First-order methods for online and stochastic optimization, and approximate compiling

by

Liam Matthew Madden

B.S., California Polytechnic State University, 2017

M.S., University of Colorado Boulder, 2020

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Applied Mathematics

2022

Committee Members:

Emiliano Dall'Anese and Stephen Becker, Chair

Andrea Simonetto

Jem Corcoran

François Meyer

Madden, Liam Matthew (Ph.D., Applied Mathematics)

First-order methods for online and stochastic optimization, and approximate compiling

Thesis directed by Profs. Emiliano Dall'Anese and Stephen Becker

The oracle complexity model for optimization has been the source of many advances in optimization algorithms with applications to machine learning, artificial intelligence, and high-dimensional statistics. For example, many adaptive methods have their beginnings as algorithms with tune-able hyper-parameters appearing in theoretical complexity bounds. This thesis investigates three different problems from the oracle complexity model.

The first problem, presented in Chapter 2, considers the online smooth convex optimization problem, motivated by time-varying applications. Two important results are a lower bound on the upper bounds of general online first-order methods and an algorithm with a matching upper bound. The algorithm presented applies Nesterov's method to a stale cost function until sufficient convergence has been achieved at which point the cost function is updated to the current one.

The second problem, presented in Chapter 3, considers the approximate compiling problem of quantum computing. We focus on the CNOT+rotation gate set and consider three different CNOT patterns. Despite the non-convexity of the problem, we find that local minima perform well for random target circuits. We also find that simple CNOT patterns seem to achieve the lower bound on the number of CNOTs required to exactly compile random target circuits. We also use the optimization framework to explore and find new decompositions of particular target circuits such as the Toffoli gate.

The third problem, presented in Chapter 4, considers the stochastic smooth non-convex optimization problem, motivated by machine learning applications. Two important results are a Freedman-type concentration inequality that breaks through the sub-exponential threshold to heavier-tailed martingale difference sequences and a high probability convergence bound for stochastic gradient descent with sub-Weibull gradient noise.

Dedication

To the music of small streams, to clouds dreaming through blue skies, to purple lupines and red paintbrush, to the songs of chickadees enduring through snow, to the smell of sagebrush, to the warmth of sun and coolness of breeze, to the drum of rain and soft clapping of yellow aspen leaves, to the strength of ponderosa pines and poetry of junipers, and to the beautiful earth that lives through us all.

Acknowledgements

First and foremost, I am grateful for my advisors Emiliano Dall'Anese and Stephen Becker. Thank you for supporting me as a student as well as a human over the years. Not only did you teach me how to research (starting with that first academic paper you gave me Emiliano), but you encouraged me to collaborate with other researchers and gently advised me in life too. I can't imagine having two better advisors.

I am grateful for my advisor during two internships at IBM in the summers of 2020 and 2021. Thank you Andrea Simonetto for giving me free reign to explore problems while also gently guiding me so that those explorations could take shape as complete projects. And thank you for supporting me as a human as well. It would be more accurate to say I can't imagine having three better advisors!

I am also grateful for my mentors at IBM. Thank you Albert Akhriev, Anton Dekusar, Martin Mevissen, Ali Javadi, and Jiri Vala. I am grateful for my collaborators at CU. Thank you Nicola Bastianello and Seunghyun Kim. I am grateful for my committee. Thank you Jem Corcoran and François Meyer. Jem, I learned so much in your measure-theoretic probability class. And François, I learned so much in your high-dimensional data analysis class.

I am so grateful for all the experiences I had while in Colorado and the friends I shared them with. Thank you. And, of course, thank you to my parents, Pat and Polly, and my sisters, Kailee and Clare, for always being so supportive. I also gratefully acknowledge support from the National Science Foundation (NSF), Division of Mathematical Sciences, through the award # 1923298.

Contents

Tables

Table

Figures

Figure

Chapter 1

Introduction

The optimization framework presented 40 years ago in [3] has found a happy home in today's age of machine learning, artificial intelligence, and high-dimensional statistics. Deriving theoretical guarantees on the number of gradient calls required to reach a certain optimization precision (or, equivalently, on the precision for a particular number of gradient calls) can lead to new insights and developments in existing first-order algorithms. In this thesis, we focus on three different problems framed in such a way. For each, we give practical motivation for the problem, present new algorithms, and provide theoretical guarantees for the algorithms.

The basic optimization framework we focus on is the following: (1) we are given a smooth function f with Lipschitz continuous gradient, with minimum $f^\star > -\infty$, and with minimizer set $X^\star \neq \emptyset$; (2) we iteratively construct a sequence of vectors (x_t), making one gradient call per iteration, then output a final vector x_{out}; and (3) theoretical guarantees are in the form of bounds on $f(x_{\text{out}}) - f^\star$, $\text{dist}(x_{\text{out}}, X^\star)^2$, or $\|\nabla f(x_{\text{out}})\|^2$.

The online optimization framework is a variation on the basic framework: (1) we are given a sequence of smooth convex functions (f_t) with Lipschitz continuous gradients and similarly denoted minima and minimizer sets; (2) we iteratively construct a sequence of vectors (x_t), making one gradient call per iteration, where the gradient can be the current one or a past one; and (3) theoretical guarantees are in the form of bounds on the tracking error: $\limsup_t f(x_t) - f_t^\star$, $\limsup_t \text{dist}(x_t, X_t^\star)^2$, or $\limsup_t \|\nabla f(x_t)\|^2$.

The stochastic optimization framework is another variation on the basic framework: (1) we

are given a smooth function f with Lipschitz continuous gradient; (2) we iteratively construct a sequence of vectors (x_t), making one stochastic gradient call per iteration where the expected value of the stochastic gradient is assumed to equal the true gradient, then output a final vector x_{out}; and (3) theoretical guarantees are in the form of bounds on $f(x_{\text{out}}) - f^\star$, $\text{dist}(x_{\text{out}}, X^\star)^2$, or $\|\nabla f(x_{\text{out}})\|^2$.

The approximate compiling problem is a particular application of optimization to quantum computing. The problem can be written as

$$\min_{x \in \mathbb{R}^n} \frac{1}{2d} \|V(x) - U\|_F^2$$

where $V(x)$ and U are both d by d unitary matrices and $\|\cdot\|_F$ is the Frobenius norm. A forthcoming work (not included in this thesis) considers a stochastic form of the objective,

$$\frac{1}{2d} \|V(x) - U\|_F^2 = \mathbb{E}\left[\frac{1}{2} \|\xi\xi^\dagger [V(x) - U]\|_F^2\right]$$

where ξ is sampled from the uniform distribution on the unit sphere in $\mathbb{C}^d$; and a sketched form of the objective,

$$\frac{1}{2m} \|QQ^\dagger [V(x) - U]\|_F^2.$$

Chapter 2 deals with the online problem and was published in March 2021:

> Bounds for the Tracking Error of First-Order Online Optimization Methods. Liam Madden, Stephen Becker, Emiliano Dall'Anese. *Journal of Optimization Theory and Applications*, 189:437-457, 2021.

with abstract:

> This paper investigates online algorithms for smooth time-varying optimization problems, focusing first on methods with constant step-size, momentum, and extrapolation-length. Assuming strong convexity, precise results for the tracking iterate error (the limit supremum of the norm of the difference between the optimal solution and the iterates) for online gradient descent are derived. The paper then considers a general first-order framework, where

a universal lower bound on the tracking iterate error is established. Furthermore, a method using "long-steps" is proposed and shown to achieve the lower bound up to a fixed constant. This method is then compared with online gradient descent for specific examples. Finally, the paper analyzes the effect of regularization when the cost is not strongly convex. With regularization, it is possible to achieve a non-regret bound. The paper ends by testing the accelerated and regularized methods on synthetic time-varying least-squares and logistic regression problems, respectively.

Chapter 3 deals with the approximate compiling problem and was published in March 2022:

Best Approximate Quantum Compiling Problems. Liam Madden, Andrea Simonetto. *ACM Transactions on Quantum Computing,* 3(2):1-29, 2022.

with abstract:

We study the problem of finding the best approximate circuit that is the closest (in some pertinent metric) to a target circuit, and which satisfies a number of hardware constraints, like gate alphabet and connectivity. We look at the problem in the CNOT+rotation gate set from a mathematical programming standpoint, offering contributions both in terms of understanding the mathematics of the problem and its efficient solution. Among the results that we present, we are able to derive a 14-CNOT 4-qubit Toffoli decomposition from scratch, and show that the Quantum Shannon Decomposition can be compressed by a factor of two without practical loss of fidelity.

Chapter 4 deals with the approximate compiling problem and was submitted to a journal in November 2021. It can be found on arXiv:

High-probability Convergence Bounds for Non-convex Stochastic Gradient Descent. Liam Madden, Emiliano Dall'Anese, Stephen Becker. *arXiv preprint: arxiv:2006.05610*, 2021.

with abstract:

Stochastic gradient descent is one of the most common iterative algorithms used in machine learning. While being computationally cheap to implement, recent literature suggests it may have implicit regularization properties that prevent over-fitting. This paper analyzes the properties of stochastic gradient descent from a theoretical standpoint to help bridge the gap between theoretical and empirical results. Most theoretical results either assume convexity or only provide convergence results in mean, while this paper proves convergence bounds in high probability without assuming convexity. Assuming strong smoothness, we prove high probability convergence bounds in two settings: (1) assuming the Polyak-Łojasiewicz inequality and norm sub-Gaussian gradient noise and (2) assuming norm sub-Weibull gradient noise. In the first setting, we combine our convergence bounds with existing generalization bounds in order to bound the true risk and show that for a certain number of epochs, convergence and generalization balance in such a way that the true risk goes to the empirical minimum as the number of samples goes to infinity. In the second setting, as an intermediate step to proving convergence, we prove a probability result of independent interest. The probability result extends Freedman-type concentration beyond the sub-exponential threshold to heavier-tailed martingale difference sequences.

Chapter A contains the appendices for Chapter 4.

Chapter 2

Tracking error bounds for online convex optimization

2.1 Introduction

Modern optimization applications have increased in scale and complexity [4], and furthermore, some applications require solutions to a series of problems with low latency. For example, in contrast to legacy power distribution systems that were built for constant and unidirectional flow, modern power systems that include solar power at the residential nodes must incorporate variable and bidirectional flow. Thus, in order to preserve efficiency, control decisions, solved via optimization, need to be made frequently, at the time scale of changing renewable generation and non-controllable loads (e.g., seconds). Making the problem even harder is that the problem must be solved for a greater number of control points, and so it takes longer to find a suitable solution. In this example, and many others, batch algorithms take longer to find a suitable solution than is allowed, and so we have to abandon them in favor of online algorithms. Motivated by applications requiring a time-varying framework, such as power systems [5, 6], transportation systems [7], and communication networks [8], this chapter evaluates such online algorithms.

In particular, we consider online first-order algorithms. If it takes time $h > 0$ to make one gradient call (or one gradient call and one evaluation of the proximal operator), then we encode this into the time-varying problem by discretizing the cost function with respect to h, leading to a sequence of cost functions. The goal of an online algorithm is to track the minimum of this sequence, taking only one step at each time.

In the presence of strong convexity, upper bounds for online gradient descent have been proved

in, e.g., [9, 10]. For completeness, we include such an upper bound in Theorem 1. Furthermore, for the first time it is established (Theorem 2) that this is a tight bound. Beyond online gradient descent, there are dynamic regret bounds for online accelerated methods, but these are not shown to be optimal [11]. This chapter goes further. First, Theorem 4 proves a lower bound for online first-order methods in general. Then, we define a method that we call online long-step Nesterov's method, and prove a proportional upper bound for it in Theorem 5. In the absence of strong convexity, we show, in Theorem 6, that regularizing online gradient descent leads to a useful bound that is not in terms of the regret. Finally, this chapter demonstrates the performance of the algorithms by applying them to synthetic time-varying least squares and logistic regression problems.

The rest of the chapter is organized as follows. Section 2.2 provides all the necessary preliminaries. Section 2.3 considers online gradient descent, online Polyak's method, and online Nesterov's method. Section 2.4 considers the full class of online first-order methods, including online long-step Nesterov's method. Section 2.5 details the results for online regularized gradient descent. Section 2.6 demonstrates the performance of the algorithms on some numerical examples. Section 2.7 concludes the chapter.

2.2 Preliminaries

This section reviews useful results from convex analysis [12, 13, 14, 15, 16, 17], and introduces key definitions that will be used throughout the chapter. The chapter considers functions on a Hilbert space $\mathcal{H}$ with corresponding norm $\|\cdot\|$. We assume all functions, $f_t : \mathcal{H} \to \mathbb{R}$, are proper, lower semicontinuous, convex, and strongly smooth (meaning differentiable with Lipschitz continuous gradient); $t \in \mathbb{N} \cup \{0\}$ denotes the time index [4, 18, 19, 20]. The discretized time-varying optimization problem of interest is the sequence of convex problems (one for each t):

$$\min_{x \in \mathcal{H}} f_t(x)\,, \qquad t \in \mathbb{N} \cup \{0\} \tag{2.1}$$

along with a time-varying minimizer set. In particular, assume that the minimizer set, denoted as $X_t^\star$, is nonempty for each t. Let $x_t^\star$ denote an element of $X_t^\star$ and $f_t^\star = f_t(x_t^\star)$. We denote the set of

sequences of L-strongly smooth functions by $\mathcal{S}'(L)$. For function sequences that are additionally μ-strongly convex, $X_t^\star$ contains only one point; therefore, we can measure the temporal variability of the optimal solution trajectory of (2.1) with the sequence

$$\sigma_t := \|x_{t+1}^\star - x_t^\star\|, \quad t \in \mathbb{N} \cup \{0\}.$$

We define $\mathcal{S}(\kappa^{-1}, L, \sigma)$, where κ is the condition number, as the set of L-strongly smooth, $\kappa^{-1}L$-strongly convex functions sequences with $\sigma_t \leq \sigma$ for all t. For both $\mathcal{S}'(L)$ and $\mathcal{S}(\kappa^{-1}, L, \sigma)$, the Hilbert space, along with its dimension, is left implicit since the results are independent of it.

Throughout the chapter, we will consider various measures of optimality [18, 19, 20, 4, 21, 22, 23, 24, 9]: the iterate error $\|x_t - x_t^\star\|$, the function error $f_t(x_t) - f_t^\star$, and the gradient error $\|\nabla f_t(x_t)\|$. For functions that are not strongly convex, the iterate error yields the strongest results, while the gradient error is the weakest. That is,

$$\|\nabla f_t(x)\| \leq L\|x - x_t^\star\|, \tag{2.2}$$

$$f_t(x) - f_t^\star \leq \frac{L}{2}\|x - x_t^\star\|^2, \tag{2.3}$$

$$\text{and } \|\nabla f_t(x)\|^2 \leq 2L\left(f_t(x) - f_t^\star\right) \tag{2.4}$$

where the first two inequalities follow from standard arguments in convex analysis, and the third inequality follows by comparing a point and a gradient step from it in the definition of strong smoothness [25, Sec. 12.1.3]. For functions that are strongly convex, bounds in the opposite directions can be found.

Due to the temporal variability, we cannot expect any of the error sequences to converge to zero in general. Thus, we will characterize the performance of online algorithms via bounds on the limit supremum of the errors, which we term "tracking" error, rather than bounds on the convergence rate (to the tracking error ball).

Note that for $\mathcal{S}'(L)$, the regret literature uses the dynamic regret, $Reg_T := \sum_{t=0}^{T} f_t(x_t) - f_t^\star$, as a measure of optimality [23, 21, 24, 22]. The path variation, function variation, and gradient

variation—respectively,

$$V_T^p = \max_{\{x_t^\star \in X_t^\star\}_{t=0}^T} \sum_{t=1}^T \|x_t^\star - x_{t-1}^\star\|,$$
$$V_T^f = \sum_{t=1}^T \sup_x |f_t(x) - f_{t-1}(x)|,$$
$$\text{and } V_T^g = \sum_{t=1}^T \sup_x \|\nabla f_t(x) - \nabla f_{t-1}(x)\|$$

—are used to bound the dynamic regret. On the other hand, our approach to $\mathcal{S}'(L)$ is inspired by regularization reduction [26]. Through regularization, we are able to bound the tracking gradient error via the (σ_t) corresponding to the regularized problem. But, there are a couple of reasons why our bound cannot be compared with the bounds in the regret literature. First, it is possible for $Reg_T/(T+1)$ to be bounded while the function error has a subsequence going to infinity. For example, let $(\Delta_t)_{t\in\mathbb{N}\cup\{0\}} = (1, 0, 2, 0, 0, 3, 0, 0, 0, \ldots)$. Then $\frac{1}{T+1}\sum_{t=0}^T \Delta_t \leq 1$ even though there is a subsequence of (Δ_t) that goes to infinity. Second, in order for the function variation or gradient variation to be finite, the constraint set must be compact. Our bound only requires that the optimal solution trajectory lie in a compact set.

In this chapter, we consider three general algorithms: $\texttt{ALG}(\alpha, \beta, \eta)$ presented in Section 2.3, online long-step Nesterov's method ($\texttt{OLNM}(T)$) presented in Section 2.4, and online regularized gradient descent ($\texttt{ORGD}(\delta, x_c)$), presented in Section 2.5. $\texttt{ALG}(\alpha, \beta, \eta)$ encompasses methods such as online gradient descent, online Polyak's method (also known as the online heavy ball method), and online Nesterov's method as special cases (see Table 2.1).

The proposed OLNM is motivated by the following observation: suppose that the optimizer has access to all the previous functions; depending on the temporal variability of the problem, the question we pose is whether it is better to use the same function for some number of iterations before switching to a new one, or utilize a new function at each iteration. This question is relevant especially in the case where "observing" a new function may incur a cost (for example, in sensor networks, where acquisition of new data may be costly from a battery and data transmission standpoint). We call the former idea "long-steps," drawing an analogy to interior-point methods [27,

Sec. 14], [16, Sec. 11]. Specifically, we pause at a function for some number of iterations, and we apply a first-order algorithm with the optimal coefficients for the number of iterations, in the sense of [28, 29, 30]. Thus, short-steps are online gradient steps. Restarting Nesterov's method has been shown to be optimal for strongly convex functions in, e.g. [31], [32, App. D]. It turns out that the optimal restart count as a long-step length is optimal for the strongly convex time-varying problem.

On the other hand, the literature on batch algorithms with inexact gradient calls has shown that accelerated methods accumulate errors for some non-strongly convex functions [33, Prop. 2], [34, Thm. 7], [35, 36, 37]. This suggests that there may be some non-strongly convex function sequences such that all online accelerated methods perform worse than online gradient descent. On the other hand, only averaged function error bounds exist for online gradient descent. We get around this via regularization; in particular, we balance the error introduced by the regularization term with a smaller tracking error achieved by the regularized algorithm. This allows us to derive a non-averaged error bound for `ORGD`.

While the chapter considers strongly smooth functions for simplicity of exposition, we note how to extend results to the composite case (where the cost function is the sum of f and a possibly nondifferentiable function with computable proximal operator, such as an indicator function encoding constraints or an ℓ_1 penalty). We leave the analysis for Banach spaces as a future direction. See [38, Sec. 3.1] for an excellent treatment of acceleration for Banach spaces.

2.3 Simple first-order methods

In this section, we focus on the class of algorithms that, given x_0, construct (x_t) via:

$$x_{t+1} = x_t - \alpha \nabla f_t(y_t) + \beta(x_t - x_{t-1}) \qquad (\texttt{ALG}(\alpha, \beta, \eta))$$

$$y_{t+1} = x_{t+1} + \eta(x_{t+1} - x_t)$$

which will be referred to as $\texttt{ALG}(\alpha, \beta, \eta)$, where $\alpha > 0$ is the step-size, β is the momentum, and η is the extrapolation-length. $\texttt{ALG}(\alpha, 0, 0)$ corresponds to online gradient descent, $\texttt{ALG}(\alpha, \beta, 0)$ to an online version of Polyak's method [39], and $\texttt{ALG}(\alpha, \beta, \beta)$ to an online version of Nesterov's method

Form of $\texttt{ALG}(\alpha,\beta,\eta)$	Typical parameter choice	Name
$\texttt{ALG}(\alpha,0,0)$	$0<\alpha<\frac{2}{L}$	Online gradient descent
$\texttt{ALG}(\alpha,\beta,0)$	$\alpha=\frac{4}{\left(\sqrt{L}+\sqrt{\mu}\right)^2}$, $\beta=\left(\frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}\right)^2$	Online Polyak's method [39]
$\texttt{ALG}(\alpha,\beta,\beta)$	$\alpha=1/L$, $\beta=\eta=\frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}$	Online Nesterov's method [15]

Table 2.1: Special cases of $\texttt{ALG}(\alpha,\beta,\eta)$ for $(f_t)\in\mathcal{S}(\kappa^{-1},L,\sigma)$ where $\mu=L/\kappa$.

[15].

It should be pointed out that, while the analysis of online algorithms for time-varying optimization such as $\texttt{ALG}(\alpha,\beta,\eta)$ share commonalities with online learning [40, 21, 22], the two differ in their motivations and aspects of their implementations, as noted in [4]. For example, in online learning frameworks, the step size may depend on the time-horizon or, in the case of an infinite time-horizon, a doubling-trick may be utilized [41, Sec. 2.3.1]. On the other hand, every step of an online algorithm applied to a time-varying problem is essentially the *first* step at that time. Hence, the parameters of the algorithm should be cyclical or depend on measurements of the temporal variation. We only consider the former. The simplest subset of cyclical algorithms is the set of algorithms with constant parameters, as in ALG.

In the following subsections, we give a tight bound on the performance of online gradient descent and analyze ALG for two examples.

2.3.1 Online gradient descent

When the function f_t is strongly convex for all t, upper bounds on the tracking errors for online gradient descent are available in the literature (see, e.g., [9, 10, 42, 43]); these results are tailored to the setting considered in this chapter by the following theorem, which is followed by two examples used to give intuition and prove tightness results.

Theorem 1. *Suppose that* $(f_t)\in\mathcal{S}(\kappa^{-1},L,\sigma)$ *and let* $\alpha\in]0,2/(\mu+L)]$. *Then, given* x_0, $\texttt{ALG}(\alpha,0,0)$

constructs a sequence (x_t) *such that*

$$\limsup_{t\to\infty} \|x_t - x_t^\star\| \leq (\alpha\mu)^{-1}\sigma \tag{2.5}$$

where $\mu = \kappa^{-1}L$*. In particular, the bound is minimized for* $\alpha = \frac{2}{\mu+L}$*, in which case,*

$$\limsup_{t\to\infty} \|x_t - x_t^\star\| \leq \frac{\kappa+1}{2}\sigma. \tag{2.6}$$

The proof follows by using the fact that the map $I - \alpha\nabla f_t$ is Lipschitz continuous with parameter $c = \max\{|1-\alpha\mu|, |1-\alpha L|\}$ [44, Sec. 5.1] and $x_t^\star$ is a fixed point of that map. The result straightforwardly extends to the composite case by using the nonexpansiveness property of the proximal operator and a similar fixed point result.

2.3.2 Example: translating quadratic

For quadratic functions with constant positive definite matrix but minimizer moving at constant speed on a straight line, the analysis of [45] can be extended, using the Neumann series, to show that the set of (α, β, η) such that `ALG` has a finite worst-case tracking iterate error is exactly the stability set of (α, β, η) in the batch setting (i.e., in a setting where the cost does not change during the execution of the algorithm). As an example, the stability set for online Nesterov's method is given in [46, Prop. 3.6]. Thus, in the online setting, one should consider only (α, β, η) that are in the batch stability set.

As a particular example, let $f(x) = \frac{1}{2}x^T Ax$, with $A = \mathrm{diag}(\mu, L, L, ...)$. Given (α, β, η) and initialization x_0, we want to construct $(f_t) \in \mathcal{S}(\mu/L, L, \sigma)$ in such a way that the iterates of $\texttt{ALG}(\alpha, \beta, \eta)$ trail behind $(x_t^\star)$ at a constant distance. Towards this end, define $\xi = \left(\frac{1-\beta}{\alpha\mu} + \eta\right)\sigma$, which will end up being the trailing distance. Let e denote the first canonical basis vector, and define $f_t = f(\cdot - x_t^\star)$ where $x_t^\star = x_0 + (t\sigma + \xi)e$. By induction, we will show that $\Delta_t \coloneqq x_t - x_t^\star = -\xi e \ \forall t \in \mathbb{N} \cup \{0\}$. The base case follows by construction. Now, assume the result holds for all

$t' \leq t$. Then,

$$\begin{aligned}
\Delta_{t+1} &= (1+\beta)x_t - \beta x_{t-1} - x^\star_{t+1} - \alpha\nabla f_t\big((1+\eta)x_t - \eta x_{t-1}\big) \\
&= (1+\beta)\Delta_t - \beta\Delta_{t-1} + (1+\beta)(x^\star_t - x^\star_{t+1}) - \beta(x^\star_{t-1} - x^\star_{t+1}) \\
&\qquad - \alpha\nabla f\left((1+\eta)\Delta_t - \eta\Delta_{t-1} - \eta(x^\star_{t-1} - x^\star_t)\right) \\
&= -(1+\beta)\xi e + \beta\xi e - (1+\beta)\sigma e + 2\beta\sigma e \\
&\qquad - \alpha\nabla f\left(-(1+\eta)\xi e + \eta\xi e + \eta\sigma e\right) \\
&= -\xi e - (1-\beta)\sigma e - \alpha\nabla f(-\xi e + \eta\sigma e) \\
&= -\xi e + \alpha\mu\xi e - (1 - \beta + \alpha\mu\eta)\sigma e \\
&= -\xi e
\end{aligned}$$

and so the result holds for all t by induction. Figure 2.1 shows how the iterates trail behind the minimizers.

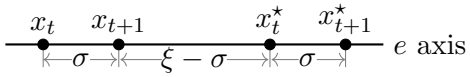


Figure 2.1: Movement of iterates and minimizers

For online gradient descent, we have $\xi = (\alpha\mu)^{-1}\sigma$; this shows that the bound in Theorem 1 is tight. We formalize this tightness result (which is a contribution of the present chapter) in Theorem 2.

Theorem 2. *For all $\alpha \in]0, 2/(\mu + L)]$ and initialization x_0, $\exists (f_t) \in \mathcal{S}(\mu/L, L, \sigma)$ such that `ALG`$(\alpha, 0, 0)$ constructs a sequence (x_t) with*

$$\limsup_{t\to\infty} \|x_t - x^\star_t\| = (\alpha\mu)^{-1}\sigma. \tag{2.7}$$

For Polyak's method, using the usual parameters, one gets $\xi = \sqrt{\kappa}\sigma$. On the other hand, for Nesterov's method one gets $\xi = (2\sqrt{\kappa} - 1)\,\sigma$. Thus, when applied to this example, the tracking iterate error scales with the square root of the condition number for both of these methods. However, as we will see in the next example, this is not always the case for these methods.

2.3.3 Example: rotating quadratic

Consider the function f utilized in the previous example, and consider rotating the first two canonical basis directions every iteration. We can reduce the full problem to one in $\mathbb{R}^2$. Define

$$f_t(x) = \frac{1}{2}x^T A_t x$$
$$A_{2t} = \begin{pmatrix} L & 0 \\ 0 & \mu \end{pmatrix}$$
$$A_{2t+1} = \begin{pmatrix} \mu & 0 \\ 0 & L \end{pmatrix}$$

and note that $(f_t) \in \mathcal{S}(\mu/L, L, \sigma)$ for all $\sigma \geq 0$. Now, let (x_t) be the sequence generated by $\texttt{ALG}(\alpha, \beta, \eta)$ given x_0. Denote $a_+ = (1+\beta) - (1+\eta)\alpha L$, $a_- = (1+\beta) - (1+\eta)\alpha\mu$, $b_+ = -\beta + \eta\alpha L$, and $b_- = -\beta + \eta\alpha\mu$. Then,

$$x_{2t} = \begin{pmatrix} a_- & 0 \\ 0 & a_+ \end{pmatrix} x_{2t-1} + \begin{pmatrix} b_- & 0 \\ 0 & b_+ \end{pmatrix} x_{2(t-1)}$$
$$x_{2t+1} = \begin{pmatrix} a_+ & 0 \\ 0 & a_- \end{pmatrix} x_{2t} + \begin{pmatrix} b_+ & 0 \\ 0 & b_- \end{pmatrix} x_{2t-1}$$
$$= \begin{pmatrix} a_+a_- + b_+ & 0 \\ 0 & a_+a_- + b_- \end{pmatrix} x_{2t-1} + \begin{pmatrix} a_+b_- & 0 \\ 0 & a_-b_+ \end{pmatrix} x_{2(t-1)}.$$

Define $z_t = [x_{2t};\ x_{2t+1}]$. Then,

$$z_t = \begin{pmatrix} b_- & 0 & a_- & 0 \\ 0 & b_+ & 0 & a_+ \\ a_+b_- & 0 & a_+a_- + b_+ & 0 \\ 0 & a_-b_+ & 0 & a_+a_- + b_- \end{pmatrix} z_{t-1}$$
$$:= Cz_{t-1}$$
$$= C^t z_0.$$

Thus, $z_t \to z^\star = 0$ precisely when $\rho(C) < 1$ (since $\rho(C) = \lim_t \|C^t\|^{1/t}$). It is easy to see that C is similar to the block-diagonal matrix with D and E on the diagonal blocks where

$$D = \begin{pmatrix} b_- & a_- \\ a_+ b_- & a_+ a_- + b_+ \end{pmatrix} \text{ and } E = \begin{pmatrix} b_+ & a_+ \\ a_- b_+ & a_+ a_- + b_-. \end{pmatrix}$$

Furthermore, D and E have the same trace and determinant:

$$\operatorname{tr}(D) = \operatorname{tr}(E) = a_+ a_- + b_+ + b_-$$

$$\det(D) = \det(E) = a_+ a_- b_- + b_+ b_- - a_+ a_- b_- = b_+ b_-.$$

Thus, $\rho(C) = \rho(D) = \rho(E)$. Note that $\rho(C) = 0$ when $\alpha = \frac{1}{L}$ and $\beta = \eta = 0$. In fact, $\texttt{ALG}(1/L, 0, 0)$ converges exactly in just two steps. Thus, online gradient descent performs better than online Polyak's method and online Nesterov's method for the rotating quadratic example. In fact, online Polyak's method actually *diverges*. For the usual parameters of Polyak's method, $\rho(C) = 6\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^2$, which is bigger than 1 precisely when $\kappa > \left(\frac{\sqrt{6}+1}{\sqrt{6}-1}\right)^2 \approx 5.7$. We state this more formally in Theorem 3.

Theorem 3. *For all $\kappa \geq 6$, $L > 0$, $\sigma \geq 0$, $\exists (f_t) \in \mathcal{S}(\kappa^{-1}, L, \sigma)$, such that online Polyak's method diverges (for any non-optimal initialization x_0).*

In the batch setting, decreasing the step-size increases robustness to noise [45, 47]. Thus, it makes sense that in the online setting, decreasing the step-size leads to greater stability. In other words, online Polyak's method diverges because of its large step-size. By reducing the stepsize to $\alpha = \frac{1}{L}$, then $\eta \leq \beta < 1$ is sufficient to guarantee $\rho(C) < 1$.

2.4 General first-order methods

In [3], Nemirovsky and Yudin proved lower bounds on the number of first-order oracle calls necessary for ϵ-convergence in both the smooth and convex setting and the smooth and strongly convex setting [3, Thm. 7.2.6]. Then, in [48], Nesterov constructed a method that achieved the lower bound in the smooth and convex setting. His method has a momentum parameter that goes

to one as the iteration count increases. By modifying the momentum sequence, it is possible to achieve the lower bound in the smooth and strongly convex setting as well. This can be done by either setting the momentum parameter to a particular constant, restarting the original method every time the iteration count reaches a particular number, or adaptively restarting the original method [31].

Nesterov presents the lower bounds in Theorems 2.1.7 and 2.1.13 respectively of [15]. These theorems involve some subtleties, which we now discuss. First, Nesterov says that (x_t) comes from a first-order method if $x_{t+1} - x_0 \in \text{span}\{\nabla f(x_0), ..., \nabla f(x_t)\}$ for all t. This is the definition that we will generalize to the time-varying setting. Second, the lower bounds do not hold for all t. The lower bound in the smooth and convex setting only holds for $t < \frac{1}{2}(d-1)$ where d is the dimension of the space. In the smooth and strongly convex setting, Nesterov only proves the bound for infinite-dimensional spaces. In fact, for finite-dimensional spaces, the lower bound can only hold for $t \leq O(d)$ since the conjugate gradient method applied to quadratic functions converges exactly in d iterations. In order to prove lower bounds that hold for all t in finite-dimensional spaces, it is necessary to restrict to smaller classes of methods. In particular, [49] excludes the conjugate gradient method by restricting to methods with "stationary" update rules. Third, the lower bounds are based on explicit adversarial functions. In particular, we will use the adversarial function that Nesterov gives in [15, 2.1.13] to construct an adversarial sequence of functions in the time-varying setting. Thus, our lower bound only holds for infinite-dimensional spaces. It is an open problem whether the function sequence can be modified to give a lower bound that holds for all t in finite-dimensional spaces or whether it is necessary to restrict to smaller classes of methods.

