time behavior of second order differential equations with asymptotically small dissipation." In: *Transactions of the American Mathematical Society* 361.11, pp. 5983–6017.

Candes, Emmanuel J, Justin K Romberg, and Terence Tao (2006). "Stable signal recovery from incomplete and inaccurate measurements." In: *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences* 59.8, pp. 1207–1223.

Carrillo, José A, Robert J McCann, and Cédric Villani (2006). "Contractions in the 2-Wasserstein length space and thermalization of granular media." In: *Archive for Rational Mechanics and Analysis* 179.2, pp. 217–263.

Casgrain, Philippe (2019). "A latent variational framework for stochastic optimization." In: *Advances in Neural Information Processing Systems*, pp. 5647–5657.

Chang, Chih-Chung (2011). "LIBSVM: a library for support vector machines." In: *ACM Transactions on Intelligent Systems and Technology*.

Chaudhari, Pratik, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina (2019). "Entropy-SGD: Biasing gradient descent into wide valleys." In: *Journal of Statistical Mechanics: Theory and Experiment* 2019.12, p. 124018.

Chen, Jinghui, Dongruo Zhou, Yiqi Tang, Ziyan Yang, Yuan Cao, and Quanquan Gu (2020a). "Closing the generalization gap of adaptive gradient methods in training deep neural networks." In: *IJCAI*.

Chen, Mia Xu, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Niki Parmar, Mike Schuster, Zhifeng Chen, et al. (2018a). "The best of both worlds: Combining recent advances in neural machine translation." In: *arXiv:1804.09849*.

Chen, Ricky TQ, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud (2018b). "Neural ordinary differential equations." In: *Advances in Neural Information Processing Systems* 31.

Chen, Xiangning, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Yao Liu, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, et al. (2023). "Symbolic discovery of optimization algorithms." In: *arXiv preprint arXiv:2302.06675*.

Chen, Yuwen, Antonio Orvieto, and Aurelien Lucchi (2020b). "An accelerated DFO algorithm for finite-sum convex functions." In: *International Conference on Machine Learning*. PMLR, pp. 1681–1690.

Cho, Kyunghyun, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio (2014). "On the properties of neural machine translation: Encoder-decoder approaches." In: *arXiv preprint arXiv:1409.1259*.

Cho, Youngmin and Lawrence Saul (2009). "Kernel methods for deep learning." In: *Advances in Neural Information Processing Systems* 22.

Choi, Kwok Pui (1994). "On the medians of gamma distributions and an equation of Ramanujan." In: *Proceedings of the American Mathematical Society* 121.1, pp. 245–251.

Choudhuri, Amitava, Subrata Ghosh, and B Talukdar (2008). "Symmetries and conservation laws of the damped harmonic oscillator." In: *Pramana* 70.4, pp. 657–667.

Chow, Shui-Nee and Erik S Van Vleck (1994). "A shadowing lemma approach to global error analysis for initial value ODEs." In: *SIAM Journal on Scientific Computing* 15.4, pp. 959–976.

Chung, Stephen and Hava Siegelmann (2021). "Turing completeness of bounded-precision recurrent neural networks." In: *Advances in Neural Information Processing Systems*.

Cohen, Jeremy, Simran Kaur, Yuanzhi Li, J Zico Kolter, and Ameet Talwalkar (2020). "Gradient descent on neural networks typically occurs at the edge of stability." In: *International Conference on Learning Representations*.

Coleman, Rodney (2012). *Calculus on normed vector spaces*. Springer Science & Business Media.

Compagnoni, Enea Monzio, Luca Biggio, Antonio Orvieto, Frank Norbert Proske, Hans Kersting, and Aurelien Lucchi (2023a). "An SDE for modeling SAM: Theory and insights." In: *International Conference on Machine Learning*. PMLR, pp. 25209–25253.

Compagnoni, Enea Monzio, Anna Scampicchio, Luca Biggio, Antonio Orvieto, Thomas Hofmann, and Josef Teichmann (2023b). "On the effectiveness of randomized signatures as reservoir for learning rough dynamics." In: *2023 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–8.

Conneau, Alexis, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov (2019). "Unsupervised cross-lingual representation learning at scale." In: *arXiv:1911.02116*.

Coomes, Brian A, Kenneth J Palmer, et al. (1995). "Rigorous computational shadowing of orbits of ordinary differential equations." In: *Numerische Mathematik* 69.4, pp. 401–421.

Córdova, Antonio, Walter Gautschi, and Stephan Ruscheweyh (1990). "Vandermonde matrices on the circle: spectral properties and conditioning." In: *Numerische Mathematik* 57.1, pp. 577–591.

Couillet, Romain, Gilles Wainrib, Hafiz Tiomoko Ali, and Harry Sevi (2016). "A random matrix approach to echo-state neural networks." In: *International Conference on Machine Learning*. PMLR, pp. 517–525.

Curtis, Frank E and Daniel P Robinson (2019). "Exploiting negative curvature in deterministic and stochastic optimization." In: *Mathematical Programming* 176.1-2, pp. 69–94.

Cutkosky, Ashok and Francesco Orabona (2019). "Momentum-based variance reduction in non-convex SGD." In: *Advances in Neural Information Processing Systems* 32.

D'Orazio, Ryan, Nicolas Loizou, Issam Laradji, and Ioannis Mitliagkas (2021). "Stochastic mirror descent: Convergence analysis and adaptive variants via the mirror stochastic Polyak Stepsize." In: *arXiv preprint arXiv:2110.15412*.

Damian, Alex, Tengyu Ma, and Jason Lee (2021). "Label noise SGD provably prefers flat global minimizers." In: *arXiv preprint arXiv:2106.06530*.

Daneshmand, Hadi, Jonas Kohler, Francis Bach, Thomas Hofmann, and Aurelien Lucchi (2020). "Batch normalization provably avoids ranks collapse for randomly initialised deep networks." In: *Advances in Neural Information Processing Systems* 33, pp. 18387–18398.

Daneshmand, Hadi, Jonas Kohler, Aurelien Lucchi, and Thomas Hofmann (2018). "Escaping saddles with stochastic gradients." In: *International Conference on Machine Learning*. PMLR, pp. 1155–1164.

Dao, Tri, Daniel Y Fu, Stefano Ermon, Atri Rudra, and Christopher Ré (2022). "Flashattention: Fast and memory-efficient exact attention with io-awareness." In: *arXiv preprint arXiv:2205.14135*.

Daunizeau, Jean (2017). "Semi-analytical approximations to statistical moments of sigmoid and softmax mappings of normal variables." In: *arXiv preprint arXiv:1703.00091*.

Dauphin, Yann N, Angela Fan, Michael Auli, and David Grangier (2017). "Language modeling with gated convolutional networks." In: *International Conference on Machine Learning*. PMLR.

Dauphin, Yann, Harm De Vries, and Yoshua Bengio (2015). "Equilibrated adaptive learning rates for non-convex optimization." In: *Advances in Neural Information Processing Systems* 28.

De Bartolomeis, Piersilvio, Antonio Orvieto, and Giambattista Parascandolo (2022). "Enhancing unit-tests for invariance discovery." In: *ICML Workshop on Spurious Correlations, Invariance and Stability*.