We say that (x_t) comes from a first-order method if $x_{t+1} - x_0 \in$ $\text{span}\{\nabla f_{\tau_0}(x_0), ..., \nabla f_{\tau_t}(x_t)\}$ where, for each t, $\tau_t \in \{0, ..., t\}$. More generally, we still consider (x_t) to be from a first-order method if (x_t) is a simple auxiliary sequence of some (y_t) that is more precisely first-order. Now, calling the most recent gradient at each step of the algorithm, as `ALG` does, corresponds to $\tau_t = t$ for all t. While it is possible for an algorithm to call an older gradient, it is not clear how this could be helpful. In section 2.4.2 will show that having $\tau_t = T\lfloor t/T \rfloor$ for all t

makes it possible to build up momentum in the online setting.

2.4.1 Universal lower bound

In Theorem 4 we give a generalization of Nesterov's lower bound for the online setting considered in this chapter. In the proof, we omit certain details that can be found in [15, Sec. 2.1.4].

Theorem 4. *Let $\mathcal{H} = \ell_2(\mathbb{N})$. For any x_0, $\exists (f_t) \in \mathcal{S}(\kappa^{-1}, L, \sigma)$ such that, if (x_t) is generated by an online first-order method starting at x_0, then*

$$\|x_t - x_t^\star\| \geq \frac{\sqrt{\kappa} - 1}{2}\sigma.$$

Proof. First, set $\gamma = \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}$ and let c be the solution to $\gamma^c = \sigma\sqrt{\frac{1+\gamma}{1-\gamma}}$, namely $c = \log\left(\sigma\sqrt{\frac{1+\gamma}{1-\gamma}}\right) / \log(\gamma)$. Note that $\frac{\gamma}{\sqrt{1-\gamma^2}} = \frac{\sqrt{\kappa}-1}{2}\kappa^{-1/4} = \frac{\sqrt{\kappa}-1}{2}\sqrt{\frac{1-\gamma}{1+\gamma}}$. Set $\mu = \kappa^{-1}L$, define A as the symmetric tridiagonal operator on $\ell_2(\mathbb{N})$ with 2's on the diagonal and -1's on the sub-diagonal, and let $a \in [\mu, L]$. Abusing notation to write the operators as matrices, define

$$f_t(x) = \frac{1}{2}x^T \left(\begin{array}{c|c} aI_t & 0 \\ \hline 0 & \frac{L-\mu}{4}A + \mu I \end{array}\right) x - \gamma^c x^T \left(\begin{array}{c} a1_t \\ \hline \frac{L-\mu}{4}e_1 \end{array}\right).$$

$$\text{Then, } \nabla f_t(x) = \left(\begin{array}{c|c} aI_t & 0 \\ \hline 0 & \frac{L-\mu}{4}A + \mu I \end{array}\right) x - \gamma^c \left(\begin{array}{c} a1_t \\ \hline \frac{L-\mu}{4}e_1 \end{array}\right)$$

$$\text{so } x_t^\star(i) = \begin{cases} \gamma^c & i \leq t \\ \gamma^{i-t+c} & i > t \end{cases}$$

$$\text{and so } \|x_{t+1}^\star - x_t^\star\|^2 = \gamma^{2c}\frac{1-\gamma}{1+\gamma}.$$

Thus, $(f_t) \in \mathcal{S}(\kappa^{-1}, L, \sigma)$. Without loss of generality, assume that $x_0 = 0$ since shifting (f_t) does not affect membership in $\mathcal{S}(\kappa^{-1}, L, \sigma)$. Then, $x_t(i) = 0\ \forall i > t$ for any first-order online algorithm.

Thus,

$$\begin{aligned}\|x_t - x_t^\star\| &\geq \left(\sum_{i=t+1}^{\infty} \gamma^{2(i-t+c)}\right)^{1/2} \\ &= \gamma^c \frac{\gamma}{\sqrt{1-\gamma^2}} \\ &= \frac{\sqrt{\kappa}-1}{2}\gamma^c\sqrt{\frac{1-\gamma}{1+\gamma}} \\ &= \frac{\sqrt{\kappa}-1}{2}\sigma.\end{aligned}$$

□

Remark 1. *If $\kappa \geq 5$, then $\frac{L-\mu}{4} \in [\mu, L]$. If we apply* ALG$(4/(L-\mu), 0, 0)$ *to the online Nesterov function with $a = \frac{L-\mu}{4}$, then it is easy to see that $\|x_t - x_t^\star\| = \frac{\sqrt{\kappa}-1}{2}\sigma$. Thus, online gradient descent with an appropriately large step-size performs optimally against the online Nesterov function.*

In the following subsection, we will construct an algorithm that performs optimally up to a fixed constant against the full class $\mathcal{S}(\kappa^{-1}, L, \sigma)$; that is, it exhibits an upper bound that is equal to the lower bound of Theorem 4 times a fixed constant.

2.4.2 Online long-step Nesterov's method

There is a conceptual difficulty when it comes to adapting accelerated methods to the online setting. Informally, in batch optimization, "acceleration" refers to the fact that accelerated methods converge faster than gradient descent. However, the goal in the online optimization framework considered here is reduced tracking error, and not necessarily faster convergence. As shown by the rotating quadratic example, tracking and convergence actually behave differently. Fortunately, we can leverage the fast convergence of Nesterov's method towards reduced tracking error.

In this section, we present a long-step Nesterov's method; the term "long-steps" refers to the fact that the algorithm takes a certain number of steps using the same stale function before catching up to the most recent function and repeating. For this particular long-step Nesterov's method, we are able to prove upper bounds on the tracking error (on the other hand, no bounds for the online Nesterov's method are yet available, and are the subject of current efforts).

The specific sequence constructed by $\texttt{OLNM}(T)$ is defined in Algorithm 1.

Algorithm 1 $\texttt{OLNM}(T)$

Require: x_0

$y_0 \leftarrow x_0$, $z_0 \leftarrow x_0$, $a_0 \leftarrow 1$
for $t = 1, 2, \ldots$ **do**
  $z_{t+1} = y_t - \frac{1}{L}\nabla f_{T\lfloor t/T \rfloor}(y_t)$
  if $T \not| \, t+1$ **then**
    $a_{t+1} = \frac{1+\sqrt{1+4a_t^2}}{2}$
    $y_{t+1} = z_{t+1} + \frac{a_t - 1}{a_{t+1}}(z_{t+1} - z_t)$
    $x_{t+1} = x_t$ ▷ so $x_{t+1} = x_t = x_{T\lfloor t/T \rfloor}$
  else if $T \mid t+1$ **then**
    $a_{t+1} = 1$
    $y_{t+1} = z_{t+1}$
    $x_{t+1} = z_{t+1}$
  end if
end for

The method can be extended to the composite case by applying the proximal operator to the z iterates. Theorem 5 gives an upper bound on the tracking iterate error of $\texttt{OLNM}$, using results from [12, Thm. 10.34].

Theorem 5. *If $(f_t) \in \mathcal{S}(\kappa^{-1}, L, \sigma)$, then, given x_0, $\texttt{OLNM}(T)$ where $T = c\sqrt{\kappa}$ for $c > 2$ such that $c\sqrt{\kappa} \in \mathbb{N}$, constructs a sequence (x_t) such that*

$$\limsup_{t\to\infty} \|x_t - x_t^\star\| \leq \frac{2c(c-1)}{c-2}\sqrt{\kappa}\sigma. \tag{2.8}$$

Proof. First, via standard batch optimization bounds, we have

$$\|x_{kT} - x^\star_{(k-1)T}\| \leq \frac{2\sqrt{\kappa}}{T}\|x_{(k-1)T} - x^\star_{(k-1)T}\|.$$

Thus,

$$\begin{aligned}
\|x_{kT} - x^\star_{kT}\| &\leq \frac{2\sqrt{\kappa}}{T}\|x_{(k-1)T} - x^\star_{(k-1)T}\| + T\sigma \\
&\leq \left(\frac{2\sqrt{\kappa}}{T}\right)^k \|x_0 - x_0^\star\| + \frac{T\sigma}{1 - \frac{2\sqrt{\kappa}}{T}} \\
&= \left(\frac{2\sqrt{\kappa}}{T}\right)^k \|x_0 - x_0^\star\| + \frac{T^2\sigma}{T - 2\sqrt{\kappa}}
\end{aligned}$$

and so

$$\begin{aligned}\|x_t - x_t^\star\| &= \|x_{T\lfloor t/T\rfloor} - x_t^\star\| \\ &\leq \|x_{T\lfloor t/T\rfloor} - x_{T\lfloor t/T\rfloor}^\star\| + \|x_t^\star - x_{T\lfloor t/T\rfloor}^\star\| \\ &\leq \left(\frac{2\sqrt{\kappa}}{T}\right)^{\lfloor t/T\rfloor} \|x_0 - x_0^\star\| + \frac{T^2\sigma}{T - 2\sqrt{\kappa}} + T\sigma.\end{aligned}$$

Then, taking the limit supremum, we get

$$\begin{aligned}\limsup_{t\to\infty} \|x_t - x_t^\star\| &\leq \frac{T^2\sigma}{T - 2\sqrt{\kappa}} + T\sigma \\ &= \frac{2T\sigma\left(T - \sqrt{\kappa}\right)}{T - 2\sqrt{\kappa}} \\ &= \frac{2c(c-1)}{c-2}\sqrt{\kappa}\sigma.\end{aligned}$$

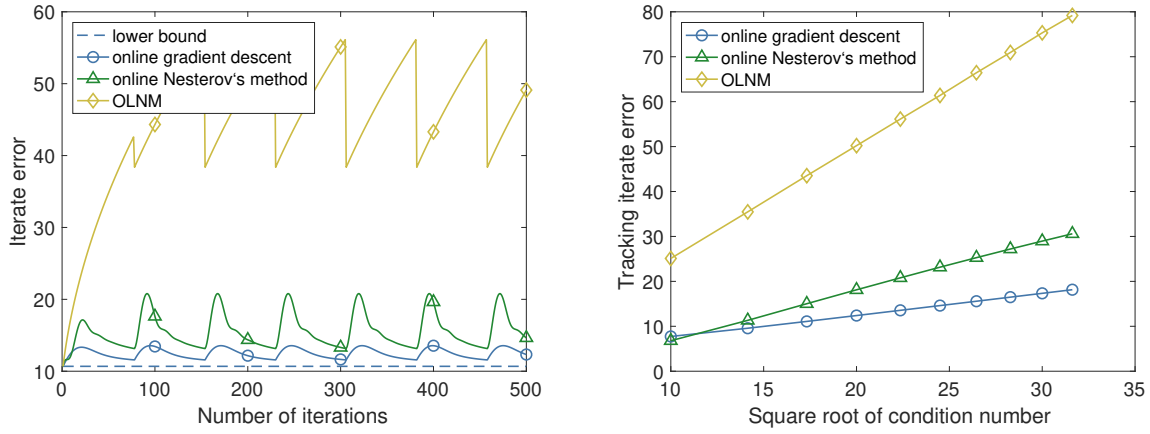
□

If we minimize the bound in Eq. (2.8) over $c \in \mathbb{R}$, then we get $c = 2 + \sqrt{2}$ with a value of $2c(c-1)/(c-2) = 6 + 4\sqrt{2} \approx 11.66$; this is in contrast with the batch setting, where $c = \sqrt{8}$. However, we have the extra restriction that $c\sqrt{\kappa} \in \mathbb{N}$ so, in general, we will take $T = \lfloor (2+\sqrt{2})\sqrt{\kappa} \rfloor$.

Note that the bound is asymptotically (as the condition number goes to infinity) optimal (hence tight) up to the constant $4c(c-1)/(c-2)$. In particular, for $\kappa \geq (4c(c-1)/(c-2))^2$, the bound is better than the bound for online gradient descent. However, these are bounds over a general class of functions. One question is how the two methods fare against specific examples, as exemplified next.

As we noted in Remark 1, online gradient descent with an appropriately large step-size performs optimally against the online Nesterov function. Even with a typical step-size, the tracking iterate error of online gradient descent still scales linearly with the square root of the condition number. Furthermore, while `OLNM` also scales linearly with the square root of the condition number, online gradient descent has a smaller constant. Figure 2.2(a) depicts the iterate error for algorithms applied to the online Nesterov function with condition number equal to 500. Online gradient descent performs the best, followed by online Nesterov's method and `OLNM`. In fact, this can be seen in the

dependence on the condition number as well. Figure 2.2(b) shows the linear dependence of the tracking iterate error on the square root of the condition number. The constant for online gradient descent is 0.481, the constant for online Nesterov's method is 1.101, and the constant for `OLNM` is 2.491. Note that, despite `OLNM` having a worse constant than online gradient descent, the former's constant is still less than its upper bound of $2c(c-1)/(c-2) \approx 11.66$.

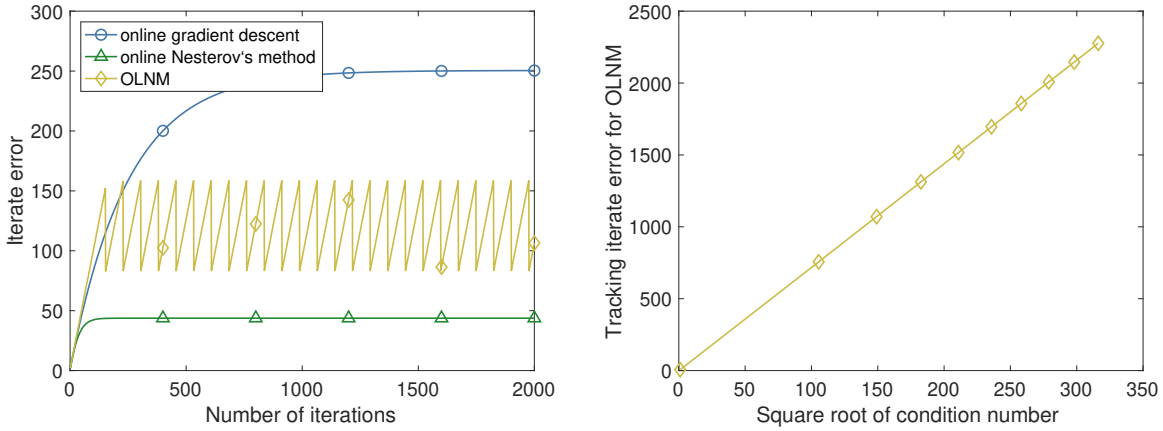


(a) (b)

Figure 2.2: Algorithms applied to the online Nesterov function with $L = 500$, $d = 1000$, $a = (L+\mu)/2$, $\sigma = 1$, and $T = \lfloor (2+\sqrt{2})\sqrt{\kappa} \rfloor$. (a) Evolution of the iterate error for the particular example $\mu = 1$. (b) Tracking iterate error for varying μ.

For the rotating quadratic example, the minimizer is fixed, so `OLNM` has the same convergence rate as Nesterov's method does in the batch setting. However, with the right step-size, online gradient descent can converge in just two steps! On the other hand, for the translating quadratic example, since the tracking iterate error of online gradient descent scales with κ, while the tracking iterate error of `OLNM` scales with $\sqrt{\kappa}$, `OLNM` outperforms online gradient descent for sufficiently high condition number. In particular, Figure 2.3(a) depicts the iterate error for algorithms applied to the translating quadratic function with condition number equal to 500. Online Nesterov's method performs the best, followed by `OLNM` and online gradient descent. Figure 2.3(b) shows the tracking iterate error for varying condition number for `OLNM` (online gradient descent and online Nesterov's

method are left out since we analytically solved for their tracking iterate error). The constant is 7.21. Note that this is larger than the constant for `OLNM` applied to the online Nesterov function. While the online Nesterov function is a universal adversary, the translating quadratic is more particularly adversarial for `OLNM` because the minimizer is maximizing its distance away from the `OLNM` iterates by moving in a straight line away from the old minimizer the `OLNM` iterates are approaching. Also note that this is more than half of the upper bound, showing that the upper bound is tight at least up to a constant less than two.



(a) (b)

Figure 2.3: Algorithms applied to the translating quadratic function with $d = 2$, $\sigma = 1$, and $T = \lfloor (2+\sqrt{2})\sqrt{\kappa} \rfloor$. (a) shows the evolution of the iterate error for $L = 500$ and $\mu = 1$. (b) shows the tracking iterate error for $L = 1$ and varying μ.

2.5 Regularization

When the functions are convex, but not strongly convex, performance metrics for online first-order methods typically rely on dynamic regret bounds. The dynamic regret is the *averaged* function error. We take a different approach, however, deriving a gradient error bound via regularization. While gradient error bounds are weaker than function error bounds, the benefit is that we bound the error, rather than the averaged error.

The main idea is that there is a trade-off between tracking error and regularization error. Regularizing by any amount means that we can apply Theorem 1. In this case, increasing the amount of regularization increases the regularization error while also decreasing the tracking error. In the following, we provide a framework for balancing these two errors.

Given x_0, let **online regularized gradient descent,** $\texttt{ORGD}(\delta, x_c)$, with $\delta > 0$, construct the sequence (x_t) via

$$x_{t+1} = x_t - \frac{2}{L + 2\delta}(\nabla f_t(x_t) + \delta(x_t - x_c)). \qquad (\texttt{ORGD}(\delta, x_c))$$

It is easy to see that $\texttt{ORGD}(\delta, x_c)$ is vanilla online gradient descent for the regularized problem $f_t(\cdot; \delta, x_c) = f_t + \frac{\delta}{2}\|\cdot - x_c\|^2$. Since we don't vary x_c in the analysis, we write $f_t(\cdot; \delta)$ for simplicity.

Now, in order to bound the algorithm error of ORGD, we need to bound the regularization error. As with the algorithm error, we can measure the regularization error in terms of the variable, the cost, or the gradient. Unfortunately, it is impossible, without further assumptions, to bound the variable regularization error, $\|x^\star - x_r^\star\|$ where $x_r^\star$ is the unique minimizer to the regularized problem [50, 51]. For example, if we regularize a constant function by any amount, then there are minimizers arbitrarily far away from the regularized minimizer. However, if we assume the function is coercive ($\|x\| \to \infty \implies f(x) \to \infty$), then we know the variable regularization error *is* bounded. But, it is still impossible to bound it in terms of the strong smoothness constant. For example, assume that we are going to regularize by adding $\frac{\delta}{2}\|\cdot\|^2$. Then, for an arbitrary distance ξ, there exists an L-smooth function such that the variable regularization error is greater than ξ. In particular, consider $f(x) = \frac{\lambda}{2}\|x - 2\xi e\|^2$ where $\lambda \leq \min\{L, \delta\}$ and e is a unit vector. Then $x_r^\star = \frac{\lambda}{\lambda+\delta}e$ and so $\|x^\star - x_r^\star\| = \frac{\delta}{\lambda+\delta}\|2\xi e\| \geq \xi$.

Fortunately, even without coercivity, it *is* possible to bound the gradient regularization error [15, Thm. 2.2.7], which allows us to bound the tracking gradient error of ORGD. However, we do have to make an additional assumption. Loosely, we have to assume that the sequence of minimizer sets doesn't "drift." To make the assumption more precise, we need some definitions.

First, let $x_t^\star(\delta) = \operatorname{argmin}_x f_t(x; \delta)$ for $\delta > 0$. We know the right-hand side is a singleton by

strong convexity. Furthermore, since f_t is proper, closed, and convex, ∂f_t is maximally monotone, and so we can apply [13, Thm. 23.44], which tells us that $x_t^\star(\cdot)$ is continuous and $\lim_{\delta\to 0} x_t^\star(\delta)$ is the unique projection of x_c onto the zero set of ∂f_t. Thus, we can define $x_t^\star(0) = \lim_{\delta\to 0} x_t^\star(\delta)$. We also have that $\delta \mapsto \|x_t^\star(\delta) - x_c\|$ is monotonically decreasing (this is not hard to show and can be found in the proof of [15, Thm. 2.2.7]). Let $R(\delta; x_c) = \sup_{t\in\mathbb{N}\cup\{0\}} \|x_t^\star(\delta) - x_c\|$ for all $\delta \geq 0$. Again, since we do not vary x_c in the analysis, we write $R(\delta)$ for simplicity. Note that $R(\cdot)$ is also monotonically decreasing. Thus, if $R(0) < \infty$, then $R(\delta) < \infty$ for all $\delta \geq 0$. We will assume this is true:

$$R(0) < \infty. \tag{bounded drift}$$

While this assumption precludes problems like the translating quadratic, it is realistic when the problem is data-dependent. For machine learning problems it is common to have normalized data and to be learning normalized weights. Then, the minimizer will, in fact, lie in a bounded set.

Let $\sigma(\delta) = \sup_{t\in\mathbb{N}} \|x_t^\star(\delta) - x_{t-1}^\star(\delta)\|$ for $\delta \geq 0$. Note that $\sigma(\delta)$ is bounded above by $2R(\delta)$, via the triangle inequality, which in turn is bounded above by $R(0) < \infty$, via the monotonicity of $R(\cdot)$.

Finally, consider the function $h(\delta) = \frac{\sigma(\delta)}{R(\delta)} - 2\left(\frac{\delta}{L}\right)^2$ for $\delta \geq 0$. $h(\cdot)$ is continuous since $x_t^\star(\cdot)$ is continuous and $R(\delta) > 0$ for all $\delta \geq 0$ (unless $x_c = x_t^\star(0)$, which would be the trivial case). Also, $h(0) = \frac{\sigma(0)}{R(0)} > 0$ and $h(L) \leq 0$. Thus, via the Intermediate Value Theorem, we have just proved the following lemma.

Lemma 1. *If $(f_t) \in \mathcal{S}'(L)$ has bounded drift, then $\exists\ 0 < \delta \leq L$ s.t. $\delta = L\sqrt{\frac{\sigma(\delta)}{2R(\delta)}}$.*

In Theorem 6, we derive a bound on the tracking gradient error in terms of $\sigma(\delta)$ and $R(\delta)$. However, since $\sigma(\delta)$ and $R(\delta)$ both depend on δ, it is impossible, without further information about the function sequence, to minimize the bound explicitly with respect to δ. The δ in Lemma 1 corresponds to what the minimizing δ would be if $\sigma(\delta)$ and $R(\delta)$ were constant.

Theorem 6. *If $(f_t) \in \mathcal{S}'(L)$ has bounded drift, then $\exists\ 0 < \delta \leq L$ such that, given x_0, `ORGD`(δ, x_c) constructs a sequence (x_t) with*

$$\limsup_{t\to\infty} \|\nabla f_t(x_t)\| \leq 2\sqrt{2}L\sqrt{\sigma(\delta)R(\delta)}. \tag{2.9}$$

Proof. Let δ be as in Lemma 1. Observe,

$$\begin{aligned} 0 &= \nabla f_t(x_t^\star(\delta); \delta) \\ &= \nabla f_t(x_t^\star(\delta)) + \delta(x_t^\star(\delta) - x_c) \end{aligned}$$

so

$$\|\nabla f_t(x_t^\star(\delta))\| = \delta \|x_t^\star(\delta) - x_c\|.$$

Thus,

$$\begin{aligned} \|\nabla f_t(x_t)\| &\leq \|\nabla f_t(x_t^\star(\delta))\| + \|\nabla f_t(x_t) - \nabla f_t(x_t^\star(\delta))\| \\ &\leq \delta \|x_t^\star(\delta) - x_c\| + L\|x_t - x_t^\star(\delta)\| \\ &\leq \delta R(\delta) + L\|x_t - x_t^\star(\delta)\| \end{aligned}$$

so

$$\begin{aligned} \limsup_{t\to\infty} \|\nabla f_t(x_t)\| &\leq \delta R(\delta) + \frac{L(L+2\delta)\sigma(\delta)}{2\delta} \\ &= L\sqrt{2\sigma(\delta)R(\delta)} + L\sigma \\ &\leq L\sqrt{2\sigma(\delta)R(\delta)} + L\sqrt{2\sigma(\delta)R(\delta)} \\ &= 2\sqrt{2}L\sqrt{\sigma(\delta)R(\delta)}. \end{aligned}$$

□

Note that if $\sigma(0) \approx \sigma(\delta) \approx R(\delta) \approx R(0)$ then the bound in Eq. 2.9 is $\approx 2\sqrt{2}LR(0)$, which is of the same form but has worse constants than the bound for the algorithm which abstains from tracking (i.e., $\forall t\ x_t = x_c$):

$$\begin{aligned} \|\nabla f_t(x_t)\| &= \|\nabla f_t(x_c)\| \\ &= \|\nabla f_t(x_c) - \nabla f_t(x_t^\star(0))\| \\ &\leq L\|x_c - x_t^\star(0)\| \\ &\leq LR(0). \end{aligned}$$

Conversely, if $\sigma(\delta) \ll R(\delta)$, then δ is small so $\sigma(0) \approx \sigma(\delta)$ and $R(0) \approx R(\delta)$. In this case, Eq. 2.9 becomes $\approx 2\sqrt{2}\sqrt{\frac{\sigma(0)}{R(0)}}LR(0) \ll LR(0)$. Thus, we can only guarantee the usefulness of `ORGD` when $\sigma(\delta) \ll R(\delta)$.

2.6 Illustrative numerical examples

Our code is online at github.com/liammadden/time-varying-experiments. In this section, we consider time-varying least-squares regression and time-varying logistic regression. At each $t \in \mathbb{N} \cup \{0\}$, we are given an input matrix $A^{(t)} \in \mathbb{R}^{n \times d}$, with i-th row vector denoted by $a_i^{(t)}$, and an n-dimensional output vector $b^{(t)}$, where each $(a_i^{(t)}, b_i^{(t)})$ corresponds to a single data point. For least-squares regression, $b^{(t)} \in \mathbb{R}^n$. For logistic regression, $b^{(t)} \in \{-1, 1\}^n$. For simplicity, we assume the inputs are constant across time, i.e. $A_t = A$ for all t, while only the outputs change. This type of time-variation fits applications with time-series data where the time-dependency is not captured by any of the input features. For example, a problem may have a discrete number of states that it switches between based on a hidden variable.

The cost function for least-squares regression is

$$\begin{aligned} f_t(x) &= \frac{1}{2}\sum_{i=1}^{n}\left(\langle a_i, x\rangle - b_i^{(t)}\right)^2 \\ &= \frac{1}{2}\|Ax - b^{(t)}\|^2 \end{aligned}$$

and for logistic regression is

$$f_t(x) = \frac{1}{n}\sum_{i=1}^{n}\log\left(1 + \exp(-b_i^{(t)}\langle a_i, x\rangle)\right).$$

The least-squares cost function is strongly convex while the logistic cost function is only strictly convex. However, it can be shown that gradient descent still achieves linear convergence when applied to the logistic cost function [52].

A "weight vector," $x \in \mathbb{R}^d$, predicts an output, b, from an input, a. For least-squares regression, $b = \langle a, x\rangle$. For logistic regression, $b = \text{sign}(\langle a, x\rangle)$; more precisely, x predicts $b = 1$ with

probability $(1 + \exp(-\langle a, x\rangle))^{-1}$ and $b = -1$ with the remaining probability. The minimizer, $x_t^\star$, of the respective objective function is the "optimal" weight vector.

2.6.1 Least squares regression

For the least-squares regression we considered the case $n = 20$ and $d = 5$. We generated A by defining its singular value decomposition. For its left and right-singular vectors, we sampled two Haar distributed orthogonal matrices, $U \in \mathbb{R}^{n\times n}$ and $V \in \mathbb{R}^{d\times d}$. We let its singular values be equally spaced from $1/\sqrt{\kappa}$ to 1. We generated b_t by varying $x_t^\star$ via a random walk with $\sigma = 1$ and adding a low-level of Gaussian noise (mean, 0; standard deviation, 10^{-3}) to the corresponding outputs. We initialized $x_0^\star$ as the vector of ones. For each κ, we ran the experiment 200 times and took the average of the tracking errors.

For `OLNM`, we set $T = \lfloor(2 + \sqrt{2})\sqrt{\kappa}\rfloor$. However, instead of updating the x iterate by the z iterate only every T iterations, we updated it every iteration. Doing so works better in practice, though we lose the theoretical guarantees of Theorem 5. However, the theory only requires that the x iterates do not move away from the minimizer that the z iterates are converging to. Not updating the x iterates ensures they do not move away, but it is too conservative in practice.

For all the algorithms, we initialized $x_0 = x_0^\star$ so that we wouldn't need a long "burn-in" period while waiting for convergence to the tracking error. We computed the tracking error as the maximum error over the 5th cycle of T. Figure 2.4 shows the results. `OLNM` performs better than online gradient descent, however, online Nesterov's method performs much better than both of the other methods, despite its lack of guarantees.

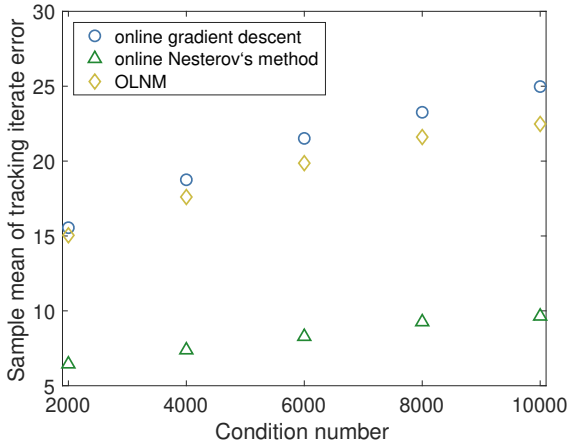


Figure 2.4: Least-squares regression with random data matrix and random walk variation of the minimizer.

2.6.2 Logistic regression with streaming data

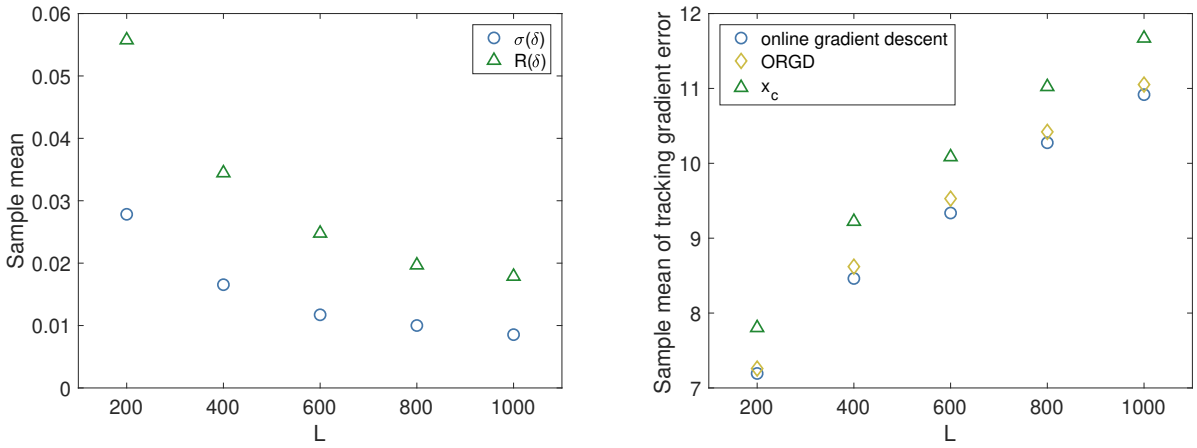
For logistic regression we again considered the case $n = 20$ and $d = 5$. We generated A in the same way as for least-squares regression, but with singular values drawn from the uniform distribution on (0,1) and then scaled to have maximum singular value equal to $2\sqrt{L}$ (since the strong smoothness constant of the cost function is $\|A\|_2^2/4$). We initialized b_t from the Rademacher distribution and uniformly at random chose one label to flip each time step.

A classic problem that logistic regression is used for is spam classification. The problem is to predict whether an email is spam or not based on a list of features. Since emails are received in a streaming fashion, this is a time-varying problem. Thus, when an email first arrives, we may report that it is spam, but then later realize it is not, or vice versa. Such an extension of spam classification can be incorporated into the time-varying logistic regression framework.

We let x_c be the zero vector. As a weight vector, x_c predicts 1 or -1 with equal probability. For each L, we approximated the solution to the fixed point equation $\delta = L\sqrt{\frac{\mathbb{E}(\sigma(\delta))}{\mathbb{E}(R(\delta))}}$ where the expectation is taken over A and the b_t's. Figure 2.5(a) shows how $\sigma(\delta)$ and $R(\delta)$ vary with L. The ratio between $\sigma(\delta)$ and $R(\delta)$ stays constant at ≈ 0.5.

We ran the experiment 200 times and took the average of the tracking errors. We calculated

the tracking error using windows of 100 iterations and stopping when the maximum error over the last two windows was the same as over the last window. Figure 2.5(b) shows the results. Online gradient descent (which lacks theoretical tracking gradient error bounds) performs slightly better than `ORGD` and both outperform x_c. Thus, with only one label flipping every time step, it is still possible to track the solution. However, if we sufficiently increased the number of labels that flip every time step, then it would be better to just guess the labels, as x_c does.



(a) (b)

Figure 2.5: Logistic regression with random data matrix and randomly flipping labels.

2.7 Conclusions

By categorizing classes of functions based on the minimizer variation, this chapter was able to generalize results from batch optimization to the online setting. These results are important for time-varying optimization, especially for applications in power systems, transportation systems, and communication networks. We showed that fast convergence does not necessarily lead to small tracking error. For example, online Polyak's method can diverge even when the minimizer variation is zero. On the other hand, online gradient descent is guaranteed to have a tracking error that does not grow more than linearly with the minimizer variation. We also gave a universal lower bound

for online first-order methods and showed that `OLNM` achieves it up to a constant factor. It is a future research direction to consider more deeply the connection to the long-steps of interior-point methods and see if a satisfactory criteria can be found for adaptively deciding when to long-step. Perhaps, this enquiry may eventually lead to error bounds for online Nesterov's method itself.

Chapter 3

Best approximate quantum compiling problems

3.1 Introduction

With the steady advances in quantum hardware and volume [53], quantum computing is well on track to become widely adopted in science and technology in the near future. One of the core challenges to enable its use is the availability of a flexible and reliable quantum compiler, which can translate any target quantum circuit into a circuit that can be implemented on real hardware with gate set, connectivity, and length limitations.

Since the celebrated Solovay-Kitaev theorem [54, 55], quantum compiling has been a rich research area. Works have investigated how to efficiently map different gates into canonical (universal) gate sets up to an arbitrary accuracy [56, 57, 58, 59, 60, 61, 62, 63], or how to "place" the target circuit onto the real connectivity-limited hardware [64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78]

A more holistic strategy in quantum compiling has been the construction of universal parametric circuits to serve as templates to compile any target circuit. This line of research, which we will call decomposition-based, focuses primarily on templates based on the versatile CNOT+rotation gate set [79, 80, 81, 82, 83, 84, 85]. These works unveiled fundamental lower bounds on the number of CNOTs that almost all target circuits require in order to be compiled in such a gate set and delivered a constructive method for doing so, the quantum Shannon decomposition (QSD), which was only a factor of two off from the lower bound of efficiency. This research is formalized mainly in the language of Lie theory as a recursive sequence of Cartan decompositions. Despite the constructive and sound theory, the QSD decomposition does not have the flexibility of trading off precision and

length.

The quantum compiling problem in its essence can be cast as an optimization problem in the space of unitary matrices. Here one needs to find a unitary matrix that can be realized in hardware (with various constraints, e.g., gate set and connectivity) that is the "closest" to a target unitary (i.e., the circuit that one wants to realize, or compile). Here "closest" is intended with respect to a pertinent metric. On the one hand, this optimization problem could encompass the whole quantum compiling research; on the other hand, it is a very difficult mathematical problem and even for a small number of qubits cannot be stored in memory. Recently, a series of papers [86, 2], have revised this optimization-based compiling approach with some simplifying assumptions and heuristics, and have introduced the idea of computing the cost and its gradients in a quantum-assisted way. With the same optimization lens, but with other classical heuristics, the works [87, 88] have looked at hierarchical compilations, whereby one need not compile a unitary directly to a two-qubit gate circuit, but can instead start with higher-qubit gates, and then continue recursively to the two-qubit gate target. Finally, the recent works [89, 90] arrange the CNOTs to explicitly decouple the qubits one at a time using a particular cost function. Moreover, they provide numerical evidence that their approach can compile arbitrary target circuits very close to the lower bound on the number of CNOTs.

In this chapter, we aim at analyzing the optimization approach in more depth and offering some sound evidence on the justification of critical assumptions and solution methods. Further, we aim at helping to bridge the gap between Cartan decomposition-based research and optimization-based compiling. Our approach consists in formulating best approximate compiling problems, which trade-off exact compilation with constraint violation. In particular, we offer the following contributions.

- We start by analyzing the quantum compiling mathematical problem in the CNOT+rotation gate set and show how to construct versatile "CNOT units," i.e., elementary building blocks, that can be used to parametrize any circuit of a given number of qubits;

- We show that optimizing over the parametrized circuit consists of optimizing the structure (i.e., where to place the CNOT units) and optimizing the rotation angles of the rotation gates. We show that the former is largely unimportant once past the so-called surjectivity bound (even when imposing hardware constraints), while the latter is easy from an optimization perspective, by using e.g., Nesterov's accelerated gradient descent [48, 91];

- With the intention to further compress the compiled circuit, allowing for approximation errors in terms of gate fidelity, we propose a novel regularization based on group LASSO [92], which (among other things) can reduce the length of the QSD down to the CNOT theoretical lower bound without affecting fidelity noticeably (i.e., a factor of two compression);

- Various numerical results support our findings. In particular, we showcase how to use our approach to discover new decompositions in the CNOT+rotation gate set of special gates (e.g., the Toffoli gate) and how to use the compression mechanism as an extension of any compilation code available (e.g., Qiskit transpile [93]) that can trade-off accuracy for circuit *length*, where length is the number of CNOTs.

While discussing the main contributions, the chapter focuses on three complementary goals,

[G1] Approximate quantum compiling for random unitary matrices: here the goal is to derive results in terms of the number of CNOTs to use to compile any given random unitary matrix, with connectivity constraints;

[G2] Approximate quantum compiling for special gates: here the goal is to derive results in terms of the number of CNOTs to use to compile special gates;

[G3] Approximate quantum compiling for circuit compression: here the goal is to derive results that allow for compression of circuits, freeing the possibility to have inexact compilation.

Organization. This chapter is organized as follows. In Section 3.2, we report the mathematical and physical preliminaries to our algorithmic development. In Section 3.3, we

formalize the approximate quantum compiling problem as a mathematical program. In Section 3.4, we devise a programmable unit (the two-qubit CNOT unit) with which we can build any circuit. In Section 3.5, we discuss the property of the mathematical program introduced in Section 3.3 when specified for the parametric circuit presented in Section 3.4.

From Section 3.6 on, we focus on particular layout patterns, namely the sequential, spin, and Cartan, and we present their properties. In Section 3.7, we discuss the use of gradient descent to optimize the rotation angles once the layout is fixed, and we showcase numerical results in Sections 3.7.2-3.7.3. Section 3.8 discusses our proposed compression strategy to trade-off accuracy and length, both theoretically and numerically. We then conclude in Section 3.9.

3.2 Preliminaries

We work with n-qubit quantum circuits, which are represented either by a collection of ordered gate operations, or by a d by d unitary matrix with $d = 2^n$. The class of unitary matrices of dimension $d \times d$ together with the operation of matrix multiplication have an important group structure [94], denoted as the unitary group $\mathrm{U}(d)$. In particular, the group is a Lie group of dimension d^2 as a real manifold; we let $\mathfrak{u}(d)$ be its Lie algebra, which consists of anti-Hermitian matrices. Unitary matrices with determinant equal to 1 are called special unitary matrices. Special unitary matrices of dimension $d \times d$ together with the operation of matrix multiplication form the the special unitary group of degree d: $\mathrm{SU}(d)$, which is a Lie group of dimension $d^2 - 1$ as a real manifold. We let $\mathfrak{su}(d)$ denote its Lie algebra, which consists of traceless anti-Hermitian matrices.

A useful property of the determinant $\det(\cdot)$ of any squared d by d matrix A and scalar c is that $\det(cA) = c^d A$. Hence if $U \in \mathrm{U}(d)$ then $U/\det(U)^{1/d} \in \mathrm{SU}(d)$. Since scalars (e.g., global phases in quantum computing) are easy to implement on a quantum computer (and in fact unimportant), we normalize unitary matrices as above, and so we only need to know how to "work with" special unitary matrices. We remark that if $U \in \mathrm{SU}(d)$, then adding any global phase multiple of $2\pi/d$ does not alter the matrix determinant, i.e., $e^{i\frac{2\pi m}{d}} U \in \mathrm{SU}(d)$ as well for any integer $m \in \mathbb{Z}$. The equivalence classes from this relation form the projective special unitary group, $\mathrm{PSU}(d)$, which is

isomorphic to the projective unitary group, $\mathrm{PU}(d) \coloneqq \mathrm{U}(d)/\mathrm{U}(1)$. Thus, compiling a matrix from $\mathrm{SU}(d)$ also provides a compilation for its $d-1$ equivalent matrices in $\mathrm{PSU}(d)$.

A single-qubit gate on the jth qubit is a unitary matrix of the form $I_{2^{j-1}} \otimes u \otimes I_{2^{n-j}}$ where $u \in \mathrm{U}(2)$, I_q is the identity matrix of dimension q, and $\otimes$ represents the Kronecker product. An important set of matrices in $\mathrm{U}(2)$ is the set of Pauli matrices, which we denote with their usual notation as X, Y, Z, which are unitary, Hermitian, traceless, and have determinant equal to -1. From the Pauli matrices, one obtains the rotation matrices $R_x(\theta), R_y(\theta), R_z(\theta)$ by matrix exponentiation. The rotation matrices are special unitary. An important fact that we use extensively in this chapter is that any $u \in \mathrm{SU}(2)$ can be written as a product of any three rotation matrices with no consecutive two the same [54].

Among multiple-qubit gates, we focus on the two-qubit CNOT gate, or controlled-X gate, with control qubit j and target qubit k, which is the matrix

$$\mathrm{CNOT}_{jk} = I_{2^{j-1}} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \otimes I_{2^{n-j}} + \begin{cases} I_{2^{j-1}} \otimes \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \otimes I_{2^{k-j-1}} \otimes X \otimes I_{2^{n-k}} & \text{if } j < k \\ I_{2^{k-1}} \otimes X \otimes I_{2^{j-k-1}} \otimes \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \otimes I_{2^{n-j}} & \text{if } k < j \end{cases}.$$

For example, for $n = 2$, we have

$$\mathrm{CNOT}_{12} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \qquad \mathrm{CNOT}_{21} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

Note that for $n = 2$, CNOT has determinant -1, and so we have to normalize it. On the other hand, for $n > 2$, CNOT has determinant 1.

3.3 Approximate quantum compiling as mathematical optimization

We are interested in compiling a quantum circuit, which we formalize as finding the "best" circuit representation in terms of an ordered gate sequence of a target unitary matrix $U \in \mathrm{U}(d)$, with some additional hardware constraints. In particular, we look at representations that could be constrained in terms of hardware connectivity, as well as circuit length, and we choose a gate basis in terms of CNOT and rotation gates. The latter choice is motivated by an implementation in the Qiskit software package [93]. We recall that the combination of CNOT and rotation gates is universal in $\mathrm{SU}(d)$ and therefore it does not limit compilation (i.e. we can compile everything with only CNOTs and rotations) [54].

To properly define what we mean by "best" circuit representation, we define the metric as the Frobenius norm between the unitary matrix of the compiled circuit V and the target unitary matrix U, i.e., $\|V - U\|_{\mathrm{F}}$. This choice is motivated by mathematical programming considerations, and it is related to other formulations that appear in the literature (see Remark 2).

We are now ready to formalize the approximate quantum compiling problem as follows.

Given a target special unitary matrix $U \in \mathrm{SU}(2^n)$ and a set of constraints, in terms of connectivity and length, find the closest special unitary matrix $V \in \mathcal{V} \subseteq \mathrm{SU}(2^n)$, where $\mathcal{V}$ represents the set of special unitary matrices that can be realized with rotations and CNOT gates alone and satisfy both connectivity and length constraints, by solving the following mathematical program:

$$\textbf{(AQCP)} \qquad \min_{V \in \mathcal{V} \subseteq SU(2^n)} f(V) := \frac{1}{2}\|V - U\|_{\mathrm{F}}^2. \tag{3.1}$$

Note that the cost function can be equivalently written

$$\begin{aligned} \frac{1}{2}\|V - U\|_F^2 &= \frac{1}{2}\mathsf{Tr}[U^\dagger U - U^\dagger V - V^\dagger U + V^\dagger V] = \frac{1}{2}\mathsf{Tr}[2I] - \frac{1}{2}\mathsf{Tr}[U^\dagger V] - \frac{1}{2}\mathsf{Tr}[V^\dagger U] \\ &= d - \mathbb{R}\mathsf{e}\ \mathsf{Tr}[U^\dagger V] \end{aligned} \tag{3.2}$$

where we used that U and V are unitary matrices.

We call (3.1) the approximate quantum compiling (master) problem (AQCP). A solution of the problem is an optimal V indicated as V^*, along with an ordered set of gate operations that respect the constraints.

If V^* is such that the cost is null, then we say that the compilation is exact, otherwise it is approximate and the approximation error is computed by the cost $\frac{1}{2}\|V^* - U\|_{\mathrm{F}}^2$. Without further specifications (and simplifications), Problem (3.1) is intractable but for very small circuits. How to efficiently solve (3.1) is our aim.

Remark 2. *In [2], a different metric was used, namely the Hilbert-Schmidt test defined as*

$$C_{HST}(U,V) = 1 - \frac{1}{d^2}|\mathsf{Tr}[V^\dagger U]|^2 = \frac{d+1}{d}(1 - \bar{F}(U,V)),$$

where $\bar{F}(U,V)$ is the fidelity averaged over the Haar distribution. Given that $f(V) = \frac{1}{2}\|V - U\|_{\mathrm{F}}^2 = d - \mathbb{R}\mathsf{eTr}[V^\dagger U]$, then $\mathbb{R}\mathsf{eTr}[V^\dagger U] = d - f(V)$. Hence,

$$\bar{F}(U,V) = 1 - \frac{d}{d+1}C_{HST}(U,V) = 1 - \frac{d}{d+1} + \frac{1}{d(d+1)}((\mathbb{R}\mathsf{eTr}[V^\dagger U])^2 + (\mathbb{I}\mathsf{mTr}[V^\dagger U])^2) \geq$$
$$1 - \frac{d}{d+1} + \frac{1}{d(d+1)}(d - f(V))^2 =: \bar{F}_{\mathrm{F}}(U,V),$$

where we have defined the new quantity $\bar{F}_{\mathrm{F}}(U,V)$ as the Frobenius fidelity, which is always lower than the $\bar{F}(U,V)$. By minorization arguments, minimizing $f(V)$ has the effect of maximizing the Frobenius fidelity.

The stepping stones of our work are the papers [86, 2]. There, the AQCP is approached by a bi-level technique. Since V needs to be sought in the space of matrices that can be realized with rotation and CNOT gates, one can solve for V by interleaving an optimization on the structure (at fixed length), i.e., which gate needs to be implemented where, and an optimization on the rotation angles. The first problem (deciding the structure) is combinatorial and non-convex, and [86, 2] propose a meta-heuristic based on simulated-annealing and compactifications, while the second problem (deciding the rotation angles) is continuous and non-convex, and [86, 2] include a gradient descent algorithm. [86, 2] also offer an algorithm to fix the CNOT structure, more or less arbitrarily,

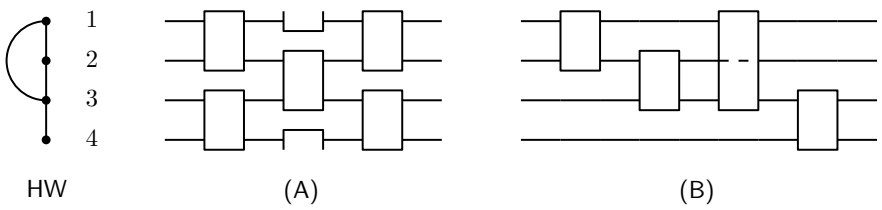


Figure 3.1: Possible two-qubit gate units structures. HW depicts the hardware connectivity of a four-qubit quantum computer. (A) the structure, or ansatz, proposed in [2], where each block represents a CNOT gate with rotation gates before and after, is easily implementable but may miss some hardware connections or introduce some that are not there; (B) a possible sequential structure, which alternates among all the possible two-qubit blocks and could capture hardware connectivity better.

and optimize only for rotation angles, which is more efficient in terms of computation time, but less optimal.

Despite the encouraging results, key challenges in these works remain. First, optimizing the structure via simulated-annealing is far from optimal and it can be very time consuming. But most importantly it is physics-agnostic, therefore one may wonder if one could do better by using physics.

Second, fixing the structure, as is done in [2], is by no means universal nor does it mirror the hardware connectivity; a better structure could be a sequential structure (see Figure 3.1).

Third, the gradient descent algorithm used in [2] seems to work surprisingly well, despite the non-convexity of the problem at hand, and one may wonder why this is so, which is a non-trivial question.

The rest of this chapter answers these questions by using a pertinent parametrization of the matrix V in terms of CNOTs and rotation gates.

3.4 A programmable two-qubit gate and the CNOT unit

We start by defining a flexible two-qubit gate, which is parametrized by four rotation angles, and which will be the building block of the parametric circuit.

Let us focus for now on basis set constraints alone. Given the target special unitary $U \in \mathrm{SU}(d)$, compiling means to find a product of matrices from the basis set (as said CNOT+rotations) that

either approximates or equals U. We define the "length" of the compilation as the number of CNOTs in the product. In 2004, Shende, Markov, and Bullock derived a universal lower bound on the length for exact compilation in this setting [81]. The bound specifies a **sufficient** minimal number of CNOTs for exact compilation, for **all** the matrices in $\mathrm{SU}(d)$. And, while a particular special unitary matrix may be exactly compiled with a smaller length than the universal lower bound (the bound is not **necessary**), there exists a special unitary matrix that cannot (the bound is sufficient and necessary for at least one special unitary matrix). Moreover, the set of special unitary matrices that can be compiled with a smaller length than the universal lower bound has measure zero in $\mathrm{SU}(d)$.

We summarize the proof of this fact, as it motivates our parametrization ideas.

Lemma 2. *[81, Prop. III.1] The set of special unitary matrices $U \in \mathrm{SU}(d)$, $d = 2^n$, that do not need at least $\left\lceil \frac{1}{4}(4^n - 3n - 1) \right\rceil$ CNOTs to be exactly compiled has measure zero in* $\mathrm{SU}(d)$.

Proof. First, since each rotation gate has 1 real parameter while $\mathrm{SU}(d)$ has real dimension $d^2 - 1$, Sard's theorem [95] can be used to show that the set of special unitary matrices that do not need $d^2 - 1$ rotation gates to be exactly compiled has measure zero in $\mathrm{SU}(d)$ [81, Lem. II.2]. Thus, if a product of matrices from the basis set can be reduced to a product with less than $d^2 - 1$ rotation gates, then it is in the measure zero set. In particular, more than 3 consecutive rotation gates on the same qubit in a product can be reduced to only 3 rotation gates. So, without CNOTs, any product can be reduced to one with only $3n < d^2 - 1$ rotation gates. Furthermore, R_z commutes with the control of CNOTs and R_x commutes with the target of CNOTs. Thus, whenever 3 rotation gates are applied after a CNOT on either the control or target qubit, we can rewrite them as 3 rotation gates such that one of the rotation gates commutes with the CNOT. Thus, we can reduce any product of matrices from the basis set to a product with 3 rotation gates applied to each qubit, followed by CNOTs, each followed by only 4 rotation gates. Thus, a product with L CNOTs can be reduced to a product with $4L + 3n$ rotation gates. So, the set of special unitary matrices that do not need at least $\left\lceil \frac{1}{4}(4^n - 3n - 1) \right\rceil$ CNOTs to be exactly compiled has measure zero in $\mathrm{SU}(d)$ [81,

Prop. III.1]. □

Therefore, for an arbitrary n-qubit circuit, one would need at least $\Omega(4^n)$ CNOT gates for exact compilation. Henceforth, we refer to the bound $\lceil \frac{1}{4}(4^n - 3n - 1) \rceil$ as the theoretical lower bound, or TLB for short. Figure 3.2 gives a glimpse of how the reduction in terms of CNOTs and rotations can be carried out for a 3-qubit circuit.

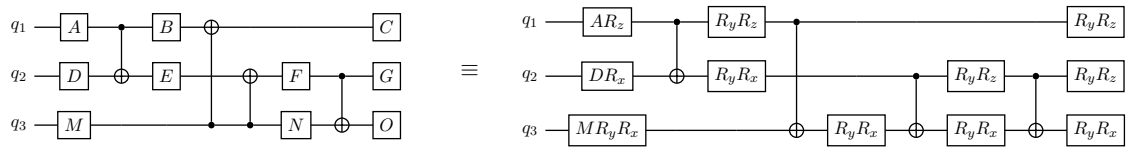
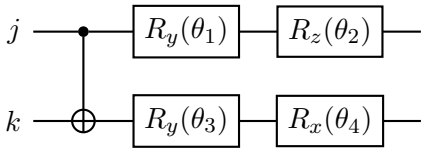


Figure 3.2: An example of an equivalent quantum circuit with CNOT and rotations gates, all the capital letter gates, e.g. A, are single-qubit unitaries that can be compiled via three rotation gates up to an irrelevant global phase. Note that the R_y next to M results from flipping the two upward-facing CNOTs and has angle $-\pi/2$.

While [81] only proved a lower bound, their reasoning motivates a flexible parametric circuit construction. Since every product of matrices from the basis set can be reduced to a product of the form described in the previous paragraph (see also Figure 3.2), we will consider products of that form with different sequences of CNOTs. Let $\mathrm{CU}_{j\to k}(\theta_1, \theta_2, \theta_3, \theta_4)$ be the matrix represented by



.

We call this a "CNOT unit" and we use it as a programmable two-qubit block in the parametric circuit. Define

$$J(L) = \{(j_1 \to k_1, ..., j_L \to k_L) \mid j_m < k_m \in \{1, \ldots, n\} \text{ for all } m \in \{1, \ldots, L\}\},$$

as the set of L-long lists of pairs of control-target indices with the control index smaller than the target index. The latter constraint is without loss of generality since a CNOT can be flipped with y-rotations ($R_y(\pi/2)$ on the left of the control and right of the target, and $R_y(-\pi/2)$ on the right of the control and left of the target). The set $J(L)$ represents all the possible CNOT unit sequences that one can have of length L. In Figure 3.2, we have one element of $J(4)$ as $(1 \to 2, 1 \to 3, 2 \to 3, 2 \to 3)$.

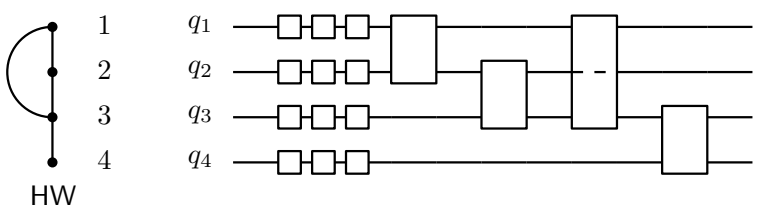


Figure 3.3: Decomposition of a realizable quantum circuit in terms of $V_{\mathsf{ct}}(\boldsymbol{\theta})$. The first three one-qubit gates for all qubits are rotation gates, R_z, R_y, R_z, while the two-qubit blocks represents CNOT units as specified. Here $L = 4$, hence there are 28 rotation angles variables. Moreover, $\mathsf{ct} = (1 \to 2, 2 \to 3, 1 \to 3, 3 \to 4)$, so the hardware connectivity is satisfied.

The cardinality of $J(L)$ can be shown to be $|J(L)| = (n(n-1)/2)^L$. The notation ct is used for an element of $J(L)$, i.e. an L-long list of pairs defining the location of our CNOT units. For example, ct could be $(1 \to 2, 1 \to 3, 2 \to 3, 2 \to 3)$. Given $\mathsf{ct} \in J(L)$, define $\mathsf{ct}(i)$ as the pair at position i and let $V_{\mathsf{ct}} : [0, 2\pi)^{3n+4L} \to \mathrm{SU}(2^n)$ be the following circuit

$$\begin{aligned} V_{\mathsf{ct}}(\boldsymbol{\theta}) =& \mathrm{CU}_{\mathsf{ct}(L)}(\theta_{3n+4L-3}, ..., \theta_{3n+4L}) \cdots \mathrm{CU}_{\mathsf{ct}(1)}(\theta_{3n+1}, ..., \theta_{3n+4}) \\ & [R_z(\theta_1)R_y(\theta_2)R_z(\theta_3)] \otimes \cdots \otimes [R_z(\theta_{3n-2})R_y(\theta_{3n-1})R_z(\theta_{3n})], \end{aligned}$$

where we have collected all the angles $(\theta_1, \ldots, \theta_{3n+4L})$ into the vector $\boldsymbol{\theta}$. The circuit V_{ct} corresponds to the reduced product in the lower bound proof and will be the main object of our study: it will be the circuit blueprint for any circuit realizable in hardware (see also Figure 3.3).

Remark 3. *A useful circuit decomposition is the quantum Shannon decomposition (QSD), proposed in 2006 by Shende, Bullock, and Markov, which uses* $\frac{23}{48}4^n - \frac{3}{2}2^n + \frac{4}{3}$ *CNOTs to decompose any* n*-qubit circuit [82]. This number of CNOTs is only twice the TLB. The resulting circuit is formalized as a sequence of recursive Cartan decompositions in [84], and Theorems 4, 8, and 12 in [82] can be recursively used to (uniquely) find an explicit* ct *corresponding to the QSD. For* $n = 3$, $\mathsf{ct} = (1 \to 2, 1 \to 2, 1 \to 2, 2 \to 3, 1 \to 3, 2 \to 3, 1 \to 2, 1 \to 2, 2 \to 3, 1 \to 3, 2 \to 3, 1 \to 3, 1 \to 2, 1 \to 2, 2 \to 3, 1 \to 3, 2 \to 3, 1 \to 2, 1 \to 2, 1 \to 2)$ *[84, Fig. 3].*

At this point, we can already reformulate the approximate quantum compiling problem (3.1),

in the equivalent form

$$\textbf{(AQCP-CT)} \qquad \min_{L\in[1,\ldots,\bar{L}],\boldsymbol{\theta}\in[0,2\pi)^{3n+4L},\mathsf{ct}\in\mathcal{C}(L)} f_{\mathsf{ct}}(\boldsymbol{\theta}) := \frac{1}{2}\|V_{\mathsf{ct}}(\boldsymbol{\theta}) - U\|_{\mathrm{F}}^2, \tag{3.3}$$

where the set $\mathcal{C}(L)$ represents the set of L-long realizable lists in hardware (for connectivity limitations), and $\bar{L}$ is the length limit.

Problem (3.3) is by no means easier than the original Problem (3.1). However, it is instructive to understand its properties and this will help us devise better solution strategies. For instance, if we were to drop hardware constraints and adopt the QSD strategy of Remark 3, then Problem (3.3) could simplify into:

$$\textbf{(AQCP-QSD)} \qquad \min_{L=\frac{23}{48}4^n-\frac{3}{2}2^n+\frac{4}{3},\boldsymbol{\theta}\in[0,2\pi)^{3n+4L},\mathsf{ct}=\mathrm{QSD}(n)} f_{\mathsf{ct}}(\boldsymbol{\theta}) := \frac{1}{2}\|V_{\mathsf{ct}}(\boldsymbol{\theta}) - U\|_{\mathrm{F}}^2, \tag{3.4}$$

which is a non-convex (but continuous) optimization problem in the $\boldsymbol{\theta}$ variables. In fact, since the structure is fixed, the only free parameters are the rotation angles. But on the other hand, can we do better in terms of number of CNOTs and in terms of incorporating the hardware constraints?

3.5 The optimization landscape

We start by deriving some useful properties of the cost function $f_{\mathsf{ct}}(\boldsymbol{\theta}) := \frac{1}{2}\|V_{\mathsf{ct}}(\boldsymbol{\theta}) - U\|_{\mathrm{F}}^2$ given the target unitary U and a fixed structure. In particular, we will look at V_{ct} and its properties of differentiability, as well as the first and second order derivatives of f_{ct}, which are important for optimization purposes.

First, we would like to prove that $V_{\mathsf{ct}}(\boldsymbol{\theta})$ is a smooth mapping in the rotation angles $\boldsymbol{\theta}$. To do so, we apply the partial derivative operator, $\frac{\partial}{\partial\theta_k}$, to V_{ct}, with respect to an arbitrary angle θ_k, where k is an index in $[1, 3n+4L]$. We further let R_{g_k} be the rotation gate associated to that angle and σ_{g_k} be the respective Pauli operator (g can be x, y, z depending on which type of rotation gate the angle refers to). With this notation, $\frac{d}{d\theta_k}R_{g_k}(\theta_k) = -\frac{i}{2}\sigma_{g_k}R_{g_k}(\theta_k)$, and $\frac{d^2}{d\theta_k^2}R_{g_k}(\theta_k) = -\frac{1}{4}R_{g_k}(\theta_k)$. Thus, if a sequence of partial derivative operators are applied to V_{ct}, to compute its gradient, or Hessian, or higher order derivative, the result is a complex constant times V_{ct} with Pauli gates inserted at specific locations. Thus, we have proved the following theorem.

Theorem 7. *The circuit $V_{\mathsf{ct}}(\boldsymbol{\theta})$ is infinitely differentiable with respect to the rotation angles $\boldsymbol{\theta}$.*

Next, we look at the cost $f_{\mathsf{ct}}(\boldsymbol{\theta})$. Using Eq. (3.2), we get

$$\frac{\partial}{\partial\theta_k}\frac{1}{2}\|V_{\mathsf{ct}}(\boldsymbol{\theta})-U\|_F^2 = -\mathbb{R}\mathsf{e}\ \mathsf{Tr}\left[\frac{\partial}{\partial\theta_k}V_{\mathsf{ct}}(\boldsymbol{\theta})^\dagger U\right]$$

and

$$\frac{\partial^2}{\partial\theta_k\partial\theta_\ell}\frac{1}{2}\|V_{\mathsf{ct}}(\boldsymbol{\theta})-U\|_F^2 = -\mathbb{R}\mathsf{e}\ \mathsf{Tr}\left[\frac{\partial^2}{\partial\theta_k\partial\theta_\ell}V_{\mathsf{ct}}(\boldsymbol{\theta})^\dagger U\right]. \tag{3.5}$$

In particular, by the fact that $\frac{d^2}{d\theta_k^2}R_{g_k}(\theta_k) = -\frac{1}{4}R_{g_k}(\theta_k)$, for the diagonal elements:

$$\frac{\partial^2}{\partial\theta_k^2}\frac{1}{2}\|V_{\mathsf{ct}}(\boldsymbol{\theta})-U\|_F^2 = \frac{1}{4}\mathbb{R}\mathsf{e}\ \mathsf{Tr}[V_{\mathsf{ct}}(\boldsymbol{\theta})^\dagger U] = \frac{d}{4}-\frac{1}{8}\|V_{\mathsf{ct}}(\boldsymbol{\theta})-U\|_F^2. \tag{3.6}$$

Now we are ready to show that $f_{\mathsf{ct}}(\boldsymbol{\theta})$ is strongly smooth in the optimization sense (i.e., its gradient is Lipschitz continuous).

Theorem 8. *For all $\boldsymbol{\theta}$, the Hessian of $f_{\mathsf{ct}}(\cdot) = \frac{1}{2}\|V_{\mathsf{ct}}(\cdot)-U\|_F^2$ evaluated at $\boldsymbol{\theta}$ has spectrum in $[-(3n+4L-3/4)d, (3n+4L-3/4)d]$.*

Proof. First of all, the Hessian is real and symmetric, so its eigenvalues lie on the real line. Then, note that for all unitary matrices, $W \in \mathrm{U}(d)$: $\|W\|_{\mathrm{F}}^2 = d$ and the farthest unitary matrix from any W is $-W$. Thus, $\mathbb{R}\mathsf{e}\ \mathsf{Tr}[\cdot^\dagger\cdot] : \mathrm{U}(d)^2 \to [-d, d]$. Hence, using (3.5), we can show that each off-diagonal element of the $(3n+4L)$ by $(3n+4L)$ Hessian matrix is bounded in absolute value by d. As for the diagonal elements, Eq. (3.6) says that they are bounded below by $-d/4$ and above by $d/4$.