De, Soham and Sam Smith (2020). "Batch normalization biases residual blocks towards the identity function in deep networks." In: *Advances in Neural Information Processing Systems*.

Defazio, Aaron (2019). "On the curved geometry of accelerated optimization." In: *Advances in Neural Information Processing Systems* 32.

Defazio, Aaron and Konstantin Mishchenko (2023). "Learning-rate-free learning by D-adaptation." In: *International Conference on Machine Learning*.

Défossez, Alexandre, Leon Bottou, Francis Bach, and Nicolas Usunier (2022). "A simple convergence proof of Adam and Adagrad." In: *Transactions on Machine Learning Research*.

Devroye, Luc (2006). "Nonuniform random variate generation." In: *Handbooks in operations research and management science* 13, pp. 83–121.

Diakonikolas, Jelena and Michael I Jordan (2019). "Generalized momentum-based methods: A Hamiltonian perspective." In: *arXiv preprint*.

Dieker, Ton (2004). "Simulation of fractional Brownian motion." PhD thesis. Masters Thesis, Department of Mathematical Sciences, University of Twente.

Dinh, Laurent, Razvan Pascanu, Samy Bengio, and Yoshua Bengio (2017). "Sharp minima can generalize for deep nets." In: *International Conference on Machine Learning*. PMLR, pp. 1019–1028.

Dong, Yihe, Jean-Baptiste Cordonnier, and Andreas Loukas (2021). "Attention is not all you need: Pure attention loses rank doubly exponentially with depth." In: *International Conference on Machine Learning*. PMLR, pp. 2793–2803.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. (2020). "An image is worth 16x16 words: transformers for image recognition at scale." In: *International Conference on Learning Representations*.

Draxler, Felix, Kambis Veschgini, Manfred Salmhofer, and Fred Hamprecht (2018). "Essentially no barriers in neural network energy landscape." In: *International Conference on Machine Learning*, pp. 1309–1318.

Du, Simon S, Chi Jin, Jason D Lee, Michael I Jordan, Aarti Singh, and Barnabas Poczos (2017). "Gradient descent can take exponential time to escape saddle points." In: *Advances in Neural Information Processing Systems* 30.

Du, Simon, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai (2019). "Gradient descent finds global minima of deep neural networks." In: *International Conference on Machine Learning*. PMLR, pp. 1675–1685.

Duchi, John, Elad Hazan, and Yoram Singer (2011). "Adaptive subgradient methods for online learning and stochastic optimization." In: *Journal of Machine Learning Research* 12.7.

Durrett, Rick (2010). *Probability: theory and examples*. Cambridge university press.

Duruisseaux, Valentin and Melvin Leok (2021). "A variational formulation of accelerated optimization on Riemannian manifolds." In: *arXiv preprint*.

Dynkin, Evgenij Borisovic (1965). "Markov processes." In: *Markov Processes*. Springer, pp. 77–104.

Dziugaite, Gintare Karolina and Daniel M Roy (2018). "Data-dependent PAC-Bayes priors via differential privacy." In: *arXiv:1802.09583*.

Elman, Jeffrey L (1990). "Finding structure in time." In: *Cognitive science*.

Erichson, N. Benjamin, Omri Azencot, Alejandro Queiruga, Liam Hodgkinson, and Michael W. Mahoney (2021). "Lipschitz recurrent neural networks." In: *International Conference on Learning Representations*.

Foret, Pierre, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur (2020). "Sharpness-aware minimization for efficiently improving generalization." In: *International Conference on Learning Representations*.

França, Guilherme, Michael I Jordan, and René Vidal (2021). "On dissipative symplectic integration with applications to gradient-based optimization." In: *Journal of Statistical Mechanics: Theory and Experiment* 2021.4, p. 043402.

Funahashi, Ken-Ichi (1989). "On the approximate realization of continuous mappings by neural networks." In: *Neural networks* 2.3, pp. 183–192.

Garrigos, Guillaume and Robert M Gower (2023). "Handbook of convergence theorems for (stochastic) gradient methods." In: *arXiv preprint arXiv:2301.11235*.

Gautschi, Walter (1975). "Optimally conditioned Vandermonde matrices." In: *Numerische Mathematik* 24, pp. 1–12.

Gautschi, Walter and Gabriele Inglese (1987). "Lower bounds for the condition number of Vandermonde matrices." In: *Numerische Mathematik* 52.3, pp. 241–250.

Ge, Rong, Furong Huang, Chi Jin, and Yang Yuan (2015). "Escaping from saddle points — online stochastic gradient for tensor decomposition." In: *Conference on Learning Theory*, pp. 797–842.

Geering, H. P., G. Dondi, F. Herzog, and Keel S. (2011). *Stochastic Systems*.

Gelfand, IM and SV Fomin (2000). *Calculus of Variations (Translated and edited by Silverman)*.

Ghadimi, Euhanna, Hamid Reza Feyzmahdavian, and Mikael Johansson (2015). "Global convergence of the heavy-ball method for convex optimization." In: *2015 European control conference (ECC)*. IEEE.

Ghadimi, Saeed and Guanghui Lan (2013). "Stochastic first-and zeroth-order methods for nonconvex stochastic programming." In: *SIAM Journal on Optimization* 23.4.

Gidel, Gauthier, Francis Bach, and Simon Lacoste-Julien (2019). "Implicit regularization of discrete gradient dynamics in linear neural networks." In: *Advances in Neural Information Processing Systems* 32.

Ginibre, Jean (1965). "Statistical ensembles of complex, quaternion, and real matrices." In: *Journal of Mathematical Physics*.

Glorot, Xavier and Yoshua Bengio (2010). "Understanding the difficulty of training deep feedforward neural networks." In: *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 249–256.

Głuch, Grzegorz and Rüdiger Urbanke (2021). "Noether: The more things change, the more stay the same." In: *arXiv preprint*.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. MIT press.

Gotmare, Akhilesh, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher (2018). "A closer look at deep learning heuristics: learning rate restarts, warmup and distillation." In: *International Conference on Learning Representations*.

Gower, Robert Mansel, Nicolas Loizou, Xun Qian, Alibek Sailanbayev, Egor Shulgin, and Peter Richtárik (2019). "SGD: General analysis and improved rates." In: *International Conference on Machine Learning*.

Gower, Robert, Othmane Sebbouh, and Nicolas Loizou (2021). "SGD for structured nonconvex functions: Learning rates, minibatching and interpolation." In: *International Conference on Artificial Intelligence and Statistics*.

Gu, Albert, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré (2020). "Hippo: Recurrent memory with optimal polynomial projections." In: *Advances in Neural Information Processing Systems*.

Gu, Albert, Karan Goel, and Christopher Re (2021). "Efficiently modeling long sequences with structured state spaces." In: *International Conference on Learning Representations*.

Gu, Albert, Ankit Gupta, Karan Goel, and Christopher Ré (2022a). "On the parameterization and initialization of diagonal state space models." In: *arXiv preprint arXiv:2206.11893*.

Gu, Albert, Isys Johnson, Aman Timalsina, Atri Rudra, and Christopher Ré (2022b). "How to train your Hippo: State space models with generalized orthogonal basis projections." In: *arXiv preprint arXiv:2206.12037*.