With this in place, we can use Gershgorin's disc theorem [96, Theorem 6.1.1.] with centers in $[-d/4, d/4]$ and radii in $[0, (3n+4L-1)d]$ to prove the claim. □

Theorem 8 says that $f_{\mathsf{ct}}(\boldsymbol{\theta})$ is a strongly smooth function (since the spectrum of the Hessian is uniformly bounded), which implies fast convergence to a stationary point (i.e., points for which the gradient vanishes) for gradient descent. In addition, even though $f_{\mathsf{ct}}(\boldsymbol{\theta})$ is non-convex, gradient descent with random initialization [97], perturbed gradient descent [98], and perturbed Nesterov's method [99] all converge to second-order stationary points under strong smoothness. Second-order

stationary points are stationary points where the Hessian is positive semi-definite, and therefore are local minima. So, applying these methods to $f_{\mathsf{ct}}(\boldsymbol{\theta})$, we will be sure to reach at least a local minimum.

In some cases, such as in non-convex low rank problems, most second-order stationary points are, in fact, global minima [100]. Even though this is not the case for f_{ct}, we do not need to converge to a global minimum necessarily. We would be content finding a $\boldsymbol{\theta}^*$ such that $V_{\mathsf{ct}}(\boldsymbol{\theta}^*)$ is a global minimum up to a global phase transformation.

To this aim, we now present supporting facts that lead us to conjecture that, under certain conditions, we can easily find global minima up to a global phase transformation.

First, we report a technical lemma.

Lemma 3. *Indicate with* $^{\perp_{\mathbb{R}\mathsf{e}\langle\cdot,\cdot\rangle}}$ *the orthogonal complement with respect to the* $\mathbb{R}\mathsf{e}\mathsf{Tr}[(\cdot)^\dagger(\cdot)]$ *operation. The orthogonal complement* $\mathfrak{u}(d)^{\perp_{\mathbb{R}\mathsf{e}\langle\cdot,\cdot\rangle}}$ *is the set of Hermitian matrices and* $\mathfrak{su}(d)^{\perp_{\mathbb{R}\mathsf{e}\langle\cdot,\cdot\rangle}} = \mathfrak{u}(d)^{\perp_{\mathbb{R}\mathsf{e}\langle\cdot,\cdot\rangle}} + \mathrm{span}_{\mathbb{R}}\{iI\}$.

Proof. Recall that $\mathfrak{u}(d)$ consists of anti-Hermitian matrices and $\mathfrak{su}(d)$ consists of traceless anti-Hermitian matrices. The orthogonal complement $\mathfrak{u}(d)^{\perp_{\mathbb{R}\mathsf{e}\langle\cdot,\cdot\rangle}}$ is the set of all the matrices H for which $\mathbb{R}\mathsf{e}\mathsf{Tr}[H^\dagger U] = 0$, for all U anti-Hermitian. By direct calculation, indicating with h_{ij} the element i,j of matrix H, and with $u_{i,j}$ the one of matrix U, as well as $\bar{h}_{i,j}$ the conjugate of $h_{i,j}$, then,

$$\mathbb{R}\mathsf{e}\mathsf{Tr}[H^\dagger U] = \mathbb{R}\mathsf{e}\left[\sum_i \sum_j \bar{h}_{i,j} u_{i,j}\right].$$

If the above has to be 0 for all anti-Hermitian matrices U, then H has to be Hermitian by $u_{j,i} = -\bar{u}_{i,j}$ (in particular, the diagonal of U is only imaginary). On the other hand, if H is Hermitian (therefore its diagonal is only real), then,

$$\mathbb{R}\mathsf{e}\mathsf{Tr}[H^\dagger U] = \underbrace{\mathbb{R}\mathsf{e}\Big[\sum_{j>i} \bar{h}_{i,j} u_{i,j} + \sum_{j<i} h_{i,j}\bar{u}_{i,j}\Big]}_{=0} + \underbrace{\mathbb{R}\mathsf{e}\Big[\sum_{i=j} \bar{h}_{i,j} u_{i,j}\Big]}_{=0} = 0$$

Hence, the orthogonal complement $\mathfrak{u}(d)^{\perp_{\mathbb{R}\mathsf{e}\langle\cdot,\cdot\rangle}}$ is the set of Hermitian matrices.

In addition, $\mathfrak{su}(d)$ consists of traceless anti-Hermitian matrices, so its orthogonal complement consists not only of Hermitian matrices, but also of any matrix of the form $H + ciI$, where H is Hermitian and $c \in \mathbb{R}$. Thus, the thesis follows. □

With the Lemma in place, we are ready for another intermediate result that pertains to stationary points.

Theorem 9. *Let $U \in \mathrm{SU}(d)$ be a special unitary matrix, and $V(\cdot) : \mathbb{R}^p \to \mathrm{SU}(d)$ be a function that is infinitely differentiable, surjective, and of constant rank. For all $U \in \mathrm{SU}(d)$, a parameter value $\boldsymbol{\theta} \in [0, 2\pi)^p$ is a stationary point of the cost $f(\cdot) = \frac{1}{2}\|V(\cdot) - U\|_{\mathrm{F}}^2$ if and only if $V(\boldsymbol{\theta})^\dagger U$ has eigenvalues in $\{e^{i\alpha}, -e^{-i\alpha}\}$ for some $\alpha \in [0, 2\pi)$.*

Proof. Assume the hypotheses. We have the following line of implications:

$$
\begin{aligned}
\boldsymbol{\theta} \text{ is a stationary point of } f \quad &\overset{1}{\iff} \mathbb{R}\mathrm{e}\ \mathsf{Tr}\left[\frac{\partial}{\partial \theta_k} V(\boldsymbol{\theta})^\dagger U\right] = 0 \text{ for all } k \in [p] \\
&\overset{2}{\iff} U \in \mathrm{span}_{\mathbb{R}}\left\{\frac{\partial}{\partial \theta_k} V(\boldsymbol{\theta})\right\}^{\perp_{\mathbb{R}\mathrm{e}\langle\cdot,\cdot\rangle}} \\
&\overset{3}{=} \left[\mathcal{T}_{V(\boldsymbol{\theta})}\mathrm{SU}(d)\right]^{\perp_{\mathbb{R}\mathrm{e}\langle\cdot,\cdot\rangle}} \\
&\overset{4}{=} \left[V(\boldsymbol{\theta})\mathfrak{su}(d)\right]^{\perp_{\mathbb{R}\mathrm{e}\langle\cdot,\cdot\rangle}} \\
&\overset{5}{=} V(\boldsymbol{\theta})\mathfrak{su}(d)^{\perp_{\mathbb{R}\mathrm{e}\langle\cdot,\cdot\rangle}} \\
&\iff V(\boldsymbol{\theta})^\dagger U \in \mathfrak{su}(d)^{\perp_{\mathbb{R}\mathrm{e}\langle\cdot,\cdot\rangle}}.
\end{aligned}
$$

Step 1 follows from the definition of a stationary point (a point where the gradient vanishes) and our derivation of the gradient for $f(\cdot) = \frac{1}{2}\|V(\cdot) - U\|_{\mathrm{F}}^2$. Step 2 follows from the definition of an orthogonal complement and the linearity of the trace (i.e., U is in the span of the orthogonal complement of the derivative of V if and only if the gradient is zero). Step 3 follows from the global rank theorem [95, Thm 4.14] under infinite differentiability, surjectivity, and constant rank assumptions, where we use $\mathcal{T}$ to denote the tangent space (which appears since we are taking the derivative of $V(\boldsymbol{\theta})$). Concretely, given $W \in \mathrm{SU}(d)$, $\mathcal{T}_W\mathrm{SU}(d) \coloneqq \{\gamma'(0) \mid \gamma : \mathbb{R} \to \mathrm{SU}(d) \text{ smooth with } \gamma(0) = W\}$. For Step 4, a little more care has to be put. First, it can be shown,

via the left or right group action, that $\mathcal{T}_W \mathrm{SU}(d)$ is isomorphic to $\mathcal{T}_I \mathrm{SU}(d) = \mathfrak{su}(d)$. We want now to show that $\mathcal{T}_W \mathrm{SU}(d) = W\mathfrak{su}(d)$. Since they are isomorphic, we only have to show one direction. Let $A \in \mathfrak{su}(d)$. We want to show $WA \in \mathcal{T}_W \mathrm{SU}(d)$. Towards this end, define $\gamma(t) = W \exp(tA)$. Note $\gamma(0) = W$, and $\gamma'(0) = WA$. Furthermore, $\gamma(t) \in \mathrm{SU}(d)$ (which is easy to see by direct computations[1]). Thus, $WA = \gamma'(0) \in \mathcal{T}_W \mathrm{SU}(d)$ and so $\mathcal{T}_W \mathrm{SU}(d) = W\mathfrak{su}(d)$, proving Step 4. Step 5 follows via properties of the Hermitian transpose. Explicitly, for a set $\mathcal{U}$,

$$\begin{aligned}
[W\mathcal{U}]^{\perp_{\mathbb{R}\mathrm{e}\langle\cdot,\cdot\rangle}} &= \{Y \mid \mathbb{R}\mathrm{e}\ \mathsf{Tr}[Y^\dagger WA] = 0 \text{ for all } A \in \mathcal{U}\} \\
&= \{Y \mid \mathbb{R}\mathrm{e}\ \mathsf{Tr}[(W^\dagger Y)^\dagger A] = 0 \text{ for all } A \in \mathcal{U}\} \\
&= W\{W^\dagger Y \mid \mathbb{R}\mathrm{e}\ \mathsf{Tr}[(W^\dagger Y)^\dagger A] = 0 \text{ for all } A \in \mathcal{U}\} \\
&= W\mathcal{U}^{\perp_{\mathbb{R}\mathrm{e}\langle\cdot,\cdot\rangle}}.
\end{aligned}$$

Now all that is left is to determine the contents of $\mathfrak{su}(d)^{\perp_{\mathbb{R}\mathrm{e}\langle\cdot,\cdot\rangle}} \cap \mathrm{U}(d)$. We have from Lemma 3 that $\mathfrak{su}(d)^{\perp_{\mathbb{R}\mathrm{e}\langle\cdot,\cdot\rangle}}$ is the set of matrices that equal $A + ciI$ for some Hermitian A and some $c \in \mathbb{R}$. Furthermore, Hermitian matrices are precisely those that equal $Q^\dagger DQ$ for some unitary Q and diagonal D with real entries. So, $\mathfrak{su}(d)^{\perp_{\mathbb{R}\mathrm{e}\langle\cdot,\cdot\rangle}}$ is the set of matrices that are unitarily diagonalizable with eigenvalues in $\{\lambda + ci \mid \lambda \in \mathbb{R}\}$ for some $c \in \mathbb{R}$. But, unitary matrices are precisely the matrices that are unitarily diagonalizable with eigenvalues in $\mathrm{U}(1)$. Thus, $\mathfrak{su}(d)^{\perp_{\mathbb{R}\mathrm{e}\langle\cdot,\cdot\rangle}} \cap \mathrm{U}(d)$ is the set of matrices that are unitarily diagonalizable with eigenvalues in $\{e^{i\alpha}, -e^{-i\alpha}\}$ for some α, since $\{\lambda + ci \mid \lambda \in \mathbb{R}\} \cap \mathrm{U}(1) = \{e^{i\alpha}, -e^{-i\alpha}\}$ for $\alpha = \arcsin(c)$. □

Theorem 9 describes the space of stationary points for infinitely differentiable, surjective, constant rank mappings. Note that the image of points that do not have constant rank has measure zero in $\mathrm{Im}(V) = \mathrm{SU}(d)$ by Sard's theorem [95, Ch. 6], so this assumption is not restrictive. Moreover, we have from Theorem 7 that V_{ct} is infinitely differentiable, so we already have a relevant mapping that is infinitely differentiable. Finally, the surjectivity assumption appears as though it could be relaxed. But, unfortunately, this is not the case.

[1] We have $\gamma(t)^\dagger \gamma(t) = \exp(tA)^\dagger W^\dagger W \exp(tA) = \exp(tA)^\dagger \exp(tA) = \exp(tA^\dagger)\exp(tA) = \exp(-tA)\exp(tA) = \exp(-tA + tA) = I$ and $\det(\gamma(t)) = \det(W\exp(tA) = \det(W)\det(\exp(tA)) = \det(\exp(tA)) = \exp(t\mathsf{Tr}(A)) = \exp(0) = 1$.

Remark 4. *One might wonder if the surjectivity assumption can be relaxed to the assumption* $U \in \mathrm{Im}(V)$*. The answer is no. We found counter-examples when we were running the numerical experiments. In particular, for certain structures* ct *with length smaller than the TLB, hence lacking surjectivity, and with the Toffoli gate as the target unitary* U*, we computed some stationary points of* f_{ct} *that exactly compiled* U *and some that failed the eigenvalue condition of Theorem 9. In other words, for these examples,* $U \in \mathrm{Im}(V_{\mathsf{ct}})$ *but the conclusion of Theorem 3 does not hold. These results are shown in Table 3.2: the stationary points in the unsuccessful compilations did not satisfy the eigenvalue condition of Theorem 9.*

Since V_{ct} is infinitely differentiable, if only it is surjective as well then its stationary points are precisely the points such that $V_{\mathsf{ct}}(\boldsymbol{\theta})^\dagger U$ has eigenvalues in $\{e^{i\alpha}, -e^{-i\alpha}\}$ for some $\alpha \in [0, 2\pi)$. Having more than one eigenvalue means that the stationary points are not in general global minima up to a global phase transformation though. But, we conjecture that the form of stationary points in Theorem 9 simplifies even further for second-order stationary points of V_{ct}.

Conjecture 1. *If the parametric circuit* $V_{\mathsf{ct}}(\cdot)$ *is surjective and the Hessian of the cost function* $f_{\mathsf{ct}}(\cdot) = \frac{1}{2}\|V_{\mathsf{ct}}(\cdot) - U\|_{\mathrm{F}}^2$ *is positive semi-definite at a stationary point* $\boldsymbol{\theta}$*, then* $V_{\mathsf{ct}}(\boldsymbol{\theta})^\dagger U = e^{i\alpha} I$ *for some* $\alpha \in [0, 2\pi)$*, meaning that any stationary point* $\boldsymbol{\theta}$ *is a global minimum of the cost function* $f_{\mathsf{ct}}(\cdot)$*, and therefore a solution to the quantum compiling problem, up to a global phase.*

We tested Conjecture 1 extensively. In order to falsify the conjecture, we would need an example of V_{ct}, $\boldsymbol{\theta}$, and U such that V_{ct} is surjective, the Hessian is positive semi-definite, and $V_{\mathsf{ct}}(\boldsymbol{\theta})^\dagger U$ is not a scalar matrix. We searched for counter-examples in two experiments. In both, we used structures with length smaller than the TLB as the non-surjective mappings, and the QSD decomposition structure as the surjective mapping. In the first experiment, we computed $\boldsymbol{\theta}$ via gradient descent. Hence, there was no need to explicitly compute the Hessian. In the second experiment, we randomly generated $W \in \mathfrak{su}(d)^{\perp_{\mathbf{Re}\langle\cdot,\cdot\rangle}} \cap \mathrm{SU}(d)$ and $\boldsymbol{\theta} \in \mathbb{R}^p$, then set $U = V_{\mathsf{ct}}(\boldsymbol{\theta})W$. From the proof of Theorem 9, we have that $\mathfrak{su}(d)^{\perp_{\mathbf{Re}\langle\cdot,\cdot\rangle}} \cap \mathrm{SU}(d)$ is the set of matrices that are unitarily diagonalizable with eigenvalues in $\{e^{-\alpha}, -e^{-i\alpha}\}$ for some α such that the multiplicity of

$e^{i\alpha}$ times α plus the multiplicity of $-e^{-i\alpha}$ times $\pi - \alpha$ is an integer multiple of 2π. So, in order to generate W, we sampled $Q \in U(d)$ randomly and chose a non-scalar eigenvalue matrix Λ, then set $W = Q^\dagger \Lambda Q$. Thus, $V_{\mathsf{ct}}(\boldsymbol{\theta})^\dagger U$ was a non-scalar matrix by construction. We did not find any counter-examples in either experiment. Note that we did have to explicitly compute the Hessian and its eigendecomposition for the second experiment, but this is the only place in the chapter where we actually compute the Hessian.

Conjecture 1 says that, despite the difficulty in finding global minima due to non-convexity, the stationary points we find are global minima up to a global phase. However, this is only in the case of surjectivity, and so the non-convexity causes greater difficulty for lengths shorter than the minimum required for surjectivity. We discuss this issue further in the next section. In particular, the divide between easy compilation problems and harder ones is the surjectivity of $V_{\mathsf{ct}}(\boldsymbol{\theta})$.

3.6 Special structures

While the results and discussion in the previous section were for arbitrary CNOT unit structures, we would like to focus on particular structures henceforth. The properties that we desire such structures to have are:

(1) they have to be able to capture all circuits of a given qubit size (i.e., surjectivity);

(2) their length has to be between one and two times the lower bound for surjectivity;

(3) they have to be compressible, that is, they have to include a method for finding new structures with shorter length;

(4) their maximum approximation error has to depend favorably on the compressed length;

(5) they should exactly compile special quantum gates (e.g., Toffoli gates) with length close to optimal;

(6) incorporating hardware constraints should only increase the length of the compressed structure by a constant multiplicative factor in order to preserve the same approximation

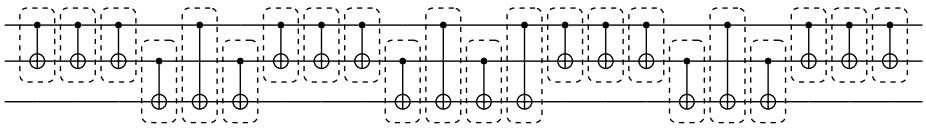
error.

Many of these properties are intertwined. For example, property (1) is necessary for property (4) and also for convergence (to a stationary point that only requires a global phase transformation), as discussed in the previous section. On the other hand, since the compressed structure will not be surjective when its length is less than the surjectivity bound, convergence becomes more difficult. Assuming property (2), then properties (1) and (5) are the same for all unitary matrices except a measure zero set. However, there are important matrices in the measure zero set, such as qubit permutations, controlled operations, and cyclic shifts [101], so we keep these requirements separate. But, as mentioned in Remark 4, if we want to exactly compile these with minimal length, the mapping may be far from surjective and so convergence will be much more difficult. We will see this when compiling for Toffoli gates.

We will consider three structures in this chapter: cart, standing for Cartan; sequ, standing for sequential; and spin, standing for spin.

Definition 1. *Let* $\mathsf{cart} \in J(\frac{23}{48}4^n - \frac{3}{2}2^n + \frac{4}{3})$ *correspond to the QSD decomposition.*

As an example, the structure of CNOT units for cart(3) is



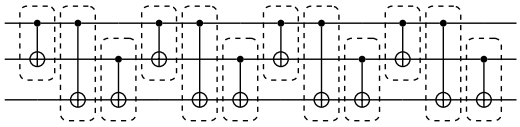
where the CNOT boxes represent our CNOT unit [84]. cart satisfies properties (1) and (2) by the constructive proof of [82]. On the other hand, as is, it does not satisfy property (3) on compressibility. So, we will propose a method in Section 3.8 that can be used to compress it to as short as the TLB with practically no error, thereby also supporting property (4). cart only increases by a multiplicative factor of 9 for the uncompressed structure, as shown in [82]. However, it is an open problem how to compile special gates close to their optimal length, and compress it in a way that does not increase connectivity, so the answer to properties (5) and (6) is unknown.

While cart has a recursive structure, the next two layouts we consider have repeating structures.

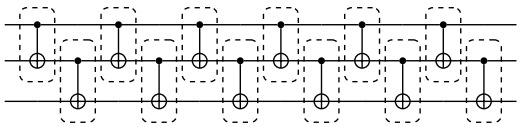
Definition 2. *Let* $\mathsf{sequ}(L) \in J(L)$ *correspond to* L *CNOT units in order. There are* $\frac{n(n-1)}{2}$ *CNOT units and the order of* $CNOT_{jk}$ *is* $n(j-1) + \left(k - \frac{j(j+1)}{2}\right)$*. Once the end of the order is reached, it repeats.*

Definition 3. *Let* $\mathsf{spin}(L) \in J(L)$ *correspond to the structure that alternates between* $(1 \to 2, 3 \to 4, \ldots)$ *and* $(2 \to 3, 4 \to 5, \ldots)$.

As an example, the structure of CNOTs for $\mathsf{sequ}(12)$ when $n = 3$ is



and for $\mathsf{spin}(12)$ when $n = 3$ is



.

Both $\mathsf{sequ}(L)$ and $\mathsf{spin}(L)$ have the length of the circuit as an input, and so satisfy property (3). But for what L, if any, do they satisfy property (1) on surjectivity? We run experiments in Section 3.7 to evaluate $\mathsf{sequ}(L)$ and $\mathsf{spin}(L)$ with respect to properties (1), (2), and (4). The results support the following conjecture about (1) and (2).

Conjecture 2. $V_{\mathsf{sequ}(L)}$ *and* $V_{\mathsf{spin}(L)}$ *are surjective for* $L \geq \frac{1}{4}(4^n - 3n - 1)$*, that is, the TLB.*

In Section 3.7, we also apply $\mathsf{sequ}(L)$ and $\mathsf{spin}(L)$ to specific quantum gates (e.g., Toffoli), thereby supporting property (5), despite non-surjectivity.

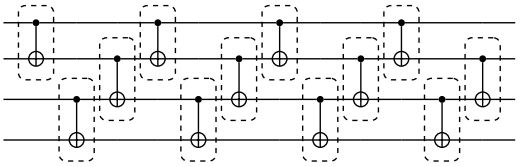
Structure $\mathsf{sequ}(L)$ can incorporate hardware constraints by simply skipping the CNOTs for unconnected qubits, but it is not obvious how much L would have to increase to preserve a given maximum approximation error.

Fortunately, the results in the next section suggest L would not have to increase at all. In the next section, we test $\mathsf{sequ}(L)$ with hardware constraints corresponding to both a star topology and a line topology. All three structures perform comparably to $\mathsf{sequ}(L)$ without any hardware constraints.

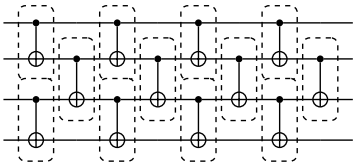
Furthermore, $\mathsf{spin}(L)$, another structure that satisfies the line topology hardware constraints, also performs comparably. The experiments suggest that different repeating structures, as long as they include all of the qubits, may fill the space equally fast, on average.

In Table 3.1, we summarize our main findings for the particular structures we have discussed.

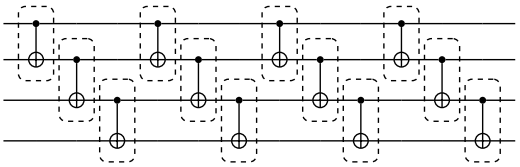
Remark 5. *Quantum computers can implement arbitrary gates on disjoint qubits simultaneously. Hence, they can implement CNOT units on disjoint qubits simultaneously. Thus, while our theory is in terms of length–the number of CNOTs in a structure, two other relevant metrics are CNOT depth—the minimum number of layers of CNOT units in a structure—and circuit depth—the minimum number of layers in a structure. Note that if all rotation angles are non-zero, then circuit depth is three plus three times CNOT depth. Also, CNOT depth is always smaller than or equal to the length. In particular,* $\mathsf{spin}(L)$ *can be implemented with CNOT depth* $\lceil 2L/(n-1) \rceil$ *for* $n > 3$*. As another example,* $\mathsf{spin}(12)$ *when* $n = 4$ *is*



which becomes, if we implement CNOT units on disjoint pairs of qubits simultaneously,



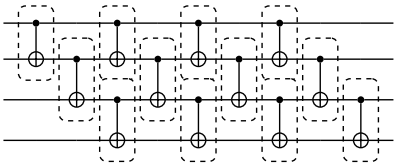
On the other hand, imposing the line topology hardware constraints on $\mathsf{sequ}(12)$ *results in*



which becomes, if we implement CNOT units on disjoint pairs of qubits simultaneously,

Table 3.1: Considered structures and their properties. Numbered references are to subsections.

Structure	Properties					
	(1)	(2)	(3)	(4)	(5)	(6)
cart	[84]	[84]	3.8.3	3.8.3	?	?
$\mathsf{sequ}(L)$	Conjecture 2		✓	3.7.2	3.7.3	3.7.2
$\mathsf{spin}(L)$	Conjecture 2		✓	3.7.2	3.7.3	✓



Structurally, $\mathsf{spin}(12)$ *and* $\mathsf{sequ}(12)$ *are almost identical on the line topology, except that* $\mathsf{spin}(12)$ *moves the last CNOT unit of* $\mathsf{spin}(12)$ *to the first layer and so decreases the CNOT depth from 9 to 8.*

3.7 Gradient descent

In Section 3.3, we formulated the mathematical optimization problem, and in Section 3.5 we discussed some of its properties once specified to CNOT unit structures. In the previous section, we discussed some special layouts. We are now ready to look at the approximate compiling problem for a fixed structure. Specifically, we look at the problem

$$\textbf{(AQCP-}\boldsymbol{\theta}\textbf{)} \qquad \min_{\boldsymbol{\theta}\in[0,2\pi)^p,\mathsf{ct}=\overline{\mathsf{ct}}} f_{\mathsf{ct}}(\boldsymbol{\theta}) := \frac{1}{2}\|V_{\mathsf{ct}}(\boldsymbol{\theta}) - U\|_{\mathrm{F}}^2, \tag{3.7}$$

where ct is fixed to one of the mentioned structures ($\mathsf{cart}, \mathsf{sequ}(L), \mathsf{spin}(L)$), which in turn specifies how many rotation angles there are (i.e., p), and we will use a first-order method to find second-order stationary points. The specific method can be tuned in practice, so we present gradient descent and then discuss the modifications that can be made for its variants.

First, we randomly initialize $\boldsymbol{\theta}[0]$ from the uniform distribution on $[0, 2\pi)^p$. Then, we compute

$$\boldsymbol{\theta}[t+1] = \boldsymbol{\theta}[t] - \alpha\nabla f_{\mathsf{ct}}(\boldsymbol{\theta}[t]) \tag{3.8}$$

until the stopping criteria, $\|\nabla f_{\mathsf{ct}}(\boldsymbol{\theta}[t])\| \leq \epsilon$, is met. The step-size, α, is tuned in practice, and the final $\boldsymbol{\theta}$ is then wrapped in the $[0, 2\pi)^p$ set. Component-wise, Eq. (3.8) reads,

$$\theta_k[t+1] = \theta_k[t] + \alpha \mathbb{R}\mathsf{e}\ \mathsf{Tr}\left[\frac{\partial}{\partial \theta_k} V_{\mathsf{ct}}(\boldsymbol{\theta})^\dagger U\right]. \tag{3.9}$$

One variant of gradient descent is Nesterov's method [48], which applies the gradient descent step to an auxiliary sequence. The auxiliary sequence is essentially the main sequence but with so-called "momentum." For smooth convex optimization, Nesterov's method is optimal among first-order methods.

Another modification that can be made to gradient descent is the injection of noise. On the one hand, diminishing Gaussian noise can be added to the gradient, in which case the method is called gradient descent with Langevin dynamics. On the other hand, noise uniformly sampled from a fixed radius ball can be added in a random direction whenever a "stuck" criteria is met, in which case the method is called perturbed gradient descent.

We found that it was not necessary to add noise and that Nesterov's method was much faster than gradient descent, so we used it as our method of choice. As mentioned previously, random initialization helps escape saddle points to find second-order stationary points. However, the random prior may also affect what kind of second-order stationary points are found. In the non-surjective setting, we would like to avoid spurious local minima as well as saddle points, so it is a future research direction to consider different initializations.

3.7.1 A note on computational complexity

Note that the gradient computation makes up the majority of the computation and memory complexity. Specifically, we have to compute $\frac{\partial}{\partial \theta_k} V_{\mathsf{ct}}(\boldsymbol{\theta})$ for each $k \in \{1, \ldots, p\}$, each of which takes the same number of flops to compute as $V_{\mathsf{ct}}(\boldsymbol{\theta})$. To compute the matrix for a single layer, via the kronecker product, takes $O(d^2)$ flops. To multiply two matrices takes $O(d^3)$ flops. There are L layers so the total computational complexity for $V_{\mathsf{ct}}(\boldsymbol{\theta})$ is $O(Ld^3)$. Thus, computing $\frac{\partial}{\partial \theta_k} V_{\mathsf{ct}}(\boldsymbol{\theta})$ from scratch for each $k \in \{1, \ldots, p\}$ comes out to $O(L^2 d^3)$. The memory complexity is $O(L + d^2)$. Since

this is the dominant part of the algorithm, the total computational complexity is $O(L^2 d^3 T)$ (where T is the number of iterations) and the memory complexity is $O(L + d^2)$.

Fortunately, it is possible to trade-off between the computational and memory complexity. We do this in a similar way to backpropogation, the algorithm for computing the gradient of the loss of a feedforward neural network applied to a sample point [102], which is an example of reverse mode automatic differentiation [103]. Instead of computing $\frac{\partial}{\partial \theta_k} V_{\mathsf{ct}}(\boldsymbol{\theta})$ from scratch for each $k \in \{1, \ldots, p\}$, we can store each of the intermediate matrix computations. Specifically, we can compute the matrix for each layer, taking $O(Ld^2)$ flops and storage. Then we can compute the matrix multiplications from both the right and left, storing the intermediate outputs. That takes $O(Ld^3)$ flops and $O(Ld^2)$ storage. Then, we compute four new versions of each layer corresponding to the partial derivative of each of the four parameters in a layer. This takes $O(Ld^2)$ flops and storage again. Finally, for the partial derivative of each parameter, we make only two matrix computations, taking $O(Ld^3)$ flops and $O(d^2)$ storage (since we don't store all p matrices at once, we multiply them by U and compute the real part of the trace to get a real number). Thus, this way of computing the gradient descent iterates amounts to $O(Ld^3 T)$ flops and $O(Ld^2)$ storage. The reduction in computational complexity from L^2 to L is significant since L can be exponential in the number of qubits. In particular, when the goal is exactly compiling general unitary matrices, we need $L = \Omega(4^n) = \Omega(d^2)$. On the other hand, if the goal is exactly compiling specific unitary matrices or approximately compiling general unitary matrices, L may be smaller. For example, to exactly compile the Toffoli gate, we only need $L = \Omega(n) = \Omega(\log_2(d))$.

These analyses of the computation and memory complexity of gradient descent are for its implementation on a classical computer. However, it can be implemented on a quantum computer as well. Algorithm 3 of [2] explains how to implement it using the power-of-two-qubits circuit. While this is a possibility, we do not follow it here.

3.7.2 Numerical tests: random unitary matrices, towards G1

We start by testing the structures and gradient descent on random unitary matrices, which is our goal [G1]. We consider three different hardware connectivity graphs with edge sets

$$E_1 = \{(j,k) \mid j,k \in \{1,\ldots,n\}, j \neq k\} \qquad \text{(Full connectivity)}$$

$$E_2 = \{(1,j+1) \mid j \in \{1,\ldots,n-1\}\} \qquad \text{(Star connectivity)}$$

$$E_3 = \{(j,j+1) \mid j \in \{1,\ldots,n-1\}\} \qquad \text{(Line connectivity).}$$

E_1 corresponds to no hardware constraints, E_2 corresponds to the star topology, and E_3 corresponds to the line topology. Corresponding to these, we consider the structures sequ_{E_1}, sequ_{E_2}, sequ_{E_3}, and spin. Note that spin corresponds to E_3, but $\mathsf{sequ}_{E_3} \neq \mathsf{spin}$ except when $n \leq 3$.

Full connectivity results. We ran the experiments for $n = 3$ and $n = 5$ qubits. In both cases, we sampled 100 random unitary matrices for each structure for different values of L. We ran gradient descent for each sample and then computed the approximation error and its probabilities. These curves are given in Figure 3.4 and Figure 3.5. As one can see, both sequ and spin are almost identical on how the error depends on the length L, thereby suggesting that the actual structure does not make a big difference. In addition, both structures reach zero error around their theoretical lower bound (which can be computed as $L = 14$ for $n = 3$, and $L = 252$ for $n = 5$), supporting our Conjecture 2. Figure 3.5 gives probabilistic statements: the x-axis represents a desired maximum approximation error (bound), while the y-axis reports the probability that starting from a random guess for a random unitary, we obtain an error larger than the desired bound. Note that this corresponds to one minus the CDF of the approximation error. Each L considered has a curve in Figure 3.5, representing the distribution of the approximation error, while in Figure 3.4 it is compressed to a single point by taking either the mean or the maximum.

Limited connectivity results.

With the same settings, we investigate the effect of different connectivity patters on sequ, for $n = 5$ qubits. For $n = 5$, the theoretical lower bound is 252 CNOTs. Figure 3.6 shows that sequ on all three connectivity patterns approach zero approximation error around its theoretical lower

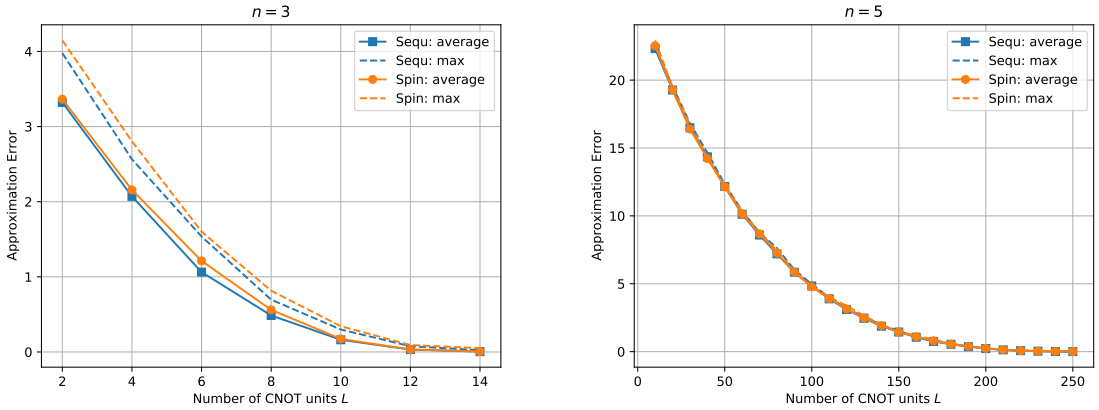


Figure 3.4: Approximation error, f_{ct}, for $n = 3$ (left) and $n = 5$ (right).

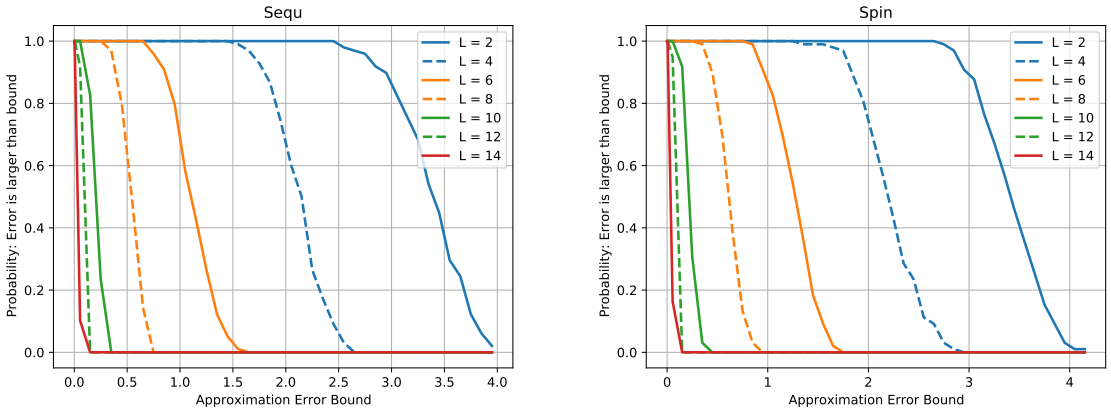


Figure 3.5: Error probabilities for sequ and spin on 3-qubits for different values of L that starting from a random initial point for a random unitary we obtain an approximation error larger than a desired bound.

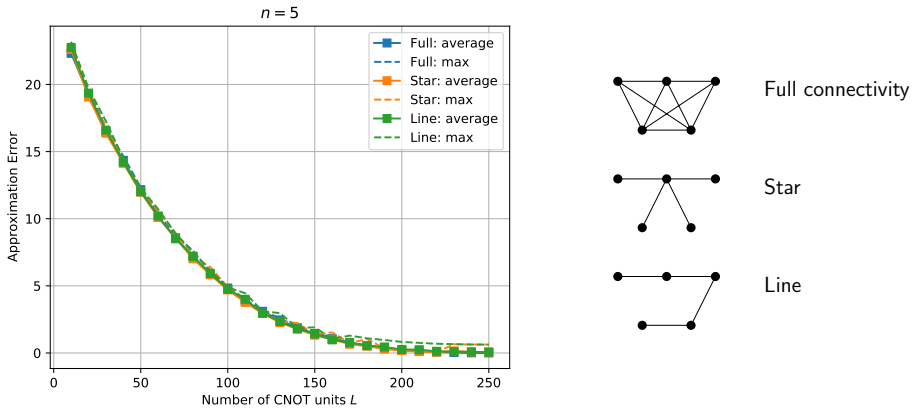


Figure 3.6: Approximation error $n = 5$ with different connectivity patterns.

bound (in the worst case, the max lines arrive at about 0.7 error, which corresponds to a fidelity of 96%). Furthermore, the dependence of the approximation error on the length is a convex curve that is basically identical for all three patterns.

From an engineering perspective, it is quite surprising that the structures with limited hardware connectivity perform as well as sequ, corresponding to full connectivity. However, from the "separating parameters" perspective that was used to derive the lower bound, this is less surprising: sequ seem to "fill out" the quantum circuit and separate the parameters equally as well on a line as on a fully connected graph.

This is in sharp contrast with the QSD which uses 20 and 444 CNOT units for $n = 3$ and $n = 5$, respectively, and increases in length by a multiplicative factor of 9 for the line topology.

These experiments suggest that both spin and sequ are surjective for L equal to the lower bound and that their approximation error depends favorably on the length, supporting properties (1) through (4). Regarding property (6), the experiments suggest that the length does not have to increase at all (or perhaps very slightly, in the worst case scenario).

Remark 6. *It is important to note here that the fact that the connectivity does not matter in a random unitary setting is a general statement about the overall smooth optimization landscape. We are saying that, if you take a random unitary, and a few random initial points, on average, the*

approximation error you obtain depends only on the number of CNOT units. This statement says that even if the different structures have different local minima and properties, with a macro-scale lens (i.e., in the worst-case), they behave very similarly. On the other hand, when one is interested in compiling a particular gate (say a Toffoli), in the least number of CNOT units, the structure plays an important role as discussed in the literature, e.g., [104, 105], and as we will see next. In this case, the fact that connectivity matters is a statement about best case scenarios.

3.7.3 Numerical tests: special gates, towards G2

The only property we are left to explore in this section is property (5), meaning how close to optimal are the various structures in the case of special (often used and well-studied) gates. So, in addition to compiling random unitary matrices as efficiently as possible, we want to recover the shorter lengths that some important gates allow. For example, k-controlled $(n-k)$-qubit operations can be compiled with $O(n)$ gates if $k < n-1$ and $O(n^2)$ gates if $k = n-1$ [101, Ex. 4.2][79, Lems. 7.2 and 7.5]. In particular, the n-qubit Toffoli gate, also known as the multi-controlled-X gate, can be thought of as a k-controlled $(n-k)$-Toffoli gate, and so can be compiled with $O(n)$ gates. Furthermore, [106] gives a lower bound of $2n$ CNOTs for the n-qubit Toffoli gate (other lower bounds are possible when considering circuits with ancillae [107]). The lower bound is tight for $n = 3$. Qiskit [93] decomposes the 3-qubit Toffoli gate into 6 CNOTs and the 4-qubit Toffoli gate into 14 CNOTs[2] . We remark that these numbers of CNOTs are significantly lower than the TLB, and therefore property (5) is not trivial to fulfill.

In addition to the Toffoli gate, we also consider the Fredkin 3 qubit gate, as well as the 1-bit full adder (which is a 4 qubit gate). The list of important gates is by no means exhaustive, but it already gives a glimpse of how well sequ and spin work for important gates.

To test property (5), we ran our gradient descent algorithm on spin and sequ many times (we report in Table 3.2 the results, as well as the number of exact compilations divided by the number

[2] A 14 CNOT implementation can be obtained from the 20 CNOT one of [79] by substituting the controlled-Vs with their 2-CNOT implementations and applying the templates presented in [60].

of tries). We compare with the Qiskit compilations on full connectivity and on line connectivity (with the usual workflow, one would compile e.g., a Toffoli gate on the full connectivity hardware and then add swap gates to transpile it on the limited connectivity one) [93].

In the case of sequ, we also consider the case of permuting the CNOT units to span more possibilities: this is, at the moment, a random search permuting all the possibilities, which is unpractical for large L's (for instance for $L = 14$ it would amount to over 150 million possibilities), but gives us a glimpse that structure does matter when compiling very specific gates in the non-surjective domain.

The results indicate that, e.g., sequ(18) and spin(18) can exactly compile a 4-qubit Toffoli, which is better than the 20 CNOT implementation of [79]. And allowing for permutations in the sequence of CNOTs, even sequ(14) can do it, which is optimal[3] . The solution we have found is reported in Figure 3.7, and it uses a different sequence of CNOT gates than the one in Qiskit.

What we observe is that there are different possibilities in compiling a certain circuit exactly, and in most cases the optimal solutions are different. This shows that our algorithm can be used as a tool to discover new exact compilations of special gates.

The experiment suggests that property (5) is satisfied to a reasonable extent for both sequ and spin.

Finally, by looking at the performance of Qiskit when compiling on a line connectivity and comparing it to our spin structure, which enforces the line connectivity by design, we further appreciate the advantage of our method in terms of CNOT count.

3.8 Circuit compression via regularization

We move now to address property (3) on compressibility in more detail. In particular, motivated by the cart structure, we notice that, while cart is provably surjective with length twice the lower bound, thus satisfying properties (1) and (2), it is not clear how to compress it and

[3] The ease at which we have found this, despite the 14!/3!/3!/2!/2!/2!/2! possible structures, indicates that there may be more than one structure that can deliver an optimal compilation. Note that, in general, for m types of CNOTs with corresponding quantities $L_1, \ldots, L_m$, there are $(L_1 + \cdots + L_m)!/(L_1! \cdots L_m!)$ permutations.

Table 3.2: Exact compilations of special gates. In parentheses the number of successful compilations vs. the number of trials starting with a different initial condition, and in the case of sequ with permutation, with a different permutation of the CNOT layout. Note that the Qiskit compilation is stochastic and can return a variable number of CNOTs. *After several trials, we have found a satisfactory layout, and the numbers correspond to this one.

Gate	Qiskit CNOTs		sequ		spin
	Full conn.	Line conn.	w/o perm.	w perm	
Toffoli 3 qubit	6	7-9	7 (19/100)	6 (27/100)	8 (38/100)
Toffoli 4 qubit	14	36	18 (4/350)	14 (1/100)*	18 (1/350)
Fredkin 3 qubit	7	10	8 (55/100)	7 (4/100)	8 (31/100)
1-bit full adder 4 qubit	10	16	10 (11/100)	10 (3/100)	14 (8/500)

so address property (3). The purpose of this section is to develop an algorithmic technique for compressing arbitrary structures, and cart in particular. Rather than choosing which CNOTs to keep in a structure (equivalently, which CNOTs to eliminate) *a priori*, we consider the question of how to design an algorithm that automatically finds the best compression for the target unitary U.

We approach this problems with two ideas: (1) we know that there are techniques to reduce consecutive CNOTs when no rotation gates are between them (let us call these compaction rules); (2) we know that, when optimizing for the angles, we can enforce sparsity of the solution by adding a pertinent regularization.

A closer look at (2) inspires us to enforce groups of four rotation angles following the CNOTs to be zero, thereby eliminating all the rotation gates after a CNOT. This leads naturally to a group LASSO regularization, and we explore it in Section 3.8.2.

A closer look at (1) suggests special compaction rules, which we discuss in Section 3.8.1.

Finally, the complete algorithm starts by enforcing sparsity, eliminating zero rotation gates, compressing the structure via compaction rules, and then re-optimizing with the regular gradient descent on the compacted structure. This algorithm is discussed in Section 3.8.3.

3.8.1 The "synthesis" algorithm

The guiding question for this subsection is: given a list of CNOTs, can we find a shorter list of CNOTs such that the matrix product, in $\mathrm{SU}(2^n)$, of the first list and the second list are equal.

While the research works in this area are many [108, 109, 110, 111, 112, 113, 101], we use here an adapted version of the "synthesis" algorithm of [108]. There the authors give an asymptotically optimal synthesis of CNOT circuits. The idea is that CNOTs on n-qubits can be identified with elementary matrices in $\mathrm{GL}(n, \mathbb{Z}_2)$. Given $A \in \mathrm{GL}(n, \mathbb{Z}_2)$, we can compute its LU decomposition in terms of elementary matrices. On the other hand, things are easier in our case, since we only consider downward-facing CNOTs, which can be related to $\mathrm{LT}(n, \mathbb{Z}_2)$, the group of n by n lower unitriangular matrices on $\mathbb{Z}_2$. This leads us to implement an adapted "synthesis" algorithm that consists of three steps: identifying CNOTs with their corresponding matrices in $\mathrm{LT}(n, \mathbb{Z}_2)$, multiplying them in $\mathrm{LT}(n, \mathbb{Z}_2)$, and reading off the locations of 1's in the product. We report the following theorem concerning this algorithm.

Theorem 10. *[108] Given an L-long circuit of downward-facing CNOTs, the "synthesis" algorithm correctly outputs an equivalent circuit of downward-facing CNOTs of length $\leq n(n-1)/2$ in $O(n^3 L)$ computations.*

The correctness of the algorithm follows from [108], the length follows from the fact that there are $n(n-1)/2$ lower-triangular entries in an n by n matrix, and the run-time is based on $L-1$ matrix multiplications. While the algorithm may output a word of length $n(n-1)/2$, it is possible for words to be further reduced. Hence, this "synthesis" algorithm is not optimal, but its simplicity is appealing. The lower bound on word lengths is $\Omega(n^2/\log(n))$ for $\mathrm{GL}(n, \mathbb{Z}_n)$, and one way to obtain it is to partition the matrix into blocks, as is done in [108]. Note that words can also be reduced via identities on three qubits, namely commutation rules, cancellations of two subsequent identical CNOTs, and mirror rules (see, e.g, [114]). However, these have little effect for $n > 3$.

We remark that, in general, the "synthesis" algorithm does not respect hardware connectivity, so it will be used only in cases in which hardware connectivity is not an issue. However, the three qubit identities do maintain the hardware connectivity constraints of the original circuit, so they can be used in all the cases. Note that there are recent works using Steiner trees to apply the synthesis algorithm in a way that does respect hardware connectivity: [115, 116, 117, 118, 119].

3.8.2 Setting parameters to zero

The guiding question for this subsection is: which parameters should we set to zero to get a good compressed structure via the techniques presented in the previous subsection? As mentioned, our approach is to enforce sparsity of the resulting $\boldsymbol{\theta}$ vector by pushing groups of the four-rotation angles following a CNOT to be zero (i.e., all the rotations of a given CNOT unit). This can be achieved via a group Lasso regularization [92] (see also [120, 91, 121]), as follows. The collection of rotation angles for each of the CNOT units is the vector $\boldsymbol{\theta}_\ell := [\theta_{3n+4\ell-3}, \ldots, \theta_{3n+4\ell}]$, with $\ell = \{1, \ldots, L\}$. Enforcing each of these vectors to be zero, amounts to adding a regularization of the form $\|\boldsymbol{\theta}_\ell\|_2$ for each group, and therefore solving the problem:

$$\textbf{(AQCP-}\boldsymbol{\theta}, \lambda\textbf{)} \qquad \min_{\boldsymbol{\theta}\in[0,2\pi)^p, \mathsf{ct}=\overline{\mathsf{ct}}} f_{\mathsf{ct}}(\boldsymbol{\theta};\lambda) := \frac{1}{2}\|V_{\mathsf{ct}}(\boldsymbol{\theta}) - U\|_{\mathrm{F}}^2 + \lambda \sum_{\ell=1}^{L} \|\boldsymbol{\theta}_\ell\|_2, \tag{3.10}$$

with regularization parameter $\lambda > 0$, which trades off approximation error and sparsity.

We can solve (3.10) by a proximal gradient descent, as done in [92]; although problem (3.10) is non-convex, we have similar convergence results as for gradient descent, meaning that for small regularization parameters λ, we can show convergence of perturbed proximal gradient descent to a second-order stationary point [122].

In particular, proximal gradient descent amounts to computing a gradient descent step and then applying a proximal operator (which, in this case, is the block-wise soft-thresholding operator [123, Eq.(7)]). Starting from a randomly initialized $\boldsymbol{\theta}[0]$, this yields the component-wise recursion for $k = 3n + 4\ell - 3, \ldots, 3n + 4\ell$, for $\ell = 1, \ldots, L$, as

$$[\boldsymbol{\theta}_\ell^+]_k = \theta_k[t] + \alpha \mathbb{R}\mathsf{e}\ \mathsf{Tr}\left[\frac{\partial}{\partial\theta_k} V_{\mathsf{ct}}(\boldsymbol{\theta}[t])^\dagger U\right] \quad \text{for all } k \in \{3n+4\ell-3, \ldots, 3n+4\ell\} \tag{3.11a}$$

$$\theta_k[t+1] = \frac{[\boldsymbol{\theta}_\ell^+]_k}{\|\boldsymbol{\theta}_\ell^+\|}(\|\boldsymbol{\theta}_\ell^+\| - \alpha\lambda)_+ \qquad \text{for all } k \in \{3n+4\ell-3, \ldots, 3n+4\ell\} \tag{3.11b}$$

$$\theta_k[t+1] = \theta_k[t] + \alpha \mathbb{R}\mathsf{e}\ \mathsf{Tr}\left[\frac{\partial}{\partial\theta_k} V_{\mathsf{ct}}(\boldsymbol{\theta}[t])^\dagger U\right] \qquad \text{for all } k \in \{1, \ldots, 3n\}, \tag{3.11c}$$

where $(y)_+ = \max\{y, 0\}$.

Eq.s 3.11 are block-wise recursions: for each CNOT unit, we run the gradient descent for each

angle of the unit, and then run the proximal operator. For the angles not belonging to the CNOT units, then it is business as usual.

3.8.3 Compression algorithm

Integrating the ideas from the previous two subsections, we propose the following algorithm for compressing an arbitrary structure.

Algorithm 2 Compression via group LASSO

Require: U, ct, λ, initial condition $\boldsymbol{\theta}[0]$

1: compute $\boldsymbol{\theta}^*_{\mathrm{GL}}$ via proximal gradient descent (3.11) on $f_{\mathsf{ct}}(\cdot, \lambda)$

2: Eliminate all the rotation gates that have zero rotation angle

3: compress ct via the "synthesis" algorithm

4: further compress ct via CNOT identities

5: compute $\boldsymbol{\theta}^*$ via standard gradient descent (3.8) on $f_{\mathsf{ct}'}$, where ct' is the compressed structure

6: **return** compressed structure, ct', and corresponding angles, $\boldsymbol{\theta}^*$

The algorithm consists of running proximal gradient descent (3.11), starting with some given initial condition $\boldsymbol{\theta}[0]$, on the regularized cost $f_{\mathsf{ct}}(\cdot, \lambda)$. Then, we eliminate all the rotation gates that have zero rotation angle and apply the compaction rules to reduce the circuit. Finally, we run standard gradient descent (3.8) on $f_{\mathsf{ct}'}$, where ct' is the compressed structure, to compute the best parameters $\boldsymbol{\theta}^*$ for the compressed structure.

Line 3 (i.e., "synthesis") can be included, when hardware connectivity constraints are not important, or not, when they are.

3.8.4 Numerical tests: random unitary matrices, towards G3

We test our compression algorithm on random unitary matrices with sequ and cart structures for both $n = 3$ and $n = 5$ qubit circuits (we do not consider spin because it corresponds to the line connectivity which is not preserved under the synthesis algorithm). In particular, we randomly

initialize the iterates and consider 100 different random unitary matrices, and we plot both mean and standard deviation of the result.

We report our results in Figure 3.8. We start with circuits of $L = 14$ and $L = 250$ for sequ and from $L = 22$ and $L = 528$ for cart, respectively for $n = 3$ and $n = 5$, and we compress them with different regularization parameters. In the x-axis, we can see the regularization parameter λ used, while in the y-axis, we can note the final number of CNOTs (in blue), as well as the Frobenius fidelity $\bar{F}_{\mathrm{F}}(U, V)$ (in orange). The lines correspond to the average, and the shaded areas to one standard deviation.

As one can appreciate, a small compression is possible for sequ, especially if the "synthesis" algorithm is used. Note that the three qubit reductions have little affect for $n = 5$ and this is true for all $n \geq 5$.

A better compression, with very high Frobenius fidelity $\bar{F}_{\mathrm{F}}(U, V)$, is possible instead for cart, showing that this structure can be tuned to be compressed to TLB ($L = 14$ and $L = 252$, respectively) without losing fidelity. One can see this by looking at the second data point on the left for both $n = 3$ and $n = 5$, where we obtain a reduction to $L = 14$ and $L < 252$ (blue curve), with no practical loss of fidelity (orange curve). This is encouraging (meaning that the recursive Cartan decomposition can be easily compressed in practice) and supports properties (3) and (4) for cart.

3.8.5 Numerical tests: compressing compiled circuits, towards G3 in Qiskit

Finally, we move to analyze the effect of the compression algorithm to already compiled circuits in Qiskit, or other compilers. The idea here is to see how one can use the approximate quantum compiler as an add-on to the usual workflow, by allowing the user to trade-off accuracy and circuit depth, as defined in Remark 5.

We consider the quantum circuits of $3, 4, 5$ qubits in the [1] database. We compile them in Qiskit and transform them into the parametric circuit of the present chapter (see Figure 3.2 for an example). Then we use this compiled circuit as a warm start for our compression algorithm, Algorithm 2 for different λ's. This yields the graphs in Figure 3.9, where we can appreciate how λ

affects compression, as well as Frobenius fidelity.

A better overview is offered by Table 3.3, where we look at the best compression we can obtain for a Frobenius fidelity as high as 90%, and its associated computational overhead. To compile the table, we have looked at different λ's parameters, and we have considered the best compression with fidelity $\geq 90\%$. The column 'Depth' refers to the depth of the original circuit in the database (which can have gates not in the gate set), the Qiskit depth is the circuit depth once compiled in Qiskit and transformed into the parametric circuit, the compressed depth is the minimal depth that we were able to obtain with Algorithm 2, with a fidelity $\geq 90\%$ (by varying λ), the fidelity column is the Frobenius fidelity of the maximal compressed circuit, while the overhead is the time that the computation of the latter circuit has taken.

As one can see, we can achieve some compression with a small loss in fidelity, and in some cases, without any loss. For some circuits, compression is quite high ($\sim 58\%$ of the Qiskit depth), for others, less so.

We remark that, ultimately, the compression capabilities are a by-product of Qiskit (or others) compilation properties. If the compiler used can achieve optimal compilation, no further compression can be obtained, no matter how sophisticated the devised algorithm is. While compilation is getting better and better (new exact or heuristic algorithms are proposed at a rapid pace, see for instance [78]), we believe that our tool can **also** be used to globally gauge if circuits can be further compressed and which part of the circuit is not optimally compiled. This has immediate practical applications in devising better compilation algorithms. For example, further studies are needed to understand what makes 'miller_11' or 'decod24-v2_43' difficult for Qiskit, and how to improve its compiler.

Finally, more research has to be dedicated towards devising better strategies to select the best parameter λ for compression, instead of a random search, and analyzing what happens with the use of different compilers rather than Qiskit.

3.9 Conclusions and open points

Table 3.3: Best achieved compression for the circuit of the [1] database, along with their Frobenius fidelity and computational overhead. Depth is the original depth; Qiskit depth is the depth once compiled in Qiskit and transformed into the parametric circuit; Compr. depth is the best compression obtained with Fr. Fidelity $\geq 90\%$, while its indicated % represents the ratio of the compressed depth w.r.t. the Qiskit depth.

File name	Depth	Qiskit depth	Fr. Fidelity $\geq 90\%$ Compr. depth (& %)	$\bar{F}_{\mathrm{F}}$ [%]	Overhead [s]
4gt5_77	74	137	136 (99%)	93.41	24.34
4gt13_91	61	119	119 (100%)	100.0	20.76
one-two-three-v0_98	82	163	163 (100%)	100.0	23.7
4gt13_90	65	121	120 (99%)	99.99	20.63
ham3_102	13	25	24 (96%)	99.99	1.28
one-two-three-v1_99	76	152	152 (100%)	100.0	23.65
4gt5_76	56	115	111 (97%)	98.75	20.57
4mod7-v0_94	92	175	171 (98%)	95.22	27.19
aj-e11_165	86	177	177 (100%)	100.0	36.16
alu-v0_26	49	99	95 (96%)	99.99	17.26
miller_11	29	65	38 (58%)	99.99	3.1
rd32-v1_68	21	35	34 (97%)	92.02	4.58
one-two-three-v2_100	40	79	79 (100%)	100.0	13.82
one-two-three-v3_101	40	80	79 (99%)	91.84	16.85
4mod5-v0_20	12	25	25 (100%)	100.0	6.01
alu-v0_27	21	43	41 (95%)	99.99	7.8
mod5mils_65	21	44	44 (100%)	100.0	8.61
ex-1_166	12	28	26 (93%)	99.99	1.28
decod24-v1_41	50	94	93 (99%)	99.99	16.32
alu-v3_34	30	60	60 (100%)	100.0	11.45
3_17_13	22	45	37 (82%)	99.98	2.11
decod24-v3_45	84	157	155 (98%)	98.4	25.86
4gt11_84	11	21	21 (100%)	100.0	2.82
decod24-v0_38	30	62	52 (84%)	99.99	5.48
4mod5-v0_19	21	45	41 (91%)	95.94	8.76
4mod5-v1_22	12	29	29 (100%)	100.0	6.06
alu-v1_29	22	43	41 (95%)	99.99	6.98
alu-v1_28	22	45	39 (87%)	99.99	7.18
4mod5-v1_23	41	83	76 (92%)	99.99	14.79
4mod5-v0_18	40	84	73 (87%)	99.99	11.06
rd32_270	47	96	84 (88%)	99.81	20.94
4gt10-v1_81	84	159	159 (100%)	100.0	29.35
rd32-v0_66	20	33	33 (100%)	100.0	3.73
alu-v3_35	22	44	42 (95%)	99.99	7.77
4gt13-v1_93	39	70	70 (100%)	100.0	14.66
4mod7-v1_96	94	164	164 (100%)	100.0	24.4
4mod5-v1_24	21	41	38 (93%)	97.4	8.68
mod5d1_63	13	30	30 (100%)	100.0	8.92
alu-v4_36	66	117	117 (100%)	100.0	20.24
4gt11_82	20	42	42 (100%)	100.0	7.18
4gt5_75	47	88	88 (100%)	100.0	16.24
alu-v2_33	22	42	38 (90%)	99.99	6.96
alu-v2_32	92	174	174 (100%)	100.0	25.21
4gt11_83	16	41	37 (90%)	99.95	9.18
decod24-v2_43	30	65	47 (72%)	99.99	5.23
4gt13_92	38	71	71 (100%)	100.0	14.1
alu-v4_37	22	44	42 (95%)	99.99	7.73
mod5d2_64	32	67	67 (100%)	100.0	14.08

In this chapter, we have examined, in deep mathematical and numerical detail, variants of the best approximate quantum compiling problem. We have shown how to build hardware-aware structures and how to optimize over them. While we have presented encouraging results to support various theoretical and numerical properties, several open points are left for future research. In particular, on top of our priority list are (1) to investigate Conjecture 2 and analytically determine the relationship between approximation error and number of CNOTs (which has been done for 2-qubits using the Weyl chamber [124, 125, 126, 53]); (2) to investigate Conjecture 1; and (3) to investigate properties (5) and (6) for cart.

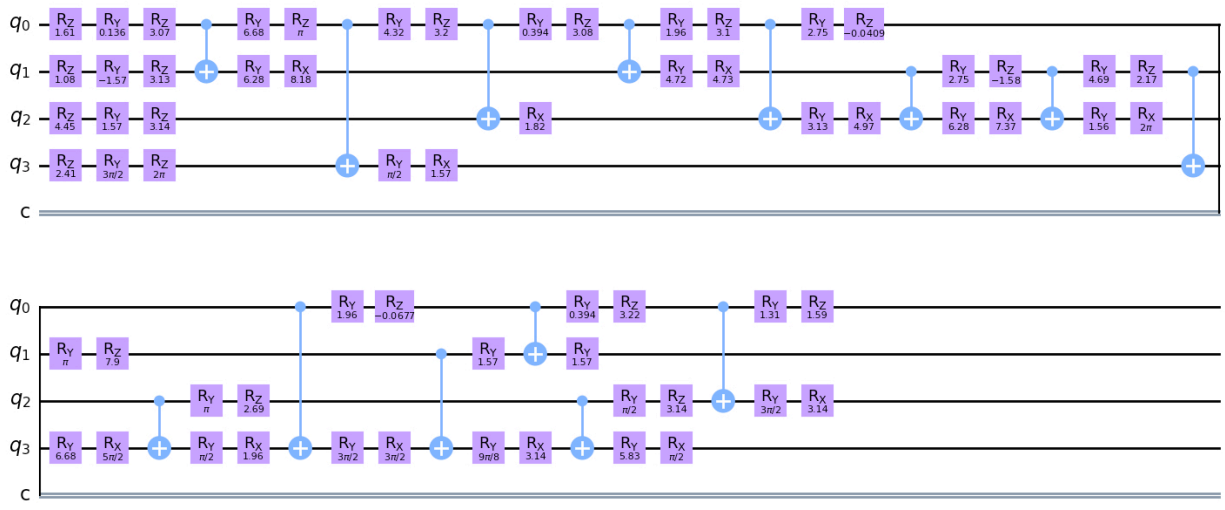


Figure 3.7: A different from Qiskit 4-qubit Toffoli exact compilation, obtained by permuting the CNOT units in the sequ(14) structure and running gradient descent with many initial points.

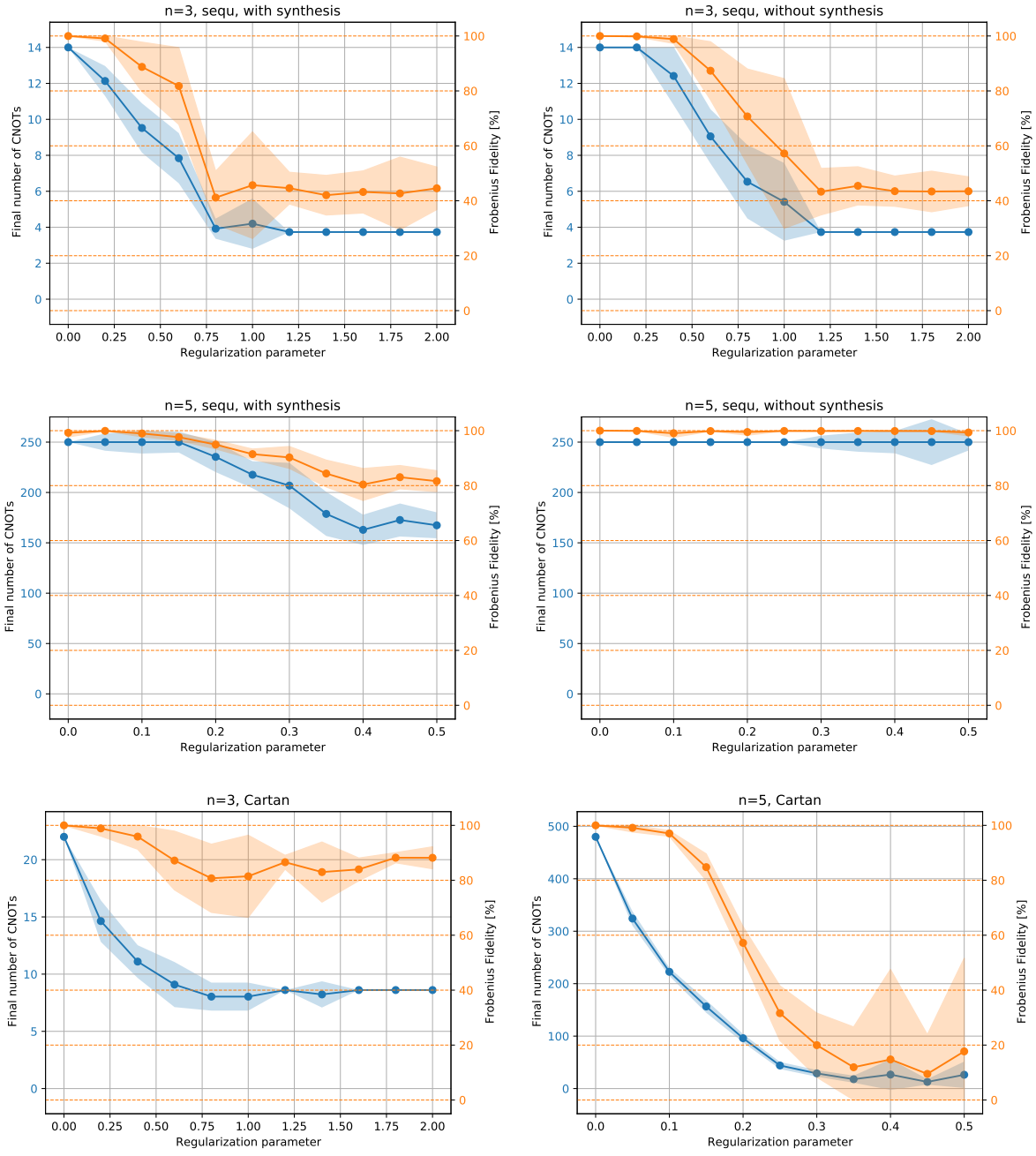


Figure 3.8: Final number of CNOT units and Frobenius fidelity, for $n = 3$ and $n = 5$ qubit circuits and different structures. We divide the cases for sequ for when "synthesis" is run, and when it is not. For cart, since hardware constraints are not imposed, "synthesis" is always run. For all graphs, the x-axis represents the regularization parameter λ, while the y-axis represents the compression obtained, in blue, and the Frobenius fidelity, in orange. The continuous line is the average, while the shaded area is one standard deviation.