Gunasekar, Suriya, Blake E. Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro (2017). "Implicit regularization in matrix factorization." In: *Advances in Neural Information Processing Systems* 30.

Gupta, Ankit, Albert Gu, and Jonathan Berant (2022a). "Diagonal state spaces are as effective as structured state spaces." In: *Advances in Neural Information Processing Systems*.

Gupta, Ankit, Harsh Mehta, and Jonathan Berant (2022b). "Simplifying and understanding state space models with diagonal linear RNNs." In: *arXiv preprint arXiv:2212.00768*.

Hairer, Ernst, Christian Lubich, and Gerhard Wanner (2003). "Geometric numerical integration illustrated by the Störmer–Verlet method." In: *Acta numerica* 12, pp. 399–450.

Hairer, Ernst and Gerhard Wanner (1996). "Solving ordinary differential equations II: Stiff and differential-algebraic problems." In: *Springer Series in Computational Mathematics* 14.

Hanin, Boris (2018). "Which neural net architectures give rise to exploding and vanishing gradients?" In: *Advances in Neural Information Processing Systems* 31.

Hanin, Boris and David Rolnick (2018). "How to start training: The effect of initialization and architecture." In: *Advances in Neural Information Processing Systems* 31.

Hanson, Joshua and Maxim Raginsky (2020). "Universal simulation of stable dynamical systems by recurrent neural nets." In: *Learning for Dynamics and Control*. PMLR, pp. 384–392.

HaoChen, Jeff Z, Colin Wei, Jason Lee, and Tengyu Ma (2021). "Shape matters: Understanding the implicit bias of the noise covariance." In: *Conference on Learning Theory*, pp. 2315–2357.

Hardt, Moritz, Tengyu Ma, and Benjamin Recht (2018). "Gradient descent learns linear dynamical systems." In: *The Journal of Machine Learning Research* 19.1, pp. 1025–1068.

Hasani, Ramin, Mathias Lechner, Tsun-Hsuan Wang, Makram Chahine, Alexander Amini, and Daniela Rus (2022). "Liquid Structural State-Space Models." In: *The International Conference on Learning Representations*.

Hayes, Wayne and Kenneth R Jackson (2005). "A survey of shadowing methods for numerical solutions of ordinary differential equations." In: *Applied Numerical Mathematics* 53.2-4, pp. 299–321.

Hayou, Soufiane, Eugenio Clerico, Bobby He, George Deligiannidis, Arnaud Doucet, and Judith Rousseau (2021). "Stable Resnet." In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 1324–1332.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2015). "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification." In: *Proceedings of the IEEE International Conference on Computer Vision*.

— (2016). "Deep residual learning for image recognition." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778.

He, Li, Qi Meng, Wei Chen, Zhi-Ming Ma, and Tie-Yan Liu (2018). "Differential equations for modeling asynchronous algorithms." In: *arXiv preprint arXiv:1805.02991*.

Helfrich, Kyle, Devin Willmott, and Qiang Ye (2018). "Orthogonal recurrent neural networks with scaled cayley transform." In: *International Conference on Machine Learning*. PMLR.

Hennigan, Tom, Trevor Cai, Tamara Norman, and Igor Babuschkin (2020). *Haiku: Sonnet for JAX*. Version 0.0.9. URL: http://github.com/deepmind/dm-haiku.

Hochreiter, Sepp (1991). "Untersuchungen zu dynamischen neuronales Netzen." In: *Diploma thesis, Institut fur Informatik, Technische Universitat Munchen*.

Hochreiter, Sepp and Jürgen Schmidhuber (1997a). "Flat minima." In: *Neural computation* 9.1, pp. 1–42.

— (1997b). "Long short-term memory." In: *Neural computation* 9.8, pp. 1735–1780.

Hoffmann, Jordan, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. (2022). "Training compute-optimal large language models." In: *arXiv preprint arXiv:2203.15556*.

Hopfield, John J (1982). "Neural networks and physical systems with emergent collective computational abilities." In: *Proceedings of the national academy of sciences*.

Hornik, Kurt (1991). "Approximation capabilities of multilayer feedforward networks." In: *Neural networks* 4.2, pp. 251–257.

Hornik, Kurt, Maxwell Stinchcombe, and Halbert White (1989). "Multilayer feedforward networks are universal approximators." In: *Neural networks* 2.5, pp. 359–366.

Huang, Xiao Shi, Felipe Perez, Jimmy Ba, and Maksims Volkovs (2020). "Improving transformer optimization through better initialization." In: *International Conference on Machine Learning*. PMLR, pp. 4475–4483.

Huang, Zhiqiu, Yuxuan Li, Nan Duan, Yan Chen, and Haizhou Li (2018). "Music transformer: generating music with long-term structure." In: *arXiv preprint arXiv:1809.04281*.

Hurst, Harold Edwin (1956). "Methods of using long-term storage in reservoirs." In: *Proceedings of the Institution of Civil Engineers* 5.5, pp. 519–543.

Ioffe, Sergey and Christian Szegedy (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift." In: *International Conference on Machine Learning*.

Irwin, Michael Charles (2001). *Smooth dynamical systems*. Vol. 17. World Scientific.

Isserlis, Leon (1918). "On a formula for the product-moment coefficient of any order of a normal frequency distribution in any number of variables." In: *Biometrika* 12.1/2, pp. 134–139.

Itô, Kiyosi (1944). "Stochastic integral." In: *Proceedings of the Imperial Academy* 20.8, pp. 519–524.

Jacquot, Raymond G (2019). *Modern digital control systems*. Routledge.

Jaeger, Herbert and Harald Haas (2004). "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication." In: *Science* 304.5667, pp. 78–80.

Jastrzebski, Stanislaw, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey (2017). "Three factors influencing minima in SGD." In: *arXiv preprint arXiv:1711.04623*.

Jiang, Yiding, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio (2019). "Fantastic generalization measures and where to find them." In: *International Conference on Learning Representations*.

Jin, Chi, Praneeth Netrapalli, Rong Ge, Sham M Kakade, and Michael I Jordan (2021). "On nonconvex optimization for machine learning: Gradients, stochasticity, and saddle points." In: *Journal of the ACM (JACM)* 68.2, pp. 1–29.

Jordan, Michael I (2018). "Dynamical, symplectic and stochastic perspectives on gradient-based optimization." In: *University of California, Berkeley*.

Jumper, John, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. (2021). "Highly accurate protein structure prediction with AlphaFold." In: *Nature*.

Jung, Alexander (2017). "A fixed-point of view on gradient methods for big data." In: *Frontiers in Applied Mathematics and Statistics* 3, p. 18.

Kalchbrenner, Nal, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Koray Kavukcuoglu (2016). "Neural machine translation in linear time." In: *arXiv preprint arXiv:1610.10099*.

Karras, Tero, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila (2020). "Training generative adversarial networks with limited data." In: *Advances in Neural Information Processing Systems* 33, pp. 12104–12114.

Kawaguchi, Kenji (2016). "Deep learning without poor local minima." In: *Advances in Neural Information Processing Systems*, pp. 586–594.