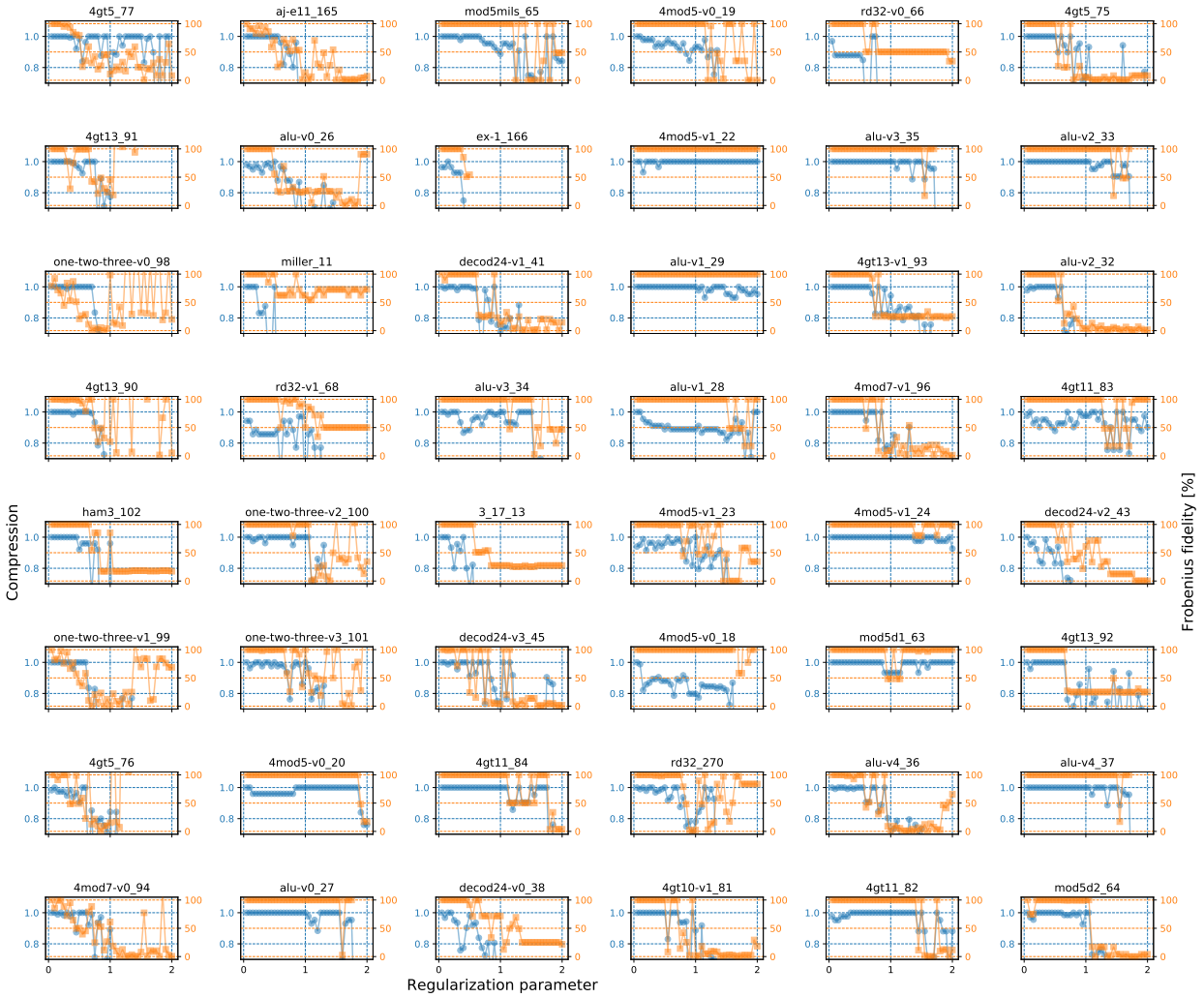


Figure 3.9: Compression of compiled circuits from the [1] database, for various regularization parameter values. For all graphs, the x-axis represents the regularization parameter $[0-2]$, while the y-axis the compression obtained in blue (normalized to the starting CNOT count), and the Frobenius fidelity in orange.

Chapter 4

High-probability convergence bounds for non-convex stochastic gradient descent

4.1 Introduction

Stochastic gradient descent (SGD) is an algorithm for minimizing a cost function by relying on noisy estimates of the gradient. Theoretical results provide guarantees for the convergence of SGD to a stationary point via convergence bounds in mean, almost sure convergence, or convergence bounds in probability. All three depend on the properties of both the cost function and the noise distribution. In particular, we consider the follow settings:

(S1) where the cost function is strongly smooth and satisfies the Polyak-Łojasiewicz (PL) inequality, and the noise is norm sub-Gaussian;

(S2a) where the cost function is strongly smooth, and the noise is norm sub-Gaussian; and

(S2b) where the cost function is strongly smooth and Lipschitz continuous, and the noise is norm sub-Weibull.

The PL inequality is a weaker form of strong convexity that does not imply convexity and is motivated by recent papers that prove PL-type inequalities for neural nets [127, 128, 129, 130]. The sub-Weibull noise assumption is weaker than the sub-Gaussian noise assumption and is motivated by recent results where the noise for empirical risk minimization applied to particular problems is not sub-Gaussian for small batch-sizes [131, 132, 133].

Almost sure convergence to a first-order stationary point can be proved assuming only strong

smoothness and a weak assumption on the noise [35, 134]. In this case, the mean convergence rate of the squared gradient norm to zero is $O(1/\sqrt{T})$, where T denotes the number of iterations [135, 136]. It is also possible to prove a matching almost sure convergence *rate* [137]. If the PL inequality is also assumed, then the mean convergence rate of the cost function of the iterates to the minimum is $O(1/T)$ [52, 136]. However, a convergence guarantee is generally required with some arbitrary probability, $1 - \delta$. A mean convergence rate is acceptable when it is possible to re-run SGD many times, since Markov's inequality can be applied and the best run chosen. On the other hand, if only one run of SGD is allowed, then the $(1/\delta)$-dependence of Markov's inequality quickly blows up. What is needed is a bound with logarithmic dependence on $1/\delta$. Assuming strong smoothness and norm sub-Gaussian noise, [138] proves a $O(\log(T)\log(T/\delta)/\sqrt{T})$ convergence bound. We do not know of any other convergence bounds in the strongly smooth setting that have only logarithmic dependence on δ. In Section 4.2, we introduce the preliminaries; delineate the optimization assumptions, the types of convergence, and the noise assumptions; and discuss how our results fit into the broader literature.

Section 4.3 is dedicated to setting (S1). In Section 4.3.1, we prove a $O(\log(1/\delta)/T)$ convergence bound, which matches the optimal convergence rate in mean of $O(1/T)$. In Section 4.3.2, we focus on the statistical learning framework and consider SGD with a mini-batch and sub-Gaussian (as a random vector) noise. We prove, as a simple application of our convergence analysis, a bound on the true risk that goes to the empirical minimum as the sample size increases to infinity. The bound has $1/\sqrt{\delta}$ dependence on δ, which, while not logarithmic, is better than the $1/\delta$ that comes from the mean convergence bound and Markov's inequality. Furthermore, it is based on a conservative step-size constant and a fixed number of epochs, corresponding to practice where a better step-size constant for convergence may paradoxically fit random labels [139] and where early stopping is used to avoid over-fitting.

Section 4.4 is dedicated to setting (S2). The convergence results are presented in Section 4.4.2. For setting (S2a), we prove a $O(\log(T)\log(1/\delta)/\sqrt{T})$ convergence bound, improving the log factors of the bound in [138]. Moreover, our bound is for an argmin over a small number of iterates, while the

bound in [138] is for an argmin over all of the iterates. This last point is an independent contribution of the chapter and is presented in Theorem 16 where we call it "efficient post-processing." For setting (S2b), we prove a $O(\log(T/\delta)^\theta \log(1/\delta)/\sqrt{T})$ convergence bound, where θ is the sub-Weibull tail weight. On our way to proving the non-convex (without the PL inequality) convergence bounds, we prove a Freedman-type inequality for martingale difference sequences with sub-Weibull tails. Previously, such inequalities only allowed for, at most, sub-exponential tails, as in [140]. In order to push beyond the sub-exponential boundary, we used the moment generating function truncation techniques of [141]. The additional constants that the sub-Weibull noise introduces fortunately balance in such a way that we are still able to get a $1/\sqrt{T}$ sub-linear convergence rate up to logarithmic factors. Section 4.4.1 presents the sub-Weibull Freedman-type concentration inequality.

Section 4.5 provides numerical examples to elucidate the theoretical results. In particular, SGD is applied to various neural-net-based synthetic problems.

4.2 Preliminaries

We are interested in the unconstrained optimization problem

$$\min_{x\in\mathbb{R}^d} f(x), \tag{4.1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is a differentiable function, and the SGD iteration

$$x_{t+1} = x_t - \eta_t g_t \ \forall t \in \mathbb{N}\cup\{0\},$$

where $g_t \in \mathbb{R}^d$ is an estimate of $\nabla f(x_t)$, η_t is the step-size, and $x_0 \in \mathbb{R}^d$ is the initial point. Formalized in this way, it remains to specify: (1) the properties of f, the cost function; (2) the types of convergence of (x_t) to a stationary point; and (3) the properties of the noise $e_t \coloneqq \nabla f(x_t) - g_t$. We restrict our attention to the setting where x_0 is deterministic, but the results easily extend to the setting where x_0 is a random vector.

One of the main examples of Eq. (4.1) is the stochastic approximation problem: $f = \mathbb{E}[F(\cdot,\xi)]$ where $(\Omega,\mathcal{F},P)$ is a probability space and $F : \mathbb{R}^d \times \Omega \to \mathbb{R}$ [142, 135, 143]. In this case, we independently sample $\xi_0, \xi_1, \ldots$ and set $g_t = \nabla F(x_t, \xi_t)$.

Note that we frequently use big-O and little-o notation. When comparing two sequences of real numbers, (a_t) and (b_t): $a_t = o(b_t)$ if $\lim a_t/b_t = 0$, $a_t = O(b_t)$ if $\limsup |a_t|/b_t < \infty$, $a_t = \Omega(b_t)$ if $b_t = O(a_t)$, $a_t = \omega(b_t)$ if $b_t = o(a_t)$, and $a_t = \Theta(b_t)$ if $a_t = O(b_t)$ and $a_t = \Omega(b_t)$. We use $a_t = \text{poly}(b_t)$ to denote that there is a polynomial function p such that $a_t = O(p(b_t))$. Finally, we use $[n]$ to denote the set $\{1, \ldots, n\}$.

4.2.1 Optimization Assumptions

Throughout the chapter we implicitly assume $\text{argmin}_{x \in \mathbb{R}^d} f(x)$ is non-empty so it is a well-posed problem, and hence $f^\star \coloneqq \min_{x \in \mathbb{R}^d} f(x) > -\infty$. We assume that f is L-strongly smooth (or L-smooth for short) for a given $L > 0$; i.e., $f \in C^1$ and its gradient is L-Lipschitz continuous.

A function f is μ-Polyak-Łojasiewicz (or μ-PL for short) if $\|\nabla f(x)\|^2 \geq 2\mu(f(x) - f^\star)\ \forall x \in \mathbb{R}^d$. Some examples of problems with PL objectives are logistic regression [52] and matrix factorization [144]. Deep linear neural networks satisfy the PL inequality in large regions of the parameter space [145, Thm. 4.5], as do two-layer neural networks with an extra identity mapping [127]. Furthermore, sufficiently wide recurrent neural networks satisfy the PL inequality locally around random initialization [128, 129, 130]. While strong convexity implies the PL inequality, a PL function need not even be convex, hence the PL condition is considerably more applicable than strong convexity in the context of neural networks.

Recall that we focus on two main settings with regards to optimization assumptions: (S1) the case where f is L-smooth and satisfies the PL inequality, and convergence is to a global optimum, and (S2) the more general case where f is L-smooth, and convergence is to a stationary point. We do not require convexity in either case, though we sometimes refer to the second case as the "non-convex" case since the convergence is to a stationary point as is typical for non-convex analysis.

We also note that some of the non-convex results require f to be ρ-Lipschitz continuous; the chapter will state when this assumption is needed. Lipschitz continuity of f follows immediately if the iterates are bounded. This might lead one to consider projected SGD, but there are certain issues preventing us from doing so which we discuss in Appendices A.5 and A.10.

Further details on standard optimization terminology and results are in Appendix A.1.

4.2.2 Types of Convergence

To analyze the convergence of SGD, we define a sequence of pertinent random variables (X_T). In the PL setting, since the PL inequality implies that all stationary points are global minima, we set $X_T = f(x_T) - f^\star$. In the non-convex setting, since SGD may converge to a stationary point, we set $X_T = \min_{0 \leq t \leq T-1} \|\nabla f(x_t)\|^2$ or $X_T = \|\nabla f(y_T)\|^2$ where y_T is carefully chosen from $x_0 \ldots, x_{T-1}$. Note that we do not consider iterate averaging in either case because the usual trick using Jensen's inequality requires convexity, so a bound on the averaged error (i.e., the regret) does not translate to a bound on the error of the average iterate.

Results bounding $\mathbb{E}[X_T]$ (expected/mean/L_1 convergence) for SGD are common but strong results bounding X_T with $(1-\delta)$-confidence are relatively rare. See [142], [146], [147], [148], [149], and [150] for such bounds in the non-smooth convex setting. Clearly, guarantees with high confidence are quite desirable. We can go from expectation to probability via Markov's inequality, $P(X_T \leq \mathbb{E}[X_T]/\delta) \geq 1 - \delta$, but we want the dependence on δ to be better than $1/\delta$ multiplicative dependence. In particular, we want poly$(\log(1/\delta))$, which we call "high probability" dependence, as opposed to poly$(1/\delta)$, which we call "low probability" dependence.

One way to get high probability dependence from Markov's inequality is through probability amplification, which involves taking multiple runs and picking the best run based on a validation set; for example, Exer. 13.1 of [25] describes probability amplification for the true risk. Their procedure takes $\Theta(\log(1/\delta))$ runs and $\Theta(\log(1/\delta)/\epsilon^2)$ validation samples and relies on the boundedness of the loss function. [151] describes probability amplification for convergence in the strongly convex setting. They use strong convexity to apply robust distance estimation and get around the $\Omega(1/\epsilon^2)$ sample requirement for mean estimation [152]. Thm. 2.4 of [135] describes probability amplification for convergence in the non-convex setting. Their procedure takes $\Theta(\log(1/\delta))$ runs and $\Theta(\log(1/\delta)/(\delta\epsilon))$ validation samples and relies on the boundedness of the variance of the norm of the gradient noise.

The above probability amplification techniques involve both an optimization cost and a

validation cost. Re-running SGD multiple times clearly increases the optimization cost, so our goal is to prove high probability convergence bounds for a *single* run of SGD that match the mean convergence bounds *without* increasing the validation cost. The only bound we have that requires validation is the non-convex bound since it is in terms of $\min_{0\leq t\leq T-1}\|\nabla f(x_t)\|^2$. If we want a bound in terms of $\|\nabla f(x_s)\|^2$ for a particular s, then we have to compute the argmin, which means performing validation on T choices instead of $\Theta(\log(1/\delta))$. Fortunately, Theorem 16 shows that there is a workaround involving only $\Theta(\log(1/\delta))$ iterates, same as for multiple runs. Then the same validation step as in [135] can be applied.

Further details on the convergence of random variables can be found in Appendix A.2.

4.2.3 Noise Assumptions

We make assumptions involving the bias, variance, and tail behavior of the noise e_t. First, let $(\Omega, \mathcal{F}, P)$ be the underlying probability space and define the natural filtration $\mathcal{F}_t = \sigma(e_0, \ldots, e_t)\ \forall t \geq 0$, $\mathcal{F}_{-1} = \{\emptyset, \Omega\}$. We assume the noise is unbiased, $\mathbb{E}\left[e_t \mid \mathcal{F}_{t-1}\right] = 0$, and that the variance is bounded by a constant, $\mathbb{E}\left[\|e_t\|^2 \mid \mathcal{F}_{t-1}\right] \leq \sigma^2$. It is possible to prove almost sure convergence of SGD in the smooth non-convex setting with weaker assumptions on the bias, namely that there is an additional noise term ϵ_t in the gradient with the properties $c_1\|\nabla f(x_t)\|^2 \leq -\langle \nabla f(x_t), \epsilon_t\rangle$ and $\|\epsilon_t\| \leq c_2\left(1 + \|\nabla f(x_t)\|\right)$ for all t, and on the variance, namely that $\mathbb{E}\left[\|e_t\|^2 \mid \mathcal{F}_{t-1}\right] \leq \sigma^2\left(1 + \|\nabla f(x_t)\|^2\right)$ for all t [35, 134]; but it is not clear how to do so for high probability convergence.

Next, while mean convergence rates do not require assumptions on the tail behavior of the noise, high probability convergence rates do. In particular, [142] makes the assumption $\mathbb{E}\left[\exp\left(\|e_t\|^2/\sigma^2\right)\right] \leq \exp(1)$, which corresponds to sub-Gaussian tail behavior. Note that this tail assumption implies the variance assumption via Jensen's inequality. We consider sub-Gaussian, sub-exponential, and sub-Weibull noise [153, 154, 141], defined below:

Definition 4. *A random variable X is K-sub-Gaussian if $\mathbb{E}\left[\exp\left(X^2/K^2\right)\right] \leq 2$. See Prop. 2.5.2 of [155] for equivalent definitions.*

Definition 5. *A random variable X is K-sub-exponential if $\mathbb{E}\left[\exp\left(|X|/K\right)\right] \leq 2$. See Prop. 2.7.1 of [155] for equivalent definitions.*

Definition 6. *A random variable X is K-sub-Weibull(θ) if $\mathbb{E}\left[\exp\left((|X|/K)^{1/\theta}\right)\right] \leq 2$. The tail parameter θ measures the heaviness of the tail—higher values correspond to heavier tails—and the scale parameter K gives us the following bound on the second moment, $\mathbb{E}\left[X^2\right] \leq 2\Gamma(2\theta+1)K^2$ (Lemma 5). See Thm 2.1 of [153] for equivalent definitions. Note that sub-Gaussian and sub-exponential are special cases with $\theta = \frac{1}{2}$ and $\theta = 1$, respectively.*

While many works in the context of SGD consider errors modeled by sub-Gaussian distributions, we provide a brief discussion on the relevance of the sub-Weibull assumption. First, [156] shows that a Gaussian prior on the weights in a Bayesian neural network induces a sub-Weibull distribution on the weights with the optimal tail parameter for a particular layer proportional to its depth. Secondly, a series of recent papers [131, 132, 133] suggest that SGD exhibits noise with tails that are heavier than sub-Gaussian. [157] and [158] suggest that the heavier-tailed noise of SGD may be vital to its ability to find machine learning models that perform well on unseen data.

To the best of the authors' knowledge, all existing results for the high probability convergence rate of SGD assume norm sub-Gaussian noise. While the central limit theorem can be used to justify the sub-Gaussian noise assumption for mini-batch SGD with large batch-sizes, it cannot for small batch-sizes. One alternative distribution that has been suggested is the α-stable distribution with $\alpha < 2$, which has infinite variance. [132] and [131] empirically determine the tail-index for stochastic gradient noise assuming it follows an α-stable distribution and [159] proves mean convergence bounds for clipped SGD in this setting. However, none of these papers actually show that the noise is α-stable, or even that it has infinite variance. On the other hand, [133] tests when the noise is or is not Gaussian. They find that the noise is Gaussian for a batch-size of 4096, is not for a batch-size of 32, and starts out Gaussian then becomes non-Gaussian for a batch-size of 256.

Even though sub-Weibull noise has finite variance, by relaxing the noise assumption from sub-Gaussian to sub-Weibull, we are taking a step towards a more realistic analysis of SGD.

Remark 7. *Our noise assumptions will be on the Euclidean norm of the noise vector. We note that it is straightforward to show that if* $(e_i) \in \mathbb{R}^d$ *is component-wise sub-Weibull, then its norm is sub-Weibull. Concretely, assume* e_i *is* K_i*-sub-Weibull*(θ) *for* $i = 1, \ldots, d$*. Let* $\|\!|\cdot|\!\|_\theta$ *denote the Orlicz norm (infimum of all possible scale parameters given the tail parameter* θ*). Then*

$$\|\!|\ \|(e_i)\|\ |\!\|_\theta = \left\|\!\left|\left(\sum_{i=1}^d e_i^2\right)^{1/2}\right|\!\right\|_\theta = \left\|\!\left|\sum_{i=1}^d e_i^2\right|\!\right\|_{2\theta}^{1/2} \leq \left(\sum_{i=1}^d \|\!|e_i^2|\!\|_{2\theta}\right)^{1/2} = \left(\sum_{i=1}^d \|\!|e_i|\!\|_\theta^2\right)^{1/2} = \|(K_i)\|.$$

Note that the components may be dependent.

4.2.4 Prior Work and Contributions

Assuming strong smoothness and strong convexity, the tight **mean** convergence rate is $O(1/T)$ [142]. The same mean convergence rate can be shown assuming strong smoothness and the PL inequality [52, 160]. Assuming strong smoothness, the PL inequality, and norm sub-Gaussian noise, our Theorem 11 provides a matching $O(\log(1/\delta)/T)$ convergence rate for the **high probability** case. Our theorem depends on Proposition 1 which is a novel probability result, essentially saying that adding two kinds of noise to a contracting sequence—sub-Gaussian noise with variance depending on the sequence itself and sub-exponential noise—results in a sub-exponential sequence.

We then apply our convergence analysis to the statistical learning problem. We find that by taking a more conservative step-size, the training error goes to zero slower, but the generalization error also increases slower in such a way that we can balance the two. In particular, we find a bound that supports running multiple, but limited epochs of mini-batch SGD. For papers developing the generalization analysis that we use, see [161], [162], and [163]. Other relevant papers are [164], [145], [165], [166], and [167]. Note that [166] proves high probability generalization bounds for SGD, but only in the convex setting. Also, they balance their generalization result with the convergence result of [149].

Assuming strong smoothness and convexity, the tight mean convergence rate of the squared gradient norm is $O(1/\sqrt{T})$ [3, Thm. 5.3.1]. In fact, this is the optimal rate for all stochastic first-order methods assuming only strong smoothness (and not convexity) [168]. The same mean

convergence rate can be shown for SGD assuming only strong smoothness [135]. However, the result of [135] requires a constant step-size equal to $\Theta(1/\sqrt{T})$ to get the $O(1/\sqrt{T})$ convergence rate. On the other hand, for a $\Theta(1/\sqrt{t})$ step-size sequence, the rate is $O(\log(T)/\sqrt{T})$. Assuming strong smoothness and norm sub-Gaussian noise, [138] proves a $O(\log(T)\log(T/\delta)/\sqrt{T})$ convergence rate. Our non-convex convergence result, Theorem 15, improves this to $O(\log(T)\log(1/\delta)/\sqrt{T})$ and, assuming strong smoothness, Lipschitz continuity, and norm sub-Weibull noise, proves a $O((\log(T/\delta)^{\theta}\log(1/\delta) + \log(T)\log(1/\delta)^{2\theta})/\sqrt{T})$ convergence rate. To the best of the authors' knowledge, all existing results for the high probability convergence rate of SGD assume norm sub-Gaussian noise.

Another contribution is that our non-convex convergence rates are for an argmin over $\lceil \log(1/\delta) \rceil$ iterates while the bound of [138] is for an argmin over all of the iterates. We are able to restrict the number of iterates we have to minimize over via a post-processing sampling strategy that extends the sampling strategy of [135] from mean to probability. The crux of the proof is Lemma 15. Then the minimization step can be done via a validation set as in [135]. One takeaway from our convergence rate is that the optimal step-size constant for mean convergence only depends on the variance, whereas the optimal step-size constant for high probability convergence depends on δ and θ as well.

Our proof of Theorem 15 depends on a novel probability result, Proposition 2. The probability result is of independent interest because it extends, for the first time, Freedman-type concentration inequalities beyond the sub-exponential threshold [169, 140, 149]. In particular, we use the truncated MGF technique of [141] to generalize the Generalized Freedman inequality of [149] from sub-Gaussian to sub-Weibull martingale difference sequences.

4.3 Smooth, PL, Sub-Gaussian Setting

In this section, we assume the PL inequality and norm sub-Gaussian gradient noise. In Section 4.3.1, we consider a stochastic recursion with two types of noise: sub-Gaussian with dependence on the main sequence and sub-exponential. In Proposition 1, we prove that the main sequence is sub-exponential. Then, we show that the error of SGD satisfies just such a recursion,

allowing us to prove the high probability convergence bound in Theorem 11.

In Section 4.3.2, we consider the statistical learning framework as an example. It turns out a more conservative step-size constant is needed, which gives a slower convergence rate; this is shown in Theorem 13. Finally, we provide the resulting true risk bound in Theorem 14.

4.3.1 Convergence

We start our analysis by recalling that strong smoothness of the function f implies that

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

for any $x, y \in \mathbb{R}^d$. Set $x = x_{t+1}$ and $y = x_t$; then, by subtracting $f^\star$ from both sides and substituting in $x_{t+1} = x_t - \eta_t g_t = x_t - \eta_t(\nabla f(x_t) - e_t)$, we get

$$\begin{aligned} f(x_{t+1}) - f^\star &\leq f(x_t) - f^\star - \eta_t \left(1 - \frac{L\eta_t}{2}\right) \|\nabla f(x_t)\|^2 + \eta_t(1 - L\eta_t)\langle \nabla f(x_t), e_t \rangle + \frac{L\eta_t^2}{2}\|e_t\|^2 \quad (4.2) \\ &\leq f(x_t) - f^\star - \frac{\eta_t}{2}\|\nabla f(x_t)\|^2 + \eta_t(1 - L\eta_t)\langle \nabla f(x_t), e_t \rangle + \frac{L\eta_t^2}{2}\|e_t\|^2 \\ &\leq (1 - \mu\eta_t)(f(x_t) - f^\star) + \eta_t(1 - L\eta_t)\langle \nabla f(x_t), e_t \rangle + \frac{L\eta_t^2}{2}\|e_t\|^2 \quad (4.3) \end{aligned}$$

where the 2nd line follows if $\eta_t \leq 1/L$, and the 3rd line follows if f satisfies the PL inequality.

In Theorem 11, we will show that the sub-Gaussian assumption on e_t causes the second term to be sub-Gaussian with an implicit dependence on $f(x_t) - f^\star$ and the third term to be sub-exponential. In the following proposition, we show that such noise ends up causing $f(x_t) - f^\star$ to be sub-exponential, identifying $X_t \approx t^2(f(x_t) - f^\star)$. Below, the indices for all sequences are the natural numbers unless otherwise specified.

Proposition 1. *Let $(\Omega, \mathcal{F}, (\mathcal{F}_i), P)$ be a filtered probability space. Let (X_i), $(\hat{w}_i)$, and $(\hat{v}_i)$ be adapted to $(\mathcal{F}_i)$, and X_0 deterministic. Let (α_i), (β_i), and (γ_i) be non-negative sequences. Assume X_i and $\hat{v}_i$ are non-negative almost surely. Assume $\mathbb{E}[\exp(\lambda \hat{w}_i) \mid \mathcal{F}_{i-1}] \leq \exp(\frac{\lambda^2}{2}\beta_i^2 X_i)$ for all $\lambda \in \mathbb{R}$ and $\mathbb{E}[\exp(\lambda \hat{v}_i) \mid \mathcal{F}_{i-1}] \leq \exp(\lambda \gamma_i)$ for all $\lambda \in [0, \gamma_i^{-1}]$. Assume*

$$X_{i+1} \leq \alpha_i X_i + \hat{w}_i + \hat{v}_i.$$

Then, for any (K_i) *such that* $K_0 \geq X_0$ *and, for all* $i \geq 0$, $K_{i+1}^2 \geq (\alpha_i K_i + 2\gamma_i)K_{i+1} + \beta_i^2 K_i$, *and any* $n \geq 0$, *we have,* w.p. $\geq 1 - \delta$ *for all* $\delta \in (0, 1)$,

$$X_n \leq K_n \log(e/\delta).$$

The proof is in Appendix A.3. It is similar to the proof of Theorem 4.1 in [149] but with some key differences. There, the recursion is $X_{i+1} \leq \alpha_i X_i + \hat{w}_i \sqrt{X_i} + v_i$ where $\mathbb{E}\left[\exp(\lambda \hat{w}_i) \mid \mathcal{F}_{i-1}\right] \leq \exp\left(\frac{\lambda^2}{2}\beta_i^2\right)$ and v_i is deterministic. We had to move the implicit dependence of the sub-Gaussian term inside of the moment generating function in order to apply it to the inner product term in Eq. (4.3). We also had to allow v_i to be a sub-exponential random variable $\hat{v}_i$, and so applied Cauchy-Schwarz, which contributed the 2 in the recursion for (K_i).

With Proposition 1, we are ready to prove high probability convergence for SGD with a particular $\Theta(1/t)$ step-size sequence. The step-size is asymptotically equal to $2/\mu/t$. If the step-size is instead set to $\eta_t = c/t$ with $c > 2/\mu$ (e.g. if μ is unknown), then the convergence bound applies by resetting t to zero once $c/t \approx 2/\mu$.

Theorem 11. *Assume* f *is* L*-smooth and* μ*-PL and that, conditioned on the previous iterates,* e_t *is centered and* $\|e_t\|$ *is* σ*-sub-Gaussian. Then, SGD with step-size*

$$\eta_t = \frac{2t + 4\kappa - 1}{\mu(t + 2\kappa)^2},$$

where $\kappa = L/\mu$, *constructs a sequence* (x_t) *such that,* w.p. $\geq 1 - \delta$ *for all* $\delta \in (0, 1)$,

$$f(x_T) - f^\star = O\left(\frac{L\sigma^2 \log(1/\delta)}{\mu^2 T}\right).$$

The proof is in Appendix A.4. The convergence bound of Theorem 11 matches the optimal convergence rate in mean of $O(1/T)$ and has $\log(1/\delta)$ dependence on δ.

Remark 8. *It is natural to ask whether we can relax the sub-Gaussian assumption to a sub-Weibull assumption. In Proposition 1, we need a bound on the moment generating functions of both* $\hat{w}_i$ *and* $\hat{v}_i$. *But, if* $\|e_t\|$ *is* σ*-sub-Weibull*(θ) *with* $\theta > 1/2$, *then* $\|e_t\|^2$ *is* σ^2*-sub-Weibull*(2θ). *Thus,* $\hat{v}_i$ *would be sub-Weibull with tail parameter greater than 1, and so may have an infinite moment generating function for all* $\lambda > 0$.