Kersting, Hans, Antonio Orvieto, Frank Proske, and Aurelien Lucchi (2023). "Mean first exit times of Ornstein–Uhlenbeck processes in high-dimensional spaces." In: *Journal of Physics A: Mathematical and Theoretical* 56.21, p. 215003.

Keskar, Nitish Shirish, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang (2016). "On large-batch training for deep learning: generalization gap and sharp minima." In: *International Conference on Learning Representations*.

Khalil, H.K. (2002). *Nonlinear Systems*. Pearson Education. Prentice Hall.

Kilian, Joe and Hava T Siegelmann (1996). "The dynamic universality of sigmoidal neural networks." In: *Information and computation*.

Kim, Sanghwan, Lorenzo Noci, Antonio Orvieto, and Thomas Hofmann (2023). "Achieving a better stability-plasticity trade-off via auxiliary networks in continual learning." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11930–11939.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization." In: *arXiv preprint arXiv:1412.6980*.

Kleinberg, Bobby, Yuanzhi Li, and Yang Yuan (2018). "An alternative view: When does SGD escape local minima?" In: *International Conference on Machine Learning*.

Kolmogorov, Andrei N (1940). "Wienersche spiralen und einige andere interessante kurven in Hilbertscen raum." In: *Acad. Sci. URSS (NS)* 26, pp. 115–118.

Königsberger, Konrad (2013). *Analysis 2*. Springer-Verlag.

Köpf, Andreas, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi-Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Richárd Nagyfi, et al. (2023). "OpenAssistant conversations – democratizing large language model alignment." In: *arXiv preprint arXiv:2304.07327*.

Krichene, Walid and Peter L Bartlett (2017). "Acceleration and averaging in stochastic descent dynamics." In: *Advances in Neural Information Processing Systems* 30.

Krichene, Walid, Alexandre Bayen, and Peter L Bartlett (2015). "Accelerated mirror descent in continuous and discrete time." In: *Advances in Neural Information Processing Systems* 28.

Krizhevsky, Alex, Geoffrey Hinton, et al. (2009). *Learning multiple layers of features from tiny images*.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks." In: *Advances in Neural Information Processing Systems* 25.

Kulakova, Anastasiya, Marina Danilova, and Boris Polyak (2018). "Non-monotone behavior of the Heavy-ball method." In: *arXiv:1811.00658*.

Kunstner, Frederik, Philipp Hennig, and Lukas Balles (2019). "Limitations of the empirical Fisher approximation for natural gradient descent." In: *Advances in Neural Information Processing Systems*, pp. 4156–4167.

Kushner, Harold and G George Yin (2003). *Stochastic Approximation and Recursive Algorithms and Applications*. Vol. 35. Springer Science & Business Media.

Landau, Lev Davidovich and Evgenii Mikhailovich Lifshitz (1976). *Mechanics: Volume 1*. Vol. 1. Butterworth-Heinemann.

Lanford, Oscar E (1985). "Introduction to hyperbolic sets." In: *Regular and chaotic motions in dynamic systems*. Springer, pp. 73–102.

Larsson, Stig and J-M Sanz-Serna (1994). "The behavior of finite element solutions of semilinear parabolic problems near stationary points." In: *SIAM Journal on Numerical Analysis* 31.4, pp. 1000–1018.

LeCun, Yann A, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller (1998). "Efficient backprop." In: *Neural networks: Tricks of the trade*. Springer, pp. 9–48.

LeCun, Yann (1998). "The MNIST database of handwritten digits." In: *http://yann. lecun. com/exdb/mnist/*.

Lee, Jason D, Max Simchowitz, Michael I Jordan, and Benjamin Recht (2016). "Gradient descent only converges to minimizers." In: *Conference on learning theory*. PMLR, pp. 1246–1257.

Levy, Kfir Y (2016). "The power of normalization: Faster evasion of saddle points." In: *arXiv preprint arXiv:1611.04831*.

Levy, Kfir Y, Alp Yurtsever, and Volkan Cevher (2018). "Online adaptive methods, universality and acceleration." In: *Advances in Neural Information Processing Systems* 31.

Li, Qianxiao, Cheng Tai, and E Weinan (2017). "Stochastic modified equations and adaptive stochastic gradient algorithms." In: *International Conference on Machine Learning*. PMLR, pp. 2101–2110.

Li, Shuai, Wanqing Li, Chris Cook, Ce Zhu, and Yanbo Gao (2018). "Independently recurrent neural network (IndRNN): Building a longer and deeper rnn." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5457–5466.

Li, Xiang and Antonio Orvieto (2022). "Should you follow the gradient flow? Insights from Runge-Kutta gradient descent." In: *ICML Workshop Continuous-time Perspectives in Machine Learning*.

Li, Xinyan, Qilong Gu, Yingxue Zhou, Tiancong Chen, and Arindam Banerjee (2020). "Hessian based analysis of SGD for deep nets: Dynamics and generalization." In: *Proceedings of the 2020 SIAM International Conference on Data Mining*. SIAM, pp. 190–198.

Li, Yuhong, Tianle Cai, Yi Zhang, Deming Chen, and Debadeepta Dey (2022a). "What makes convolutional models great on long sequence modeling?" In: *arXiv preprint arXiv:2210.09298*.

Li, Zhong, Jiequn Han, Weinan E, and Qianxiao Li (2022b). "Approximation and optimization theory for linear continuous-time recurrent neural networks." In: *The Journal of Machine Learning Research* 23.1, pp. 1997–2081.

Lin, Hongzhou, Julien Mairal, and Zaid Harchaoui (2018). "Catalyst acceleration for first-order convex optimization: from theory to practice." In: *Journal of Machine Learning Research* 18.1, pp. 7854–7907.

Liu, Hong, Zhiyuan Li, David Hall, Percy Liang, and Tengyu Ma (2023). "Sophia: a scalable stochastic second-order optimizer for language model pre-training." In: *arXiv preprint arXiv:2305.14342*.

Liu, Linqing, Huan Wang, Jimmy Lin, Richard Socher, and Caiming Xiong (2019a). "Mkd: a multi-task knowledge distillation approach for pretrained language models." In: *arXiv preprint arXiv:1911.03588*.

Liu, Liyuan, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han (2019b). "On the variance of the adaptive learning rate and beyond." In: *arXiv preprint:1908.03265*.

Liu, Liyuan, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han (2020a). "Understanding the difficulty of training transformers." In: *arXiv preprint arXiv:2004.08249*.

Liu, Shengchao, Dimitris Papailiopoulos, and Dimitris Achlioptas (2020b). "Bad global minima exist and SGD can reach them." In: *Advances in Neural Information Processing Systems* 33, pp. 8543–8552.

Liu, Tianyi, Yan Li, Song Wei, Enlu Zhou, and Tuo Zhao (2021). "Noisy gradient descent converges to flat minima for nonconvex matrix factorization." In: *International Conference on Artificial Intelligence and Statistics*, pp. 1891–1899.

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

Loshchilov, Ilya and Frank Hutter (2016). "Sgdr: Stochastic gradient descent with warm restarts." In: *arXiv preprint arXiv:1608.03983*.

— (2017). "Decoupled weight decay regularization." In: *arXiv preprint arXiv:1711.05101*.

Lu, Zhou, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang (2017). "The expressive power of neural networks: A view from the width." In: *Advances in Neural Information Processing Systems* 30.

Lubich, Ch, Kaspar Nipp, and D Stoffer (1995). "Runge–Kutta solutions of stiff differential equations near stationary points." In: *SIAM Journal on Numerical Analysis* 32.4, pp. 1296–1307.

Lucchi, Aurelien, Antonio Orvieto, and Adamos Solomou (2021). "On the second-order convergence properties of random search methods." In: *Advances in Neural Information Processing Systems* 34, pp. 25633–25645.

Lucchi, Aurelien, Frank Proske, Antonio Orvieto, Francis Bach, and Hans Kersting (2022). "On the theoretical properties of noise correlation in stochastic optimization." In: *Advances in Neural Information Processing Systems* 35, pp. 14261–14273.

Maddison, Chris J, Daniel Paulin, Yee Whye Teh, Brendan O'Donoghue, and Arnaud Doucet (2018). "Hamiltonian descent methods." In: *arXiv preprint arXiv:1809.05042*.

Magnus, Jan R and Heinz Neudecker (2019). *Matrix differential calculus with applications in statistics and econometrics*. John Wiley & Sons.

Mallows, Colin L and Donald Richter (1969). "Inequalities of Chebyshev type involving conditional expectations." In: *The Annals of Mathematical Statistics* 40.6, pp. 1922–1932.

Mandelbrot, Benoit B and John W Van Ness (1968). "Fractional Brownian motions, fractional noises and applications." In: *SIAM review* 10.4, pp. 422–437.

Mao, Xuerong (2007). *Stochastic differential equations and applications*. Elsevier.

Martens, James (2020). "New insights and perspectives on the natural gradient method." In: *The Journal of Machine Learning Research* 21.1, pp. 5776–5851.

Martens, James and Roger Grosse (2015). "Optimizing neural networks with kronecker-factored approximate curvature." In: *International Conference on Machine Learning*. PMLR, pp. 2408–2417.

Martin, Eric and Chris Cundy (2017). "Parallelizing linear recurrent neural nets over sequence length." In: *arXiv preprint arXiv:1709.04057*.

McCulloch, Warren S and Walter Pitts (1943). "A logical calculus of the ideas immanent in nervous activity." In: *The bulletin of mathematical biophysics*.

Mertikopoulos, Panayotis and Mathias Staudigl (2018). "On the convergence of gradient-like flows with noisy gradient input." In: *SIAM Journal on Optimization* 28.1, pp. 163–197.

Mil'shtejn, GN (1975). "Approximate integration of stochastic differential equations." In: *Theory of Probability & Its Applications* 19.3, pp. 557–562.

Monti, Roberto (May 2010). *Introduction to ordinary differential equations*. http://www.math.unipd.it/~monti/ED2/PC5Maggio.pdf.

Muehlebach, Michael and Michael I Jordan (2021). "Optimization with momentum: Dynamical, control-theoretic, and symplectic perspectives." In: *Journal of Machine Learning Research* 22.73, pp. 1–50.

Muehlebach, Michael and Michael Jordan (2020). "Continuous-time lower bounds for gradient-based algorithms." In: *International Conference on Machine Learning*. PMLR, pp. 7088–7096.

Nachum, Ido, Jan Hązła, Michael Gastpar, and Anatoly Khina (2021). "A Johnson–Lindenstrauss framework for randomly initialized CNNs." In: *arXiv preprint arXiv:2111.02155*.

Nemirovski, Arkadi, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro (2009). "Robust stochastic approximation approach to stochastic programming." In: *SIAM Journal on Optimization* 19.4, pp. 1574–1609.

Nemirovskii, A.S. and D.B. Yudin (1983). *Problem complexity and method efficiency in optimization*. A Wiley-Interscience publication. Wiley.

Nesterov, Yurii E (1983). "A method for solving the convex programming problem with convergence rate $O(1/k^2)$." In: *Dokl. Akad. Nauk SSSR*. Vol. 269, pp. 543–547.

Nesterov, Yurii (2013). *Introductory lectures on convex optimization: A basic course*. Vol. 87. Springer Science & Business Media.

— (2018). *Lectures on Convex Optimization*. 2nd. Springer.

Nesterov, Yurii and Vladimir Spokoiny (2017). "Random gradient-free minimization of convex functions." In: *Foundations of Computational Mathematics* 17.2, pp. 527–566.

Nesterov, Yurii et al. (2018). *Lectures on Convex Optimization*. Vol. 137. Springer.

Neyshabur, Behnam, Srinadh Bhojanapalli, David McAllester, and Nati Srebro (2017a). "Exploring generalization in deep learning." In: *Advances in Neural Information Processing Systems*.

Neyshabur, Behnam, Ryota Tomioka, Ruslan Salakhutdinov, and Nathan Srebro (2017b). "Geometry of optimization and implicit regularization in deep learning." In: *arXiv preprint arXiv:1705.03071*.

Nguyen, XuanLong, Martin J Wainwright, and Michael I Jordan (2010). "Estimating divergence functionals and the likelihood ratio by convex risk minimization." In: *IEEE Transactions on Information Theory* 56.11, pp. 5847–5861.

Nocedal, Jorge and Stephen Wright (2006). *Numerical optimization*. Springer Science & Business Media.

Noci, Lorenzo, Sotiris Anagnostidis, Luca Biggio, Antonio Orvieto, Sidak Pal Singh, and Aurelien Lucchi (2022). "Signal propagation in transformers: Theoretical perspectives and the role of rank collapse." In: *Advances in Neural Information Processing Systems* 35, pp. 27198–27211.

Noci, Lorenzo, Gregor Bachmann, Kevin Roth, Sebastian Nowozin, and Thomas Hofmann (2021). "Precise characterization of the prior predictive distribution of deep ReLU networks." In: *Advances in Neural Information Processing Systems* 34.

Noether, E. (1918). "Invariante Variationsprobleme." In: *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse*, pp. 235–257.

Noiato, Alessandro, Luca Biggio, and Antonio Orvieto (2023). "On the advantage of Lion compared to signSGD with momentum." In: *ICML Workshop on High-dimensional Learning Dynamics*.

Øksendal, Bernt (1990). "When is a stochastic integral a time change of a diffusion?" In: *Journal of theoretical probability* 3.2, pp. 207–226.

Øksendal, Bernt (2003). "Fractional Brownian motion in finance." In: *Preprint series. Pure mathematics*.

Oksendal, Bernt (2003). *Stochastic Differential Equations: An Introduction with Applications*. New York, NY, USA: Springer-Verlag New York, Inc. ISBN: 978-3-642-14394-6.

Øksendal, Bernt (2003). "Stochastic differential equations." In: *Stochastic differential equations*. Springer, pp. 65–84.

Ombach, Jerzy (1993). "The simplest shadowing." In: *Annales Polonici Mathematici*. Vol. 58, pp. 253–258.

OpenAI (2023). "GPT-4 technical report." In: *arXiv preprint arXiv:2303.08774*.

Orvieto, Antonio (2023). "An accelerated Lyapunov function for Polyak's heavy-ball on convex quadratics." In: *arXiv preprint arXiv:2301.05799*.