4.3.2 Generalization

A particular example of Eq. (4.1) that we are concerned with is the supervised statistical learning problem [25, 170]. Consider a data set $S \coloneqq (s_i)_{i=1}^n \sim \mathcal{D}^n$ consisting of n iid samples, $s_i \sim \mathcal{D}$ for $i \in [n]$, where $\mathcal{D}$ is an arbitrary and unknown distribution. Let $\ell(x, s)$ measure the loss of the model parameterized by $x \in \mathbb{R}^d$ applied to the sample point s. Given S, let $F(x, i) = \ell(x, s_i)$ and define the **empirical risk**

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} \ell(x, s_i) = \mathbb{E}_{i\sim U([n])}[F(x, i)]. \tag{4.4}$$

Sometimes we write f_n (and $f_n^\star = \min_x f_n(x)$) when we want to emphasize the dependence of f on the samples, otherwise the n is implicit. We will apply SGD to minimizing the empirical risk using the mini-batch gradient estimator

$$g(x, b) = \frac{1}{b}\sum_{i\in I} \nabla\ell(x, s_i)$$

where I is a uniform sample of b indices sampled with replacement from $[n]$. In particular, we sample $g_t \sim g(x_t, b_t)$. However, the goal is actually to find an x that has a small **true risk**

$$\tilde{f}(x) = \mathbb{E}_{s\sim\mathcal{D}}\ \ell(x, s).$$

and we define $f_\infty^\star = \min_x \tilde{f}(x)$.

Applying SGD with many fewer than n/b iterations can be thought of as a stochastic approximation to minimize $\tilde{f}$ directly, but most realistic scenarios take more than one-pass of the data in order to further reduce the training error, so we adopt the statistical learning framework for analysis. In this framework, we really care about $\tilde{f}(x_T) - f_\infty^\star$, which we write as

$$\tilde{f}(x_T) - f_\infty^\star = \underbrace{\tilde{f}(x_T) - f_n(x_T)}_{\text{generalization error}} + \underbrace{f_n(x_T) - f_n^\star}_{\text{convergence error}} + \underbrace{f_n^\star - f_\infty^\star}_{\text{finite sample error}}.$$

Since the finite sample error depends only on the data and distribution $\mathcal{D}$ and not on the algorithm, we do not analyze it here; it is often assumed to be small, or at least dominated by the other terms. We will analyze the generalization error and convergence error separately. Unlike classical theory

which bounds the generalization error by restricting x to a low-complexity set, or by adding a regularization term, we use the implicit regularization of early stopping (Thm. 12), so generalization error is poly(T). Meanwhile, convergence error **decreases** with iterates, and is poly$(1/T)$ (Thm. 13); Theorem 14 shows that for $T = \Theta(n/b)$, both the poly(T) and poly$(1/T)$ terms decay to 0 as the number of samples $n \to \infty$. Proofs for this section can be found in Appendix A.6.

The foundations for the interplay between stability and generalization go back to [171]. Most recent results build off of [161]. In particular, the results of [162] can be applied to the mean stability bound of [163] to get a low probability generalization bound.

Theorem 12. *[162, 163] Assume $\ell(x, s) \in [0, M]$ for all x and s. Assume $\ell(\cdot, s)$ is ρ-Lipschitz and L-smooth for all s. Then, T iterations of SGD with $\eta_t \leq c/(t+1)$ and $b_t = b$ satisfies,* w.p. $\geq 1 - \delta$ *over S and (I_t) for all $\delta \in (0, 1)$,*

$$\tilde{f}(x_T) - f(x_T) \leq O\left(\sqrt{\frac{M\rho^2 T^{Lc}(c + 1/L)b}{n\delta}}\right).$$

Unfortunately the generalization bound in Thm. 12 depends on $1/\sqrt{\delta}$, which makes it a low probability bound. Furthermore, it grows with the number of iterations, counteracting the decay of the convergence error. Before proceeding, we need to prove a convergence rate for a more conservative step-size constant.

Theorem 13. *Assume f is L-smooth and μ-PL and that $\nabla f(x) - g(x, 1)$ is centered and $\sigma/\sqrt{d}$-sub-Gaussian for all x. Then, T iterations of SGD with $\eta_t = c/(t+1)$, where $c < 1/\mu$ and $c \leq 1/L$, and $b_t = b$ satisfies,* w.p. $\geq 1 - \delta$ *over (I_t) for all $\delta \in (0, 1)$,*

$$f(x_T) - f^\star \leq O\left(\left(f(x_0) - f^\star + \frac{Lc^2\sigma^2}{b}\right)\frac{\log(1/\delta)}{T^{\mu c}}\right).$$

Note that we assume $\nabla f(x) - g(x, 1)$ is $\sigma/\sqrt{d}$-sub-Gaussian (a random vector X is σ-sub-Gaussian if $\langle u, X\rangle$ is σ-sub-gaussian for all unit vectors u) rather than assuming $\|e_t\|$ is σ-sub-Gaussian. The former tightly implies the latter for a batch-size of one and, more generally, it allows us to show dependence on the batch-size in the results. While the convergence rate in Theorem 13 is much slower than that of Theorem 11, it allows us to balance convergence and generalization,

by choosing T to approximately minimize the bound, and obtain true risk bounds that go to the empirical minimum as $n \to \infty$. However, similarly to [145], one has to assume that μ is constant for all n. The bound depends on SGD taking a certain number of epochs.

Theorem 14. *Assume $\ell(x, s) \in [0, M]$ for all x and s. Assume $\ell(\cdot, s)$ is ρ-Lipschitz and L-smooth for all s. Assume f is μ-PL. Let $\kappa = L/\mu$. Assume $\nabla f(x) - g(x, 1)$ is centered and $\sigma/\sqrt{d}$-sub-Gaussian for all x. Let $b_t = b$, $c = 1/(\mu + L)$, and $T = \Theta(n/b)$. Then, T iterations of SGD with $\eta_t = c/(t + 1)$ satisfies,* w.p. $\geq 1 - \delta$ *over S and (I_t) for all $\delta \in (0, 1/e)$,*

$$\tilde{f}(x_T) - f_n^\star \leq O\left(\frac{b^{1/(2\kappa+2)}}{n^{1/(2\kappa+2)}\sqrt{\delta}} + \frac{\log(1/\delta)}{b^{1-1/(\kappa+1)}n^{1/(\kappa+1)}}\right).$$

Recall that $\tilde{f}(x_T) - f_n^\star$ is the sum of the generalization and convergence errors. The proof of Theorem 14 follows from plugging c and T into Theorems 13 and 12. Note that the right-hand side goes to zero as $n \to \infty$, under the assumption that μ does not change. The form of the bound is expected for a (mini-batch) SGD: the main message is that one should take multiple passes over the data to converge sufficiently, but then stop early to avoid over-fitting. Also note that while Theorem 14 provides a low probability bound proportional to $1/\sqrt{\delta}$ it is still better than the $1/\delta$ dependence we would get without the high probability convergence.

4.4 Smooth, Non-convex, Sub-Weibull Setting

In this section, we consider norm sub-Weibull noise and a general possibly non-convex cost. We seek to find an approximate stationary point, meaning a point x where $\|\nabla f(x)\|^2$ is small. First, we prove a Freedman-type inequality in Proposition 2. There end up being three different regimes corresponding to the heaviness of the tails, namely, sub-Gaussian tails, heavier than sub-Gaussian but only up to sub-exponential tails, and heavier than sub-exponential tails.

Next, we apply Proposition 2 to the error of non-convex SGD. We are able to prove a convergence bound for all three regimes of sub-Weibull gradient noise. The convergence bounds are in terms of the error $\min_{0 \leq t \leq T-1} \|\nabla f(x_t)\|^2$. Thus, a post-processing step is still required to get

a single iterate. We provide a method for randomly picking a subset of iterates to minimize over, instead of having to minimize over all T iterates.

4.4.1 Sub-Weibull Freedman Inequality

The following proposition provides a result for the concentration of the sum of a sub-Weibull martingale difference sequence.

Proposition 2. *Let $(\Omega, \mathcal{F}, (\mathcal{F}_i), P)$ be a filtered probability space. Let (ξ_i) and (K_i) be adapted to $(\mathcal{F}_i)$. Let $n \in \mathbb{N}$, then for all $i \in [n]$, assume $K_{i-1} \geq 0$, $\mathbb{E}[\xi_i \mid \mathcal{F}_{i-1}] = 0$, and*

$$\mathbb{E}\left[\exp\left((|\xi_i|/K_{i-1})^{1/\theta}\right) \mid \mathcal{F}_{i-1}\right] \leq 2$$

where $\theta \geq 1/2$. If $\theta > 1/2$, assume there exists (m_i) such that $K_{i-1} \leq m_i$.

If $\theta = 1/2$, then for all $x, \beta \geq 0$, and $\alpha > 0$, and $\lambda \in \left[0, \frac{1}{2\alpha}\right]$,

$$P\left(\bigcup_{k\in[n]}\left\{\sum_{i=1}^{k}\xi_i \geq x \text{ and } 2\sum_{i=1}^{k}K_{i-1}^2 \leq \alpha\sum_{i=1}^{k}\xi_i + \beta\right\}\right) \leq \exp(-\lambda x + 2\lambda^2\beta), \tag{4.5}$$

and for all $x, \beta, \lambda \geq 0$,

$$P\left(\bigcup_{k\in[n]}\left\{\sum_{i=1}^{k}\xi_i \geq x \text{ and } 2\sum_{i=1}^{k}K_{i-1}^2 \leq \beta\right\}\right) \leq \exp\left(-\lambda x + \frac{\lambda^2}{2}\beta\right). \tag{4.6}$$

If $\theta \in \left(\frac{1}{2}, 1\right]$, let

$$a = (4\theta)^{2\theta}e^2$$

$$b = (4\theta)^{\theta}e.$$

For all $x, \beta \geq 0$, and $\alpha \geq b\max_{i\in[n]} m_i$, and $\lambda \in \left[0, \frac{1}{2\alpha}\right]$,

$$P\left(\bigcup_{k\in[n]}\left\{\sum_{i=1}^{k}\xi_i \geq x \text{ and } \sum_{i=1}^{k}aK_{i-1}^2 \leq \alpha\sum_{i=1}^{k}\xi_i + \beta\right\}\right) \leq \exp(-\lambda x + 2\lambda^2\beta), \tag{4.7}$$

and for all $x, \beta \geq 0$, and $\lambda \in \left[0, \frac{1}{b\max_{i\in[n]} m_i}\right]$,

$$P\left(\bigcup_{k\in[n]}\left\{\sum_{i=1}^{k}\xi_i \geq x \text{ and } \sum_{i=1}^{k}aK_{i-1}^2 \leq \beta\right\}\right) \leq \exp\left(-\lambda x + \frac{\lambda^2}{2}\beta\right). \tag{4.8}$$

If $\theta > 1$, let $\delta \in (0,1)$. Let

$$a = (2^{2\theta+1}+2)\Gamma(2\theta+1) + \frac{2^{3\theta}\Gamma(3\theta+1)}{3}$$

$$b = 2\log(n/\delta)^{\theta-1}.$$

For all $x, \beta \geq 0$, and $\alpha \geq b\max_{i\in[n]} m_i$, and $\lambda \in \left[0, \frac{1}{2\alpha}\right]$,

$$P\left(\bigcup_{k\in[n]}\left\{\sum_{i=1}^{k}\xi_i \geq x \text{ and } \sum_{i=1}^{k} aK_{i-1}^2 \leq \alpha\sum_{i=1}^{k}\xi_i + \beta\right\}\right) \leq \exp(-\lambda x + 2\lambda^2\beta) + 2\delta, \tag{4.9}$$

and for all $x, \beta \geq 0$, and $\lambda \in \left[0, \frac{1}{b\max_{i\in[n]} m_i}\right]$,

$$P\left(\bigcup_{k\in[n]}\left\{\sum_{i=1}^{k}\xi_i \geq x \text{ and } \sum_{i=1}^{k} aK_{i-1}^2 \leq \beta\right\}\right) \leq \exp\left(-\lambda x + \frac{\lambda^2}{2}\beta\right) + 2\delta. \tag{4.10}$$

For $\theta = 1/2$ without α (i.e., $\alpha = 0$), Eq. (4.6), we recover the classical Freedman's inequality [169]. For $\theta \in (1/2, 1]$ without α, Eq. (4.8), we recover Thm. 2.6 of [140]. For $\theta = 1/2$ with α, Eq. (4.5), we recover Thm. 3.3 of [149], called the "Generalized Freedman" inequality. For $\theta \in (1/2, 1]$ with α, Eq. (4.7), we extend Thm. 2.6 of [140] to include α on the one hand, and we extend Thm. 3.3 of [149] to $\theta \in (1/2, 1]$ on the other hand. For $\theta > 1$ without α, Eq. (4.10), we extend Thm. 2.6 of [140] to include α. Finally, we also provide Eq. (4.9) for the case $\theta > 1$ with α.

The proof is in Appendix A.8. For $\theta \in (1/2, 1]$ with α, the proof is a simple extension of the proof of Thm. 3.3 of [149]. For $\theta > 1$ without α, the proof is a more technical extension of the proof of Cor. 2 of [141] from i.i.d. random variables to martingale difference sequences. For $\theta > 1$ with α, the proof combines the previous two results.

4.4.2 Convergence

Rearranging and summing Eq. (4.2), we get

$$\sum_{t=0}^{T-1}\eta_t\left(1 - \frac{L\eta_t}{2}\right)\|\nabla f(x_t)\|^2 \leq f(x_0) - f(x_T) + \sum_{t=0}^{T-1}\eta_t(1-L\eta_t)\langle\nabla f(x_t), e_t\rangle + \frac{L}{2}\sum_{t=0}^{T-1}\eta_t^2\|e_t\|^2,$$

and using $f(x_T) \geq f^\star$, we get

$$\leq f(x_0) - f^\star + \sum_{t=0}^{T-1}\eta_t(1-L\eta_t)\langle\nabla f(x_t), e_t\rangle + \frac{L}{2}\sum_{t=0}^{T-1}\eta_t^2\|e_t\|^2.$$

Then, applying Proposition 2 proves the following theorem.

Theorem 15 (Non-convex case, simplified version)**.** *Assume f is L-smooth and that, conditioned on the previous iterates, e_t is centered and $\|e_t\|$ is K-sub-Weibull(θ) with $\theta \geq \frac{1}{2}$. If $\theta > \frac{1}{2}$, assume f is ρ-Lipschitz. Let $\delta \in (0,1)$ be a failure probability. Define*

$$c^\star = \frac{\sqrt{f(x_0) - f^\star}}{K\left(4e\theta \log(2/\delta)\right)^\theta \sqrt{L \log(T+1)}}.$$

Then, for T iterations of SGD with stepsize $\eta_t = c/\sqrt{t+1}$ where $c = c^\star$ and $c^\star \leq 1/L$, we have with probability at least $1-\delta$:

$$\frac{1}{\sqrt{T}} \sum_{t=0}^{T-1} \frac{1}{\sqrt{t+1}} \|\nabla f(x_t)\|^2 \leq O\left(\frac{\sqrt{\log(T)}\log(1/\delta)^\theta + \log(T/\delta)^{\max\{0,\theta-1\}} \log(1/\delta)}{\sqrt{T}} \right).$$

The full theorem, Thm. 15, is in the appendix, and covers the case when $c^\star \geq 1/L$, which essentially slightly worsens the numerator in the convergence rate, e.g., for $\theta = \frac{1}{2}$, it worsens from $\sqrt{\log(T)\log(1/\delta)}$ to $\log(T)\log(1/\delta)$. The condition $c^\star \leq 1/L$ is mild in the sense that it is guaranteed to occur in any of the following limiting cases: (1) noise variance increases, $K \to \infty$, (2) we start close to the solution, $x_0 \to x^\star$, (3) we want a high-probability bound, $\delta \to 0$, and/or (4) we take many iterations, $T \to \infty$.

The proof is in Appendix A.9. Note that the convergence rate of $O(1/\sqrt{T})$ is preserved, up to log factors, for all values of θ. It is natural to ask whether we can allow for even heavier-tailed noise beyond sub-Weibull. For sub-Weibull, we get the $\log(T/\delta)^{\theta-1}$ term from the assumed lower bound on α in Proposition 2. For heavier-tailed noise, the assumed lower bound on α would be even larger, and so the convergence rate would no longer be $1/\sqrt{T}$ up to log factors.

Note that the results of Thm. 15 are in terms of $\frac{1}{\sqrt{T}} \sum_{t=0}^{T-1} \frac{1}{\sqrt{t+1}} \|\nabla f(x_t)\|^2$ which is not a particularly useful quantity by itself. The standard trick is to observe

$$\min_{0 \leq t \leq T-1} \|\nabla f(x_t)\|^2 \leq \frac{1}{\sqrt{T}} \sum_{t=0}^{T-1} \frac{1}{\sqrt{t+1}} \|\nabla f(x_t)\|^2 \tag{4.11}$$

so one could keep track of $\|\nabla f(x_t)\|$ at every iteration and record the iterate where this is lowest. However, this requires exact gradient information, which may be more costly than the stochastic

gradient used in the algorithm, so we desire a more clever post-processing scheme. In [135], they pick index s with probability proportional to $1/\sqrt{s+1}$ so that $\mathbb{E}\left[\|\nabla f(x_s)\|^2\right]$ is proportional to the right-hand side of Eq. (4.11). They do this for $\Theta(\log(1/\delta))$ *runs* and pick the best of the runs.

We, on the other hand, sample a set S of $n_{\text{iter}} = \Theta(\log(1/\delta))$ indices and pick the best *iterate* from among these samples. Basically, while Eq. (4.11) dictates that we should output the best iterate

$$x = \underset{0 \le t \le T-1}{\operatorname{argmin}} \|\nabla f(x_t)\|^2, \tag{4.12}$$

the following Theorem allows us to output the best sampled iterate

$$x = \underset{t \in S}{\operatorname{argmin}} \|\nabla f(x_t)\|^2 \tag{4.13}$$

with slightly worse convergence error bounds.

Theorem 16 (Efficient post-processing)**.** *Under the conditions of Theorem 15, if we pick a sample set, S, of up to n of the indices $s \in \{0, \ldots, T-1\}$, choosing s with probability proportional to $\frac{1}{\sqrt{s+1}}$ independently with replacement, then,*

$$P\left(\min_{t \in S} \|\nabla f(x_t)\|^2 > e\zeta\right) \le \exp(-n) + P\left(\frac{1}{\sqrt{T}} \sum_{t=0}^{T-1} \frac{1}{\sqrt{t+1}} \|\nabla f(x_t)\|^2 > \zeta\right) \ \forall \zeta > 0.$$

The proof is in Appendix A.9. To combine with Thm. 15, set

$$n_{\text{iter}} = \lceil \log(1/\delta_{\text{iter}}) \rceil$$

where δ_{iter} is the iterate sampling failure probability.

The final issue is that in order to compute even Eq. (4.13) we need full gradient information. In the sample average approximation setting, this can be obtained by running on the full batch of data (rather than a mini-batch). However, if this is computationally infeasible or if we are in the stochastic approximation setting, then we instead have to use empirical gradients over a test or validation set. Fortunately, a straight-forward application of Thm. 2.4 of [135] shows that using the empirical gradients over a single test or validation set with sufficient samples only slightly worsens

the convergence error bound. In order to apply Thm. 2.4 of [135] we have to further specify the stochastic setting.

Suppose $\nabla f(x) = \mathbb{E}\left[G(x,\xi)\right]$ for all $x \in \mathbb{R}^d$, where $G(x,\cdot)$ is the stochastic gradient at x. Assume $\|\nabla f(x) - G(x,\xi)\|$ is K-sub-Weibull(θ). At each iteration, sample ξ_t and set $g_t = G(x,\xi_t)$. Thus, conditioned on the previous iterates, e_t is centered and $\|e_t\|$ is K-sub-Weibull(θ). So, Thm. 15 and Thm. 16 apply.

Now we follow the second post-processing step of the 2-RSG method of [135]. First, we sample $\xi_1, \ldots, \xi_{n_{\text{emp}}}$ independently. Second, we compute

$$x = \underset{t\in S}{\operatorname{argmin}} \left\| \frac{1}{n_{\text{emp}}} \sum_{j=0}^{n_{\text{emp}}} G(x_t, \xi_j) \right\|^2. \tag{4.14}$$

By Lemma 5, we have

$$\mathbb{E}\left[\|\nabla f(x) - G(x,\xi)\|^2\right] \leq 2\Gamma(2\theta+1)K^2.$$

So, we can apply Thm. 2.4 of [135] to get

$$P\left(\|\nabla f(x)\|^2 > 5e\zeta\right) \leq P\left(\min_{t\in S} \|\nabla f(x_t)\|^2 > e\zeta\right) + \frac{12\Gamma(2\theta+1)(n_{\text{iter}}+1)K^2}{e\zeta n_{\text{emp}}} \ \forall \zeta, \lambda > 0.$$

So, altogether, if

$$n_{\text{emp}} = \left\lceil \frac{12(\lceil \log(1/\delta_{\text{iter}}) \rceil + 1)\Gamma(2\theta+1)K^2}{e\zeta\delta_{\text{emp}}} \right\rceil,$$

where δ_{emp} is the empirical gradient failure probability, then

$$P\left(\|\nabla f(x)\|^2 > 5e\zeta\right) \leq P\left(\frac{1}{\sqrt{T}} \sum_{t=0}^{T-1} \frac{1}{\sqrt{t+1}} \|\nabla f(x_t)\|^2 > \zeta\right) + \delta_{\text{iter}} + \delta_{\text{emp}} \ \forall \zeta > 0.$$

4.5 Examples in Neural Networks

In this section, we will study synthetic neural net problems to provide insight for the theoretical results of this chapter. In Section 4.5.1, we consider a neural net (similar to ResNet) that comes from [127]. We provide numerical evidence that the true risk and misclassification have high probability dependence on the failure probability for this PL-type problem. This is actually better than what

we proved, which was only $(1/\sqrt{\delta})$-dependence on the failure probability δ. In Section 4.5.2, we consider a neural net with two layers of ReLU. We provide numerical evidence that a *single pass* of SGD (with post-processing) applied to this non-convex stochastic approximation problem results in convergence error with high probability dependence on the failure probability. We compare to the last iterate of SGD and find that the last iterate actually performs slightly better. In Section 4.5.3, we modify the previous experiment to be a sample average approximation problem so that we can compare with the best iterate as well. We find that the best iterate performs slightly better than the last iterate.

To determine the probability dependence, we run over n_{tri} trials and output a vector, y, of the trial results (that is, the appropriate error metric). Then we compute a vector x as $1 - (0, 1, \ldots, n_{\text{tri}} - 1)/n_{\text{tri}}$ and plot x on the x-axis, y on the y-axis. By doing so, the y-coordinate of a data point corresponds to the $(1 - \delta)$-percentile result for δ equal to the x-coordinate. Thus, this approximates the dependence on the failure probability. However, the uncertainty increases as we move away from $\delta = 0.5$ towards $\delta = 1/n_{\text{tri}}$ since we are doing less averaging.

Our code can be found at `https://github.com/liammadden/sgd`.

4.5.1 PL Generalization

Let $\phi(x) = \max\{0, x\}$ element-wise denote the ReLU function. As we are interested in training, our objective will be a function of the weights $A \in \mathbb{R}^{d\times d}$, parameterized by features $x \in \mathbb{R}^d$ and labels $y \in \mathbb{R}$:

$$\forall (x, y) \in \mathbb{R}^d \times \mathbb{R},$$
$$F(\cdot\,; x, y) : \mathbb{R}^{d\times d} \to \mathbb{R} : (A; x, y) \mapsto \frac{1}{2}\left(h(A; x) - y\right)^2 \quad \text{where} \quad h(A; x) = \|\phi(Ax + x)\|_1$$

which comes from [127] and satisfies "one-point strong convexity" in a ball around the unique minimizer, which in turn implies the PL inequality in a ball.

By defining the vector $D = D(A; x) = \mathbb{I}_{Ax+x\geq 0}$ (where $\mathbb{I}$ denotes the 1-0 indicator function),

it is not difficult to derive

$$\frac{\partial}{\partial A}F(A;x,y) = (\|\phi(Ax+x)\|_1 - y)\, Dx^\top.$$

Note that F is a *regression* loss function. If our goal is *classification*, then it is a surrogate for the misclassification rate. Thus, while we apply SGD to the regression loss, we also compute the misclassification rate of the iterates where the label we associate with y is $\operatorname{sign}(y-a)$ for some cut-off a chosen **a priori**.

Consider independent samples $x_i \sim U(S^{d-1})$, where $U(S^{d-1})$ denotes the uniform distribution over the unit sphere in $\mathbb{R}^d$. Training is done to minimize the empirical risk

$$f(A;X,Y) = \frac{1}{n}\sum_{i=1}^{n} F(A;x_i,y_i).$$

whereas our real goal is to minimize the true risk

$$\tilde{f}(A) = \mathbb{E}_{x\sim U(S^{d-1})}[F(A;x,h(A^\star;x))].$$

We will use samples $y_i = h(A^\star;x_i)$ for some fixed $A^\star$, hence the minimum of the empirical risk, as well as the true risk, is zero.

Consider dimension $d = 100$ and misclassification cut-off $a = 5$, and sample $A^\star_{ij} \sim N(0,1/d)$ and $(A_0)_{ij} \sim N(0,1/d)$ (where A_0 is the initial iterate for SGD) in that order with numpy random seed 17. We use $n = 250$ samples, step-size $\eta_t = 0.1/(t+1)$, batch-size $b = 1$, $bT/n = 10$ epochs, and $n_{\text{tri}} = 1000$ trials. Each trial uses new random data $X \in \mathbb{R}^{n\times d}$ and new random minibatch selection.

Figure 4.1 shows the dependence of the true risk and misclassification on the failure probability. For reference, we have also plotted the least-squares best fit of the data based on a $\log(1/\delta)$, $1/\delta$ and $1/\sqrt{\delta}$ form, allowing for an offset. We ignore data for $\delta > 10^{-1}$ since we are interested in the limit $\delta \to 0$. However, the uncertainty does become greater as δ approaches $1/n_{\text{tri}} = 10^{-3}$. For the true risk, the data are most consistent with a $\log(1/\delta)$ functional form, but not entirely inconsistent with $1/\sqrt{\delta}$. They do not appear to have a $1/\delta$ relationship. Our bound, which is of the $1/\sqrt{\delta}$ type,

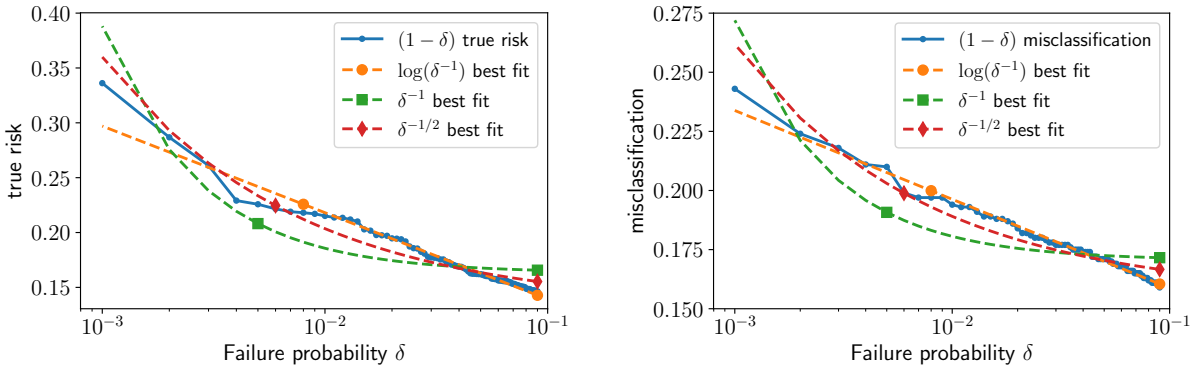


Figure 4.1: Dependence of the true risk and misclassification on the failure probability δ, for a fixed number of samples and epochs. Our theory bounds the true risk by $O(1/\sqrt{\delta})$.

explains data much better than the naive bound using $1/\delta$. However, it is possible that stronger assumptions on the distribution could be used to improve the theoretical bound to reflect $\log(1/\delta)$ dependence. For the misclassification error, for which our experiment has no theoretical guarantees (since we trained with the surrogate loss), again we see that $\log(1/\delta)$ is most consistent with the data.

4.5.2 Non-convex Stochastic Approximation Convergence

Again let ϕ denote the ReLU function and consider

$$\forall x \in \mathbb{R}^{d_1}, \quad h(\cdot\,; x) : \mathbb{R}^{m\times d_1} \times \mathbb{R}^{m\times m} \times \mathbb{R}^{d_2\times m} \to \mathbb{R}^{d_2} : (A, B, C; x) \mapsto C\phi(B\phi(Ax))$$

and

$$\forall (x, y) \in \mathbb{R}^{d_1} \times \mathbb{R}^{d_2},$$
$$F(\cdot\,; x, y) : \mathbb{R}^{m\times d_1} \times \mathbb{R}^{m\times m} \times \mathbb{R}^{d_2\times m} \to \mathbb{R} : (A, B, C; x, y) \mapsto \frac{1}{2}\|h(A, B, C; x) - y\|_2^2$$

By defining the matrices $D_1 = D_1(A, x) = \text{diag}\mathbb{I}_{Ax\geq 0}$ and $D_2 = D_2(A, B, x) = \text{diag}\mathbb{I}_{B\phi(Ax)\geq 0}$ we can rewrite h and F with matrices instead of ReLUs so that we can take the derivative. The gradients can be determined via automatic differentiation software like PyTorch, or, using Eq. (119)

of [172], we can work out the gradients by hand as follows:

$$\frac{\partial}{\partial A}F(A,B,C;x,y) = (CD_2BD_1)^\top(CD_2BD_1Ax - y)x^\top$$
$$\frac{\partial}{\partial B}F(A,B,C;x,y) = (CD_2)^\top(CD_2BD_1Ax - y)(D_1Ax)^\top$$
$$\frac{\partial}{\partial C}F(A,B,C;x,y) = (CD_2BD_1Ax - y)(D_2BD_1Ax)^\top.$$

Note that $A = B = C = 0$ is a stationary point, so we should not initialize at zero.

Consider the stochastic approximation cost function

$$f(A,B,C) = \mathbb{E}_{x\sim U(S^{d_1-1})}[F(A,B,C;x,h^\star(A^\star,B^\star,C^\star;x))],$$

where the inner dimension of $h^\star$ is $m^\star$.

We will apply SGD and compute the post-processing iterate via Eq. (4.14) with n_{iter} test iterates and a test set of n_{test} samples. We will compute the convergence error as the norm-squared of the empirical gradient over a validation set of n_{val} samples. We will compare the convergence error of both the post-processing iterate and the last iterate. We expect the convergence error of the post-processing iterate to have a linear dependence on $\log(1/\delta)$ as long as n_{iter}, n_{test}, and n_{val} are sufficiently large.

Consider input dimension $d_1 = 10$, output dimension $d_2 = 1$, true inner dimension $m^\star = 20$, and model inner dimension $m = 10$. Sample $A^\star_{ij} \sim N(0, 2/m^\star)$, $B^\star_{ij} \sim N(0, 2/m^\star)$, $C^\star_{ij} \sim N(0, 1/d_2)$, $(A_0)_{ij} \sim N(0, 2/m)$, $(B_0)_{ij} \sim N(0, 2/m)$, and $(C_0)_{ij} \sim N(0, 1/d_2)$ in that order with numpy random seed 17. We use step-size $\eta_t = 0.1/\sqrt{t+1}$, $T = 1000$ iterations, batch-size $b = 10$, and $n_{\text{tri}} = 100$ trials. Finally, we set $n_{\text{iter}} = 200$, $n_{\text{test}} = 1000$, and $n_{\text{val}} = 5000$.

Figure 4.2 shows the $(1-\delta)$-percentile convergence error over the validation set, and the right plot shows a least-squares best fit line to the form $\log(1/\delta)$ (with an offset). As expected, the convergence error of the post-processing iterate depends linearly on $\log(1/\delta)$. The last iterate actually performs slightly better than the post-processing iterate even though it does not have the same theoretical guarantees; this justifies the standard practice of just using the last iterate.

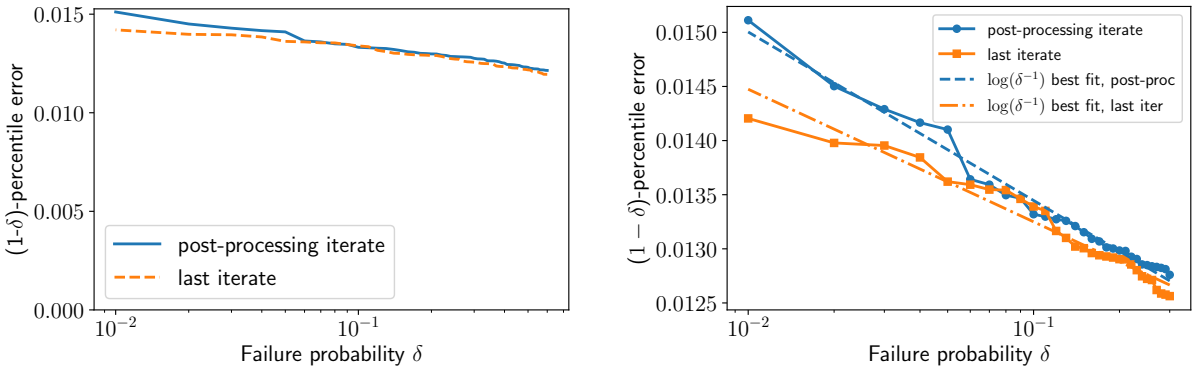


Figure 4.2: Top: Dependence of the convergence error on the failure probability. Bottom: Zoom, and line of best fit to $\log(1/\delta)$ form (allowing for an offset).