Orvieto, Antonio, Soham De, Caglar Gulcehre, Razvan Pascanu, and Samuel L Smith (2023a). "On the universality of linear recurrences followed by nonlinear projections." In: *ICML Workshop on High-dimensional Learning Dynamics*.

Orvieto, Antonio, Hans Kersting, Frank Proske, Francis Bach, and Aurelien Lucchi (2022a). "Anticorrelated noise injection for improved generalization." In: *International Conference on Machine Learning*. PMLR, pp. 17094–17116.

Orvieto, Antonio, Jonas Kohler, and Aurelien Lucchi (2019). "The role of memory in stochastic optimization." In: *Uncertainty in Artificial Intelligence*. PMLR.

Orvieto, Antonio, Jonas Kohler, Dario Pavllo, Thomas Hofmann, and Aurelien Lucchi (2022b). "Vanishing curvature in randomly initialized deep ReLU networks." In: *International Conference on Artificial Intelligence and Statistics*.

Orvieto, Antonio, Simon Lacoste-Julien, and Nicolas Loizou (2022c). "Dynamics of SGD with stochastic Polyak stepsizes: Truly adaptive variants and convergence to exact solution." In: *Advances in Neural Information Processing Systems* 35, pp. 26943–26954.

Orvieto, Antonio and Aurelien Lucchi (2019a). "Continuous-time models for stochastic optimization algorithms." In: *Advances in Neural Information Processing Systems* 32.

— (2019b). "Shadowing properties of optimization algorithms." In: *Advances in Neural Information Processing Systems* 32.

Orvieto, Antonio, Anant Raj, Hans Kersting, and Francis Bach (2023b). "Explicit regularization in overparametrized models via noise injection." In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 7265–7287.

Orvieto, Antonio, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De (2023c). "Resurrecting recurrent neural networks for long sequences." In: *International Conference on Machine Learning*.

Orvieto, Antonio and Lin Xiao (2023). "An new adaptive method for minimizing non-negative losses." In: *ICML Workshop on High-dimensional Learning Dynamics*.

Ott, Myle, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli (2019). "Fairseq: A fast, exten-

sible toolkit for sequence modeling." In: *Proceedings of NAACL-HLT 2019: Demonstrations*.

O'donoghue, Brendan and Emmanuel Candes (2015). "Adaptive restart for accelerated gradient schemes." In: *Foundations of computational mathematics* 15.3, pp. 715–732.

Palmer, Kenneth James (2013). *Shadowing in dynamical systems: theory and applications*. Vol. 501. Springer Science & Business Media.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu (2002). "Bleu: a method for automatic evaluation of machine translation." In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318.

Parascandolo, G, A Neitz, A Orvieto, L Gresele, and B Schölkopf (2021). "Learning explanations that are hard to vary." In: *Ninth International Conference on Learning Representations (ICLR 2021)*.

Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio (2013). "On the difficulty of training recurrent neural networks." In: *International Conference on Machine Learning*. PMLR.

Pennington, Jeffrey, Samuel Schoenholz, and Surya Ganguli (2017). "Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice." In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc.

Perko, Lawrence (2013). *Differential equations and dynamical systems*. Vol. 7. Springer Science & Business Media.

Pesme, Scott, Loucas Pillaud-Vivien, and Nicolas Flammarion (2021). "Implicit bias of SGD for diagonal linear networks: a provable benefit of stochasticity." In: *Advances in Neural Information Processing Systems* 34.

Pinkus, Allan (1999). "Approximation theory of the MLP model in neural networks." In: *Acta numerica* 8, pp. 143–195.

Polu, Stanislas, Jesse Michael Han, Kunhao Zheng, Mantas Baksys, Igor Babuschkin, and Ilya Sutskever (2022). "Formal mathematics statement curriculum learning." In: *arXiv preprint arXiv:2202.01344*.

Polyak, Boris T (1964). "Some methods of speeding up the convergence of iteration methods." In: *USSR Computational Mathematics and Mathematical Physics* 4.5, pp. 1–17.

Polyak, Boris (1987). *Introduction to Optimization*. Inc., Publications Division, New York.

Poole, Ben, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli (2016). "Exponential expressivity in deep neural networks through transient chaos." In: *Advances in Neural Information Processing Systems* 29.

Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu (2020). "Exploring the limits of transfer learning with a unified text-to-text transformer." In: *The Journal of Machine Learning Research* 21.1, pp. 5485–5551.

Rajan, Kanaka and Larry F Abbott (2006). "Eigenvalue spectra of random matrices for neural networks." In: *Physical review letters* 97.18, p. 188104.

Reddi, Sashank J, Satyen Kale, and Sanjiv Kumar (2018). "On the convergence of Adam and Beyond." In: *International Conference on Learning Representations*.

Robbins, Herbert and Sutton Monro (1951). "A stochastic approximation method." In: *The Annals of Mathematical Statistics*.

Rumelhart, David E, Geoffrey E Hinton, and Ronald J Williams (1985). *Learning internal representations by error propagation*. Tech. rep. California Univ San Diego La Jolla Inst for Cognitive Science.

Sauer, Tim and James A Yorke (1991). "Rigorous verification of trajectories for the computer simulation of dynamical systems." In: *Nonlinearity* 4.3, p. 961.

Saxe, Andrew M, James L McClelland, and Surya Ganguli (2013). "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks." In: *arXiv preprint arXiv:1312.6120*.

Saxe, Andrew M., James L. McClelland, and Surya Ganguli (2019). "A mathematical theory of semantic development in deep neural networks." In: *Proceedings of the National Academy of Sciences* 116.23, pp. 11537–11546.

Schoenholz, Samuel S, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein (2016). "Deep Information Propagation." In: *International Conference on Learning Representations*.

Schroeder, M. (1991). "Fractals, chaos, power laws: minutes from an infinite paradise." In: *New York. W. H. Freeman*, pp. 41–45.

Shalev-Shwartz, Shai, Ohad Shamir, and Karthik Sridharan (2011). "Learning kernel-based halfspaces with the 0-1 loss." In: *SIAM Journal on Computing* 40.6, pp. 1623–1646.

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

Shekhovtsov, Alexander and Boris Flach (2018). "Feed-forward propagation in probabilistic neural networks with categorical and max layers." In: *International Conference on Learning Representations*.

Shi, Bin, Simon S Du, Michael I Jordan, and Weijie J Su (2021). "Understanding the acceleration phenomenon via high-resolution differential equations." In: *Mathematical Programming*, pp. 1–70.

Shi, Bin, Simon S Du, Weijie Su, and Michael I Jordan (2019). "Acceleration via symplectic discretization of high-resolution differential equations." In: *Advances in Neural Information Processing Systems* 32.

Shi, Naichen and Dawei Li (2021). "Rmsprop converges with proper hyperparameter." In: *International conference on learning representation*.

Siegelmann, Hava T (2012). *Neural networks and analog computation: Beyond the Turing limit*. Springer Science & Business Media.

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

Simsekli, Umut, Levent Sagun, and Mert Gurbuzbalaban (2019). "A tail-index analysis of stochastic gradient noise in deep neural networks." In: *International Conference on Machine Learning*. PMLR, pp. 5827–5837.

Singh, Sidak Pal, Gregor Bachmann, and Thomas Hofmann (2021). "Analytic insights into structure and rank of neural network Hessian maps." In: *Advances in Neural Information Processing Systems*.

Smale, Stephen (1967). "Differentiable dynamical systems." In: *Bulletin of the American mathematical Society* 73.6, pp. 747–817.

Smith, Jimmy TH, Andrew Warrington, and Scott W Linderman (2022). "Simplified state space layers for sequence modeling." In: *arXiv preprint arXiv:2208.04933*.

Smith, Samuel L, Benoit Dherin, David GT Barrett, and Soham De (2021). "On the origin of implicit regularization in stochastic gradient descent." In: *arXiv preprint arXiv:2101.12176*.

Smith, Samuel L and Quoc V Le (2017). "A Bayesian perspective on generalization and stochastic gradient descent." In: *arXiv preprint arXiv:1710.06451*.

Smith, Samuel, Erich Elsen, and Soham De (2020). "On the generalization benefit of noise in stochastic gradient descent." In: *International Conference on Machine Learning*, pp. 9058–9067.

Song, Hwanjun, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee (2022). "Learning from noisy labels with deep neural networks: A survey." In: *IEEE Transactions on Neural Networks and Learning Systems*.

Soudry, Daniel, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro (2018). "The implicit bias of gradient descent on separable data." In: *Journal of Machine Learning Research* 19.1, pp. 2822–2878.

Staib, Matthew, Sashank Reddi, Satyen Kale, Sanjiv Kumar, and Suvrit Sra (2019). "Escaping saddle points with adaptive gradient methods." In: *International Conference on Machine Learning*, pp. 5956–5965.

Steil, Jochen J (2004). "Backpropagation-decorrelation: online recurrent learning with $O(N)$ complexity." In: *2004 IEEE International Joint Conference on Neural Networks*. IEEE.

Stich, Sebastian and Harsh Harshvardhan (2021). "Escaping local minima with stochastic noise." In: *Neurips Workshop on Optimization for Machine Learning*.

Stogin, John, Ankur Mali, and C Lee Giles (2020). "A provably stable neural network Turing machine." In: *arXiv e-prints*, arXiv–2006.

Stroock, Daniel W and SR Srinivasa Varadhan (2007). *Multidimensional diffusion processes*. Springer.

Su, Weijie, Stephen Boyd, and Emmanuel J Candes (2016). "A differential equation for modeling Nesterov's accelerated gradient method: theory and insights." In: *Journal of Machine Learning Research* 17.153, pp. 1–43.

Sutskever, Ilya, James Martens, George Dahl, and Geoffrey Hinton (2013). "On the importance of initialization and momentum in deep learning." In: *International Conference on Machine Learning*, pp. 1139–1147.

Szegleti, András and Ferenc Márkus (2020). "Dissipation in Lagrangian formalism." In: *Entropy* 22.9, p. 930.

Tan, Mingxing and Quoc Le (2019). "Efficientnet: Rethinking model scaling for convolutional neural networks." In: *International Conference on Machine Learning*. PMLR, pp. 6105–6114.

Tanaka, Hidenori and Daniel Kunin (2021). "Noether's learning dynamics: The role of kinetic symmetry breaking in deep learning." In: *arXiv preprint*.

Tay, Yi, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler (2020). "Long range arena: A benchmark for efficient transformers." In: *International Conference on Learning Representations*.

Temme, Nico M (1996). *Special functions: An introduction to the classical functions of mathematical physics*. John Wiley & Sons.

Tieleman, Tijmen and Geoffrey Hinton (2012). "Lecture 6.5 - RMSprop, Coursera: Neural networks for machine learning." In: *University of Toronto, Technical Report*.

Touvron, Hugo, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. (2023). "Llama 2: Open foundation and fine-tuned chat models." In: *arXiv preprint arXiv:2307.09288*.

Traoré, Cheik and Edouard Pauwels (2021). "Sequential convergence of AdaGrad algorithm for smooth convex optimization." In: *Operations Research Letters* 49.4, pp. 452–458.

Tselepidis, Nikolaos, Jonas Kohler, and Antonio Orvieto (2020). "Two-level K-FAC preconditioning for deep learning." In: *NeurIPS OPT Workshop*.

Tsuzuku, Yusuke, Issei Sato, and Masashi Sugiyama (2020). "Normalized flat minima: Exploring scale invariant definition of flat minima for neural networks using pac-bayesian analysis." In: *International Conference on Machine Learning*, pp. 9636–9647.

Van Vleck, Erik S (1995). "Numerical shadowing near hyperbolic trajectories." In: *SIAM Journal on Scientific Computing* 16.5, pp. 1177–1189.

Vapnik, Vladimir (1999). *The nature of statistical learning theory*. Springer science & business media.

Vaskevicius, Tomas, Varun Kanade, and Patrick Rebeschini (2019). "Implicit regularization for optimal sparse recovery." In: *Advances in Neural Information Processing Systems* 32.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is all you need." In: *Advances in Neural Information Processing Systems*.

Vaswani, Sharan, Issam Laradji, Frederik Kunstner, Si Yi Meng, Mark Schmidt, and Simon Lacoste-Julien (2020). "Adaptive gradient methods converge faster with over-parameterization (but you should do a line-search)." In: *arXiv preprint arXiv:2006.06835*.

Wang, Hongyu, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei (2022a). "DeepNet: Scaling transformers to 1000 layers." In: *arXiv preprint arXiv:2203.00555*.

Wang, Huan, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher (2018). "Identifying generalization properties in neural networks." In: *arXiv preprint arXiv:1809.07402*.

Wang, Qiang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao (2019). "Learning deep transformer models for machine translation." In: *arXiv preprint arXiv:1906.01787*.

Wang, Shida, Zhong Li, and Qianxiao Li (2022b). *The Effects of Nonlinearity on Approximation Capacity of Recurrent Neural Networks*.

Wang, Xiaoyu, Mikael Johansson, and Tong Zhang (2023). "Generalized Polyak Step Size for First Order Optimization with Momentum." In: *arXiv preprint arXiv:2305.12939*.

Ward, Rachel, Xiaoxia Wu, and Leon Bottou (2019). "AdaGrad stepsizes: Sharp convergence over nonconvex landscapes." In: *International Conference on Machine Learning*.

Watson, George Neville (1995). *A treatise on the theory of Bessel functions*. Cambridge university press.

Wen, Kaiyue, Tengyu Ma, and Zhiyuan Li (2022). "How does Sharpness-Aware Minimization minimize sharpness?" In: *arXiv:2211.05729*.

Wibisono, Andre, Ashia C Wilson, and Michael I Jordan (2016). "A variational perspective on accelerated methods in optimization." In: *proceedings of the National Academy of Sciences* 113.47, E7351–E7358.

Wilson, Ashia C, Lester Mackey, and Andre Wibisono (2019). "Accelerating rescaled gradient descent: Fast optimization of smooth functions." In: *Advances in Neural Information Processing Systems*, pp. 13533–13543.

Wilson, Ashia C, Rebecca Roelofs, Mitchell Stern, Nathan Srebro, and Benjamin Recht (2017). "The marginal value of adaptive gradient methods in machine learning." In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 4151–4161.

Wolf, Thomas et al. (Oct. 2020). "Transformers: State-of-the-art natural language processing." In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45.

Woodworth, Blake, Suriya Gunasekar, Jason D. Lee, Edward Moroshko, Pedro Savarese, Itay Golan, Daniel Soudry, and Nathan Srebro (2020). "Kernel and rich regimes in overparametrized models." In: *Conference on Learning Theory*, pp. 3635–3673.

Wu, Dong Sheng and Yi Min Xiao (2007a). "Dimensional properties of fractional Brownian motion." In: *Acta Mathematica Sinica, English Series* 23.4, pp. 613–622.

Wu, Dongsheng and Yimin Xiao (2007b). "Geometric properties of fractional Brownian sheets." In: *Journal of Fourier Analysis and Applications* 13.1, pp. 1–37.

Wu, Dongxian, Shu-Tao Xia, and Yisen Wang (2020). "Adversarial weight perturbation helps robust generalization." In: *Advances in Neural Information Processing Systems* 33, pp. 2958–2969.

Wu, Lei, Mingze Wang, and Weijie J Su (2022). "The alignment property of SGD noise and how it helps select flat minima: A stability analysis." In: *Advances in Neural Information Processing Systems*.

Xiao, Han, Kashif Rasul, and Roland Vollgraf (2017). "Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms." In: *arXiv preprint arXiv:1708.07747*.

Xiao, Lechao, Yasaman Bahri, Jascha Sohl-Dickstein, Samuel Schoenholz, and Jeffrey Pennington (2018). "Dynamical isometry and a mean field theory of cnns: How to train 10,000-layer vanilla convolutional neural networks." In: *International Conference on Machine Learning*. PMLR, pp. 5393–5402.

Xie, Yuege, Xiaoxia Wu, and Rachel Ward (2020a). "Linear convergence of adaptive stochastic gradient descent." In: *International Conference on Artificial Intelligence and Statistics*.

Xie, Zeke, Issei Sato, and Masashi Sugiyama (2020b). "A diffusion theory for deep learning dynamics: Stochastic gradient descent exponentially favors flat minima." In: *arXiv preprint arXiv:2002.03495*.

Xiong, Ruibin, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu

(2020). "On layer normalization in the transformer architecture." In: *International Conference on Machine Learning*. PMLR, pp. 10524–10533.

Xu, Pan, Tianhao Wang, and Quanquan Gu (2018). "Accelerated stochastic mirror descent: From continuous-time dynamics to discrete-time algorithms." In: *International Conference on Artificial Intelligence and Statistics*, pp. 1087–1096.

Yang, Junchi, Antonio Orvieto, Aurelien Lucchi, and Niao He (2022). "Faster single-loop algorithms for minimax optimization without strong concavity." In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 5485–5517.

Yang, Zhilin, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le (2019). "XLnet: Generalized autoregressive pretraining for language understanding." In: *Advances in Neural Information Processing Systems* 32.

Yao, Zhewei, Amir Gholami, Kurt Keutzer, and Michael W Mahoney (2020). "Pyhessian: Neural networks through the lens of the Hessian." In: *2020 IEEE International Conference on Big Data (Big Data)*, pp. 581–590.

Zavatone-Veth, Jacob and Cengiz Pehlevan (2021). "Exact marginal prior distributions of finite Bayesian neural networks." In: *Advances in Neural Information Processing Systems* 34.

Zhang, Bohang, Jikai Jin, Cong Fang, and Liwei Wang (2020a). "Improved analysis of clipping algorithms for non-convex optimization." In: *Advances in Neural Information Processing Systems*.

Zhang, Chiyuan, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals (2021a). "Understanding deep learning (still) requires rethinking generalization." In: *Communications of the ACM* 64.3, pp. 107–115.

Zhang, Guodong, Lala Li, Zachary Nado, James Martens, Sushant Sachdeva, George Dahl, Chris Shallue, and Roger B Grosse (2019a). "Which algorithmic choices matter at which batch sizes? Insights from a noisy quadratic model." In: *Advances in Neural Information Processing Systems*, pp. 8196–8207.

Zhang, Guodong, Chaoqi Wang, Bowen Xu, and Roger Grosse (2018). "Three mechanisms of weight decay regularization." In: *International Conference on Learning Representations*.

Zhang, Hongyi, Yann N Dauphin, and Tengyu Ma (2019b). "Fixup initialization: Residual learning without normalization." In: *arXiv preprint arXiv:1901.09321*.

Zhang, Jingzhao, Tianxing He, Suvrit Sra, and Ali Jadbabaie (2019c). "Why gradient clipping accelerates training: A theoretical justification for adaptivity." In: *International Conference on Learning Representations*.

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

Zhang, Peiyuan, Antonio Orvieto, and Hadi Daneshmand (2021b). "Rethinking the variational interpretation of accelerated optimization methods." In: *Advances in Neural Information Processing Systems* 34, pp. 14396–14406.

Zhang, Peiyuan, Antonio Orvieto, Hadi Daneshmand, Thomas Hofmann, and Roy S Smith (2021c). "Revisiting the role of Euler numerical integration on acceleration and stability in convex optimization." In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 3979–3987.

Zhao, Jim, Aurelien Lucchi, Frank Norbert Proske, Antonio Orvieto, and Hans Kersting (2022). "Batch size selection by stochastic optimal control." In: *NeurIPS Workshop: Has it Trained Yet?*

Zhinan, Zhang (2002). "The Jordan canonical form of a rational random matrix." In: *Science Direct Working Paper*.

Zhu, Zhanxing, Jingfeng Wu, Bing Yu, Lei Wu, and Jinwen Ma (2019). "The anisotropic noise in stochastic gradient descent: Its behavior of escaping from sharp minima and regularization effects." In: *International Conference on Machine Learning*, pp. 7654–7663.

## CURRICULUM VITAE

Antonio Orvieto
born 9th October 1993, Padua (Italy)

### EDUCATION

| | |
|---|---|
| 2019-2023 | *PhD in Computer Science*<br>supervised by Thomas Hofmann, ETH Zürich |
| 2016-2018 | *Master in Robotics, Systems and Control*<br>mit Auszeichnung, ETH Zürich |
| 2012-2015 | *Bachelor of Information Technology Engineering*<br>summa cum laude, University of Padua |

### INTERNSHIPS / VISITING POSITIONS

| | |
|---|---|
| 2023 (3 months) | *Meta (FAIR)* in Seattle, with Lin Xiao |
| 2022 (5 months) | *Deepmind* in London, with Razvan Pascanu |
| 2021 (2 months) | *INRIA* in Paris, with Francis Bach |
| 2021 (3 months) | *MILA* in Montreal, with Simon Lacoste-Julien |
| 2018 (1 month) | *MPI* in Tübingen, with Stefan Bauer |
| 2017 (6 months) | *HILTI* in Liechtenstein. |

### SERVICES

| | |
|---|---|
| Summer 2022 | Organizer for the *Optimization for Deep Learning* session at ICCOPT |
| 2020-now | Reviewer for IMA Journal of Numerical Analysis and the SIAM Journal on Numerical Analysis |
| 2019-now | Reviewer for NeurIPS, ICML, AISTATS, ICLR |