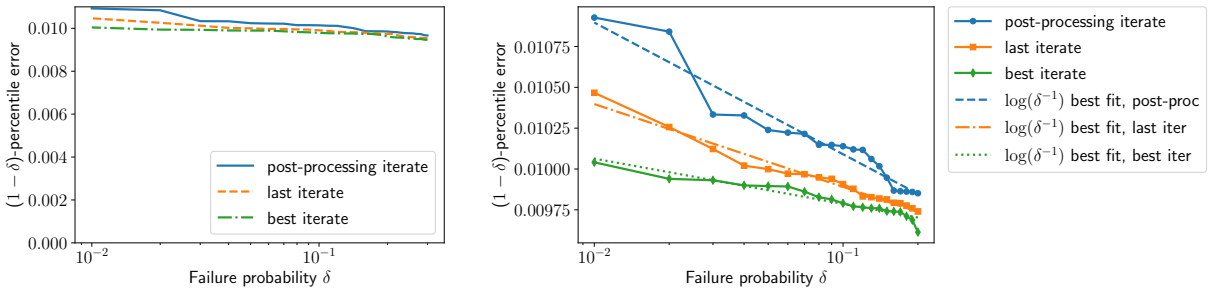


Figure 4.3: Top: Dependence of the convergence error on the failure probability. Bottom: Zoom, and line of best fit to $\log(1/\delta)$ form (allowing for an offset).

4.5.3 Non-convex Sample Average Approximation Convergence

Here we modify the previous experiment by considering the sample average approximation cost function

$$f(A, B, C) = \frac{1}{n} \sum_{i=1}^{n} F(A, B, C; x_i, h^\star(A^\star, B^\star, C^\star; x_i)).$$

Now, we actually do have access to the full gradient information for comparison if we are willing to pay the computational cost for a full batch gradient. This allows us to compare with the best iterate, computed via Eq. (4.12). Moreover, we can compute the convergence error as the norm-squared of the true gradient.

Again, consider input dimension $d_1 = 10$, output dimension $d_2 = 1$, true inner dimension $m^\star = 20$, and model inner dimension $m = 10$. Sample $A^\star_{ij} \sim N(0, 2/m^\star)$, $B^\star_{ij} \sim N(0, 2/m^\star)$, $C^\star_{ij} \sim N(0, 1/d_2)$, $(A_0)_{ij} \sim N(0, 2/m)$, $(B_0)_{ij} \sim N(0, 2/m)$, $(C_0)_{ij} \sim N(0, 1/d_2)$, $X_{ij} \sim N(0, 1)$ in that order with numpy random seed 17. Then normalize the rows of X and compute Y. Again, we use step-size $\eta_t = 0.1/\sqrt{t+1}$, $T = 1000$ iterations, batch-size $b = 10$, $n_{\text{tri}} = 100$ trials, $n_{\text{iter}} = 200$, and $n_{\text{test}} = 1000$.

Figure 4.3 shows the $(1-\delta)$-percentile convergence error. The best iterate performs slightly better than the last iterate, and the last iterate performs slightly better than the post-processing iterate. The differences are slight, and all have a roughly $O(\log(1/\delta))$ relationship. Overall, this

corroborates the theory: the iterate sampling scheme is slightly sub-optimal and scales similarly (in δ) to the optimal iterate.

4.6 Conclusions

This chapter analyzed the convergence of SGD for objective functions that satisfy the PL condition and for generic non-convex objectives. Under a sub-Gaussian assumption on the gradient error, we showed a high probability convergence rate matching the mean convergence rate for PL objectives. Our convergence result was then utilized in the context of statistical learning to support the practice of taking multiple but limited epochs of mini-batch SGD. Under a sub-Weibull assumption on the noise, we showed a high probability convergence rate matching the mean convergence rate for non-convex objectives. We also provided and analyzed a post-processing method for choosing a single iterate. To prove the convergence rate, we first proved a Freedman-type inequality for martingale difference sequences that extends previous Freedman-type inequalities beyond the sub-exponential threshold to allow for sub-Weibull tail-decay. Finally, we considered two synthetic neural net problems, one with a PL objective, and one with a non-convex objective. For the PL objective, we showed that the true risk and misclassification have high probability dependence on the failure probability. We leave improving our true risk bound to high probability and extending to classification as future work. For the non-convex objective, we showed that the convergence error has high probability dependence on the failure probability whether we choose the best iterate, the post-processing iterate, or the last iterate.

Bibliography

[1] Zulehner, A., Paler, A., Wille, R.: (2019). URL `https://github.com/iic-jku/ibm_qx_mapping`

[2] Khatri, S., LaRose, R., Poremba, A., Cincio, L., Sornborger, A.T., Coles, P.J.: Quantum-assisted quantum compiling. Quantum **3**(1), 140 (2019)

[3] Nemirovski, A., Yudin, D.: Problem complexity and method efficiency in optimization. Wiley-Interscience (1983)

[4] Dall'Anese, E., Simonetto, A., Becker, S., Madden, L.: Optimization and learning with information streams: Time-varying algorithms and applications. IEEE Signal Processing Magazine **37**(3), 71–83 (2020)

[5] Dall'Anese, E., Simonetto, A.: Optimal power flow pursuit. IEEE Trans. on Smart Grid **9**, 942–952 (2018)

[6] Tang, Y., Dvijotham, K., Low, S.: Real-time optimal power flow. IEEE Trans. on Smart Grid **8**, 2963–2973 (2017)

[7] Bianchin, G., Pasqualetti, F.: A network optimization framework for the analysis and control of traffic dynamics and intersection signaling. In: IEEE Conference on Decision and Control, pp. 1017–1022 (2018)

[8] Chen, J., Lau, V.K.N.: Convergence analysis of saddle point problems in time varying wireless systems: Control theoretical approach. IEEE Trans. on Signal Processing **60**(1), 443–452 (2012)

[9] Bernstein, A., Dall'Anese, E., Simonetto, A.: Online primal-dual methods with measurement feedback for time-varying convex optimization. IEEE Trans. on Signal Processing **67**(8), 1978–1991 (2019)

[10] Simonetto, A.: Time-varying convex optimization via time-varying averaged operators. arXiv preprint arXiv:1704.07338 (2017)

[11] Yang, T.: Accelerated gradient descent and variational regret bounds (2016)

[12] Beck, A.: First-Order Methods in Optimization. MOS-SIAM Series on Optimization (2017)

[13] Bauschke, H., Combettes, P.: Convex Analysis and Monotone Operator Theory in Hilbert Spaces, 2 edn. Springer-Verlag (2017)

[14] Nesterov, Y.: Introductory Lectures on Convex Optimization: A Basic Course. Kluwer (2004)

[15] Nesterov, Y.: Lectures on Convex Optimization, 2 edn. Springer (2018)

[16] Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press (2004)

[17] Bubeck, S.: Convex Optimization: Algorithms and Complexity, Foundations and Trends in Machine Learning, vol. 8. now (2015)

[18] Popkov, A.Y.: Gradient methods for nonstationary unconstrained optimization problems. Automation and Remote Control **66**(6), 883–891 (2005)

[19] Simonetto, A., Leus, G.: Double smoothing for time-varying distributed multiuser optimization. In: IEEE Global Conference on Signal and Information Processing, pp. 852–856 (2014)

[20] Mokhtari, A., Shahrampour, S., Jadbabaie, A., Ribeiro, A.: Online optimization in dynamic environments: Improved regret rates for strongly convex problems. In: IEEE 55th Conference on Decision and Control (CDC), pp. 7195–7201 (2016)

[21] Hall, E.C., Willett, R.M.: Online convex optimization in dynamic environments. IEEE Journal of Selected Topics in Signal Processing **9**(4), 647–662 (2015)

[22] Jadbabaie, A., Rakhlin, A., Shahrampour, S., Sridharan, K.: Online optimization: Competing with dynamic comparators. In: International Conference on Artificial Intelligence and Statistics (AISTATS), vol. 38, pp. 398 – 406 (2015)

[23] Besbes, O., Gur, Y., Zeevi, A.: Non-stationary stochastic optimization. Operations research **63**(5), 1227–1244 (2015)

[24] Yang, T., Zhang, L., Jin, R., Yi, J.: Tracking slowly moving clairvoyant: Optimal dynamic regret of online learning with true and noisy gradient. In: International Conference on Machine Learning (ICML), vol. 48, p. 449–457 (2016)

[25] Shalev-Shwartz, S., Ben-David, S.: Understanding machine learning: From theory to algorithms. Cambridge university press (2014)

[26] Allen-Zhu, Z., Hazan, E.: Optimal black-box reductions between optimization objectives. In: Neural Information Processing Systems (NeurIPS), vol. 29 (2016)

[27] Nocedal, J., Wright, S.: Numerical Optimization, 2 edn. Operations Research and Financial Engineering. Springer (2006)

[28] Drori, Y., Teboulle, M.: Performance of first-order methods for smooth convex minimization: a novel approach. Mathematical Programming **145**, 451–482 (2012)

[29] Kim, D., Fessler, J.: Optimized first-order methods for smooth convex minimization. Mathematical Programming **159**, 81–107 (2016)

[30] Taylor, A., Hendrickx, J., Glineur, F.: Smooth strongly convex interpolation and exact worst-case performance of first-order methods. Mathematical Programming **161**, 307–345 (2016)

[31] O'Donoghue, B., Candès, E.: Adaptive restart for accelerated gradient schemes. Foundations of Computational Mathematics **15**, 715–732 (2015)

[32] Allen-Zhu, Z., Orecchia, L.: Linear coupling: An ultimate unification of gradient and mirror descent. In: Innovations in Theoretical Computer Science (ITCS) (2017)

[33] Schmidt, M., Roux, N., Bach, F.: Convergence rates of inexact proximal-gradient methods for convex optimization. In: Neural Information Processing Systems (NeurIPS), vol. 24 (2011)

[34] Devolder, O., Glineur, F., Nesterov, Y.: First-order methods of smooth convex optimization with inexact oracle. Mathematical Programming (2014)

[35] Bertsekas, D., Tsitsiklis, J.: Gradient convergence in gradient methods with errors. SIAM Journal on Optimization **10**(3), 627–642 (2000)

[36] Villa, S., Salzo, S., Baldassarre, L., Verri, A.: Accelerated and inexact forward-backward algorithms. SIAM Journal on Optimization **23**(3), 1607–1633 (2013)

[37] Aujol, J., Dossal, C.: Stability of over-relaxations for the forward-backward algorithm, application to fista. SIAM Journal on Optimization **25**(4), 2408–2433 (2015)

[38] Tseng, P.: Approximation accuracy, gradient methods, and error bounds for structured convex optimization. Mathematical Programming **125**, 263–295 (2010)

[39] Polyak, B.: Some methods of speeding up the convergence of iteration methods. USSR Computational Mathematics and Mathematical Physics (1964)

[40] Hazan, E., Agarwal, A., Kale, S.: Logarithmic regret algorithms for online convex optimization. Machine Learning **69**, 169–192 (2007)

[41] Shalev-Shwartz, S.: Online Learning and Online Convex Optimization, Foundations and Trends in Machine Learning, vol. 4. now, Hanover (2012)

[42] Madden, L., Becker, S., Dall'Anese, E.: Online sparse subspace clustering. In: 2019 IEEE Data Science Workshop (DSW), pp. 248–252 (2019)

[43] Dixit, R., Bedi, A.S., Tripathi, R., Rajawat, K.: Online learning with inexact proximal online gradient descent algorithms. IEEE Trans. on Signal Processing **67**(5), 1338–1352 (2019)

[44] Ryu, E., Boyd, S.: A primer on monotone operator methods. Applied Computational Mathematics **15**(1), 3–43 (2016)

[45] Lessard, L., Recht, B., Packard, A.: Analysis and design of optimization algorithms via integral quadratic constraints. SIAM Journal on Optimization **26**, 57–95 (2016)

[46] Aybat, N., Fallah, A., Gürbüzbalaban, M., Ozdaglar, A.: Robust accelerated gradient methods for smooth strongly convex functions. SIAM Journal on Optimization **30**, 717–751 (2020)

[47] Cyrus, S., Hu, B., Scoy, B.V., Lessard, L.: A robust accelerated optimization algorithm for strongly convex functions. In: IEEE Annual American Control Conference, pp. 1376–1381 (2018)

[48] Nesterov, Y.: A method for solving the convex programming problem with convergence rate $o(1/k^2)$. Soviet Mathematics Doklady **269**(3), 543–547 (1983)

[49] Arjevani, Y., Shalev-Shwartz, S., Shamir, O.: On lower and upper bounds in smooth and strongly convex optimization. Journal of Machine Learning Research (JMLR) **17**(1), 4303–4353 (2016)

[50] Rockafeller, R.: Augmented lagrangians and applications of the proximal point algorithm in convex programming. Mathematics of Operations Research **1**(2) (1976)

[51] Koshal, J., Nedić, A., Shanbhag, U.: Multiuser optimization: distributed algorithms and error analysis. SIAM Journal on Optimization **21**(3), 1046–1081 (2011)

[52] Karimi, H., Nutini, J., Schmidt, M.: Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In: Machine Learning and Knowledge Discovery in Database - European Conference, pp. 795–811 (2016)

[53] Jurcevic, P., Javadi-Abhari, A., Bishop, L.S., Lauer, I., Bogorin, D.F., Brink, M., Capelluto, L., Günlük, O., Itoko, T., Kanazawa, N., et al.: Demonstration of quantum volume 64 on a superconducting quantum computing system. Quantum Science and Technology **6**(2), 025,020 (2021)

[54] Nielsen, M.A., Chuang, I.: Quantum computation and quantum information. American Association of Physics Teachers (2002)

[55] Dawson, C.M., Nielsen, M.A.: The Solovay-Kitaev algorithm. Quantum Information & Computation **6**(1), 81–95 (2006)

[56] Kliuchnikov, V., Maslov, D., Mosca, M.: Fast and efficient exact synthesis of single-qubit unitaries generated by Clifford and T gates. Quantum Information & Computation **13**(7-8), 607–630 (2013)

[57] Ross, N.J., Selinger, P.: Optimal ancilla-free Clifford+ T approximation of z-rotations. Quantum Information & Computation **16**(11-12), 901–953 (2016)

[58] Sarnak, P.: Letter to scott aaronson and andy pollington on the solovay-kitaev theorem and golden gates (2015)

[59] Selinger, P.: Generators and relations for n-qubit Clifford operators. Logical Methods in Computer Science **11** (2015)

[60] Maslov, D., Dueck, G.W., Miller, D.M., Negrevergne, C.: Quantum circuit simplification and level compaction. IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems **27**(3), 436–444 (2008)

[61] Amy, M., Maslov, D., Mosca, M., Roetteler, M.: A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits. IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems **32**(6), 818–830 (2013)

[62] Nam, Y., Ross, N.J., Su, Y., Childs, A.M., Maslov, D.: Automated optimization of large quantum circuits with continuous parameters. npj Quantum Information **4**(1), 23 (2018)

[63] Zhang, J., Vala, J., Sastry, S., Whaley, K.B.: Minimum construction of two-qubit quantum operations. Phys. Rev. Lett. **93**, 020,502 (2004)

[64] Maslov, D., Falconer, S.M., Mosca, M.: Quantum circuit placement. IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems **27**(4), 752–763 (2008)

[65] Bhattacharjee, D., Chattopadhyay, A.: Depth-optimal quantum circuit placement for arbitrary topologies. arXiv preprint arXiv:1703.08540 (2017)

[66] Oddi, A., Rasconi, R.: Greedy randomized search for scalable compilation of quantum circuits. In: Integration of Constraint Programming, Artificial Intelligence, and Operations Research, pp. 446–461. Cham (2018)

[67] Booth, K.E.C., Do, M., Beck, J.C., Rieffel, E., Venturelli, D., Frank, J.: Comparing and integrating constraint programming and temporal planning for quantum circuit compilation. In: Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS2018), pp. 366–374 (2018)

[68] Zulehner, A., Wille, R.: Compiling SU(4) Quantum Circuits to IBM QX Architectures. In: Proceedings of the 24th Asia and South Pacific Design Automation Conference, p. 185–190 (2019)

[69] Bhattacharjee, D., Saki, A.A., Alam, M., Chattopadhyay, A., Ghosh, S.: MUQUT: Multi-constraint quantum circuit mapping on NISQ computers: Invited paper. In: 2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 1–7 (2019)

[70] Zulehner, A., Paler, A., Wille, R.: An Efficient Methodology for Mapping Quantum Circuits to the IBM QX Architectures. IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems **38**(7), 1226–1236 (2019)

[71] Cowtan, A., Dilkes, S., Duncan, R., Krajenbrink, A., Simmons, W., Sivarajah, S.: On the Qubit Routing Problem. In: 14th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2019), vol. 135, pp. 5:1–5:32 (2019)

[72] Sivarajah, S., Dilkes, S., Cowtan, A., Simmons, W., Edgington, A., Duncan, R.: t|ket⟩: a retargetable compiler for NISQ devices. Quantum Science and Technology **6**(1), 014,003 (2020)

[73] Itoko, T., Raymond, R., Imamichi, T., Matsuo, A.: Optimization of quantum circuit mapping using gate transformation and commutation. Integration **70**, 43–50 (2020)

[74] Tan, B., Cong, J.: Optimality study of existing quantum computing layout synthesis tools. IEEE Trans. on Computers **70**(9), 1363–1373 (2021)

[75] Murali, P., Linke, N.M., Martonosi, M., Abhari, A.J., Nguyen, N.H., Alderete, C.H.: Full-stack, real-system quantum computer studies: Architectural comparisons and design insights. In: 2019 ACM/IEEE 46th Annual International Symposium on Computer Architecture (ISCA), pp. 527–540 (2019)

[76] Tan, B., Cong, J.: Optimal layout synthesis for quantum computing. In: 2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD), pp. 1–9. IEEE (2020)

[77] Rogers, L.: The synthesis of nearest neighbour compliant quantum circuits. Ph.D. thesis, Queen's University Belfast (2021)

[78] Nannicini, G., Bishop, L.S., Gunluk, O., Jurcevic, P.: Optimal qubit assignment and routing via integer programming. arXiv preprint arXiv:2106.06446 (2021)

[79] Barenco, A., Bennett, C.H., Cleve, R., DiVincenzo, D.P., Margolus, N., Shor, P., Sleator, T., Smolin, J.A., Weinfurter, H.: Elementary gates for quantum computation. Physical review A **52**(5), 3457 (1995)

[80] Bullock, S.S., Markov, I.L.: An arbitrary two-qubit computation in 23 elementary gates or less. In: Proceedings of the 40th annual Design Automation Conference, pp. 324–329 (2003)

[81] Shende, V.V., Markov, I.L., Bullock, S.S.: Minimal universal two-qubit controlled-NOT-based circuits. Phys. Rev. A **69**, 062,321 (2004)

[82] Shende, V.V., Bullock, S.S., Markov, I.L.: Synthesis of quantum-logic circuits. IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems **25**(6), 1000–1010 (2006)

[83] Vatan, F., Williams, C.: Optimal quantum circuits for general two-qubit gates. Physical Review A **69**(3), 032,315 (2004)

[84] Drury, B., Love, P.: Constructive quantum Shannon decomposition from Cartan involutions. Journal of Physics A: Mathematical and Theoretical **41**(39), 395,305 (2008)

[85] Nakajima, Y., Kawano, Y., Sekigawa, H.: A new algorithm for producing quantum circuits using KAK decompositions. Quantum Information & Computation **6**(1), 67–80 (2006)

[86] Cincio, L., Subaşı, Y., Sornborger, A.T., Coles, P.J.: Learning the quantum algorithm for state overlap. New Journal of Physics **20**(11), 113,022 (2018)

[87] Younis, E., Sen, K., Yelick, K., Iancu, C.: QFAST: Quantum synthesis using a hierarchical continuous circuit space. arXiv preprint arXiv:2003.04462 (2020)

[88] Younis, E., Sen, K., Yelick, K., Iancu, C.: QFAST: Conflating search and numerical optimization for scalable quantum circuit synthesis. arXiv preprint arXiv:2103.07093 (2021)

[89] Rakyta, P., Zimborás, Z.: Sequential quantum gate decomposer (SQUANDER) (2021). URL `https://doi.org/10.5281/zenodo.4508680`

[90] Rakyta, P., Zimborás, Z.: Approaching the theoretical limit in quantum gate decomposition. arXiv preprint arXiv:2109.06770 (2021)

[91] Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM journal on imaging sciences **2**(1), 183–202 (2009)

[92] Simon, N., Friedman, J., Hastie, T., Tibshirani, R.: A sparse-group lasso. Journal of Computational and Graphical Statistics **22**(2), 231–245 (2013)

[93] Quantum, I.: Qiskit: An Open-source Framework for Quantum Computing (2019)

[94] Knapp, A.W.: Lie groups beyond an introduction, vol. 140. Springer Science & Business Media (2013)

[95] Lee, J.M.: Introduction to Smooth Manifolds. Springer (2013)

[96] Horn, R.A., Johnson, C.R.: Matrix analysis, 2nd ed edn. Cambridge University Press, Cambridge ; New York (2012)

[97] Lee, J.D., Simchowitz, M., Jordan, M.I., Recht, B.: Gradient descent only converges to minimizers. In: Conference on Learning Theory (COLT), vol. 49, pp. 1246–1257 (2016)

[98] Jin, C., Ge, R., Netrapalli, P., Kakade, S.M., Jordan, M.I.: How to escape saddle points efficiently. In: International Conference on Machine Learning (ICML), vol. 70, pp. 1724–1732 (2017)

[99] Jin, C., Netrapalli, P., Jordan, M.I.: Accelerated gradient descent escapes saddle points faster than gradient descent. In: Conference on Learning Theory (COLT), vol. 75, pp. 1042–1085 (2018)

[100] Ge, R., Jin, C., Zheng, Y.: No spurious local minima in nonconvex low rank problems: A unified geometric analysis. In: International Conference on Machine Learning (ICML), vol. 70, p. 1233–1242 (2017)

[101] Beth, T., Rötteler, M.: Quantum algorithms: applicable algebra and quantum physics. In: Quantum information, pp. 96–150. Springer (2001)

[102] Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT press (2016)

[103] Linnainmaa, S.: Taylor expansion of the accumulated rounding error. BIT Numerical Mathematics **16**(2), 146–160 (1976)

[104] Benedetti, M., Garcia-Pintos, D., Perdomo, O., Leyton-Ortega, V., Nam, Y., Perdomo-Ortiz, A.: A generative modeling approach for benchmarking and training shallow quantum circuits. npj Quantum Information **5**(1), 45 (2019)

[105] Linke, N.M., Maslov, D., Roetteler, M., Debnath, S., Figgatt, C., Landsman, K.A., Wright, K., Monroe, C.: Experimental comparison of two quantum computing architectures. Proceedings of the National Academy of Sciences **114**(13), 3305–3310 (2017)

[106] Shende, V.V., Markov, I.L.: On the CNOT-cost of TOFFOLI gates. Quantum Info. Comput. **9**(5), 461–486 (2009)

[107] Maslov, D.: Advantages of using relative-phase Toffoli gates with an application to multiple control Toffoli optimization. Phys. Rev. A **93**, 022,311 (2016)

[108] Patel, K.N., Markov, I.L., Hayes, J.P.: Optimal synthesis of linear reversible circuits. Quantum Information & Computation **8**(3), 282–294 (2008)

[109] Aaronson, S., Gottesman, D.: Improved simulation of stabilizer circuits. Physical Review A **70**(5), 052,328 (2004)

[110] Amy, M., Chen, J., Ross, N.J.: A finite presentation of CNOT-dihedral operators. In: International Conference on Quantum Physics and Logic, pp. 84–97 (2017)

[111] Maslov, D., Roetteler, M.: Shorter stabilizer circuits via Bruhat decomposition and quantum circuit transformations. IEEE Trans. on Information Theory **64**(7), 4729–4738 (2018)

[112] Garion, S., Cross, A.W.: On the structure of the CNOT-dihedral group. arXiv preprint arXiv:2006.12042 (2020)

[113] Bravyi, S., Maslov, D.: Hadamard-free circuits expose the structure of the Clifford group. IEEE Trans. on Information Theory **67**(7), 4546–4563 (2021)

[114] Garcia-Escartin, J.C., Chamorro-Posada, P.: Equivalent quantum circuits. arXiv preprint arXiv:1110.2998 (2011)

[115] Nash, B., Gheorghiu, V., Mosca, M.: Quantum circuit optimizations for NISQ architectures. Quantum Science and Technology **5**(2), 025,010 (2020)

[116] Wu, B., He, X., Yang, S., Shou, L., Tian, G., Zhang, J., Sun, X.: Optimization of CNOT circuits on topological superconducting processors. arXiv preprint arXiv:1910.14478 (2019)

[117] Kissinger, A., Meijer-van de Griend, A.: CNOT circuit extraction for topologically-constrained quantum memories. Quantum Information and Computation **20**(7&8), 581–596 (2020)

[118] de Griend, A.M.v., Duncan, R.: Architecture-aware synthesis of phase polynomials for NISQ devices. arXiv preprint arXiv:2004.06052 (2020)

[119] Gheorghiu, V., Li, S.M., Mosca, M., Mukhopadhyay, P.: Reducing the CNOT count for Clifford+ T circuits on NISQ architectures. arXiv preprint arXiv:2011.12191 (2020)

[120] Becker, S., Bobin, J., Candès, E.J.: NESTA: A fast and accurate first-order method for sparse recovery. SIAM Journal on Imaging Sciences **4**(1), 1–39 (2011)

[121] Candes, E.J., Wakin, M.B., Boyd, S.P.: Enhancing sparsity by reweighted ℓ-1 minimization. Journal of Fourier analysis and applications **14**(5-6), 877–905 (2008)

[122] Huang, Z., Becker, S.: Perturbed proximal descent to escape saddle points for non-convex and non-smooth objective functions. In: INNS Big Data and Deep Learning Conference, pp. 58–77 (2019)

[123] Mosci, S., Rosasco, L., Santoro, M., Verri, A., Villa, S.: Solving structured sparsity regularization with proximal methods. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 418–433 (2010)

[124] Peterson, E.C., Crooks, G.E., Smith, R.S.: Two-qubit circuit depth and the monodromy polytope. Quantum **4**(1), 247 (2020)

[125] Cross, A.W., Bishop, L.S., Sheldon, S., Nation, P.D., Gambetta, J.M.: Validating quantum computers using randomized model circuits. Physical Review A **100**(3), 032,328 (2019)

[126] Crooks, G.E.: Gates, states, and circuits (2020)

[127] Li, Y., Yuan, Y.: Convergence analysis of two-layer neural networks with relu activation. In: Neural Information Processing Systems (NeurIPS), vol. 30 (2017)

[128] Allen-Zhu, Z., Li, Y., Song, Z.: On the convergence rate of training recurrent neural networks. In: Neural Information Processing Systems (NeurIPS), vol. 32 (2019)

[129] Allen-Zhu, Z., Li, Y., Song, Z.: A convergence theory for deep learning via over-parameterization. In: International Conference on Machine Learning (ICML), vol. 97, pp. 242–252 (2019)

[130] Liu, C., Zhu, L., Belkin, M.: Loss landscapes and optimization in over-parameterized non-linear systems and neural networks. Applied and Computational Harmonic Analysis (2022)

[131] Gürbüzbalaban, M., Şimşekli, U., Zhu, L.: The heavy-tail phenomenon in SGD. In: International Conference on Machine Learning (ICML), vol. 139, pp. 3964–3975 (2021)

[132] Şimşekli, U., Sagun, L., Gürbüzbalaban, M.: A tail-index analysis of stochastic gradient noise in deep neural networks. In: International Conference on Machine Learning (ICML), vol. 97, pp. 5827–5837 (2019)

[133] Panigrahi, A., Somani, R., Goyal, N., Netrapalli, P.: Non-gaussianity of stochastic gradient noise. In: Science meets Engineering of Deep Learning (SEDL) workshop, 33rd Conference on Neural Information Processing Systems (NeurIPS) (2019)

[134] Patel, V.: Stopping criteria for, and strong convergence of, stochastic gradient descent on Bottou-Curtis-Nocedal functions. Mathematical Programming **164** (2021)

[135] Ghadimi, S., Lan, G.: Stochastic first-and zeroth-order methods for nonconvex stochastic programming. SIAM Journal on Optimization **23**(4), 2341–2368 (2013)

[136] Khaled, A., Richtárik, P.: Better theory for SGD in the nonconvex world. arXiv preprint arXiv:2002.03329 (2020)

[137] Sebbouh, O., Gower, R.M., Defazio, A.: Almost sure convergence rates for stochastic gradient descent and stochastic heavy ball. In: Conference on Learning Theory (COLT), pp. 3935–3971 (2021)

[138] Li, X., Orabona, F.: A high probability analysis of adaptive sgd with momentum. arXiv preprint arXiv:2007.14294 (2020)

[139] Zhang, C., Bengio, S., Hardt, M., Recht, B., Vinyals, O.: Understanding deep learning requires rethinking generalization. In: International Conference on Learning Representations (2017)

[140] Fan, X., Grama, I., Liu, Q., et al.: Exponential inequalities for martingales with applications. Electronic Journal of Probability **20** (2015)

[141] Bakhshizadeh, M., Maleki, A., de la Pena, V.H.: Sharp concentration results for heavy-tailed distributions. arXiv preprint arXiv:2003.13819 (2020)

[142] Nemirovski, A., Juditsky, A., Lan, G., Shapiro, A.: Robust stochastic approximation approach to stochastic programming. SIAM Journal on optimization **19**(4), 1574–1609 (2009)

[143] Bottou, L., Bousquet, O.: The tradeoffs of large scale learning. In: Neural Information Processing Systems (NeurIPS), vol. 20 (2008)

[144] Sun, R., Luo, Z.Q.: Guaranteed matrix completion via non-convex factorization. IEEE Trans. on Information Theory **62**(11), 6535–6579 (2016)

[145] Charles, Z., Papailiopoulos, D.: Stability and generalization of learning algorithms that converge to global optima. In: International Conference on Machine Learning (ICML), vol. 80, pp. 745–754 (2018)

[146] Kakade, S.M., Tewari, A.: On the generalization ability of online strongly convex programming algorithms. In: Neural Information Processing Systems (NeurIPS), pp. 801–808 (2008)

[147] Rakhlin, A., Shamir, O., Sridharan, K.: Making gradient descent optimal for strongly convex stochastic optimization. In: International Conference on Machine Learning (ICML) (2012)

[148] Hazan, E., Kale, S.: Beyond the regret minimization barrier: optimal algorithms for stochastic strongly-convex optimization. Journal of Machine Learning Research (JMLR) **15**, 2489–2512 (2014)

[149] Harvey, N.J.A., Liaw, C., Plan, Y., Randhawa, S.: Tight analyses for non-smooth stochastic gradient descent. In: Conference on Learning Theory (COLT), vol. 99, pp. 1579–1613 (2019)

[150] Harvey, N.J., Liaw, C., Randhawa, S.: Simple and optimal high-probability bounds for strongly-convex stochastic gradient descent. arXiv preprint arXiv:1909.00843 (2019)

[151] Davis, D., Drusvyatskiy, D., Xiao, L., Zhang, J.: From low probability to high confidence in stochastic convex optimization. Journal of Machine Learning Research (JMLR) **22**, 1–38 (2021)

[152] Catoni, O.: Challenging the empirical mean and empirical variance: a deviation study. Annales de l'IHP Probabilités et statistiques **48**(4), 1148–1185 (2012)

[153] Vladimirova, M., Girard, S., Nguyen, H., Arbel, J.: Sub-Weibull distributions: Generalizing sub-Gaussian and sub-exponential properties to heavier tailed distributions. Stat **9**(1), e318 (2020)

[154] Wong, K.C., Li, Z., Tewari, A.: Lasso guarantees for β-mixing heavy-tailed time series. The Annals of Statistics **48**(2), 1124–1142 (2020)

[155] Vershynin, R.: High-Dimensional Probability: An Introduction with Applications in Data Science, vol. 47. Cambridge University Press (2018)

[156] Vladimirova, M., Verbeek, J., Mesejo, P., Arbel, J.: Understanding priors in Bayesian neural networks at the unit level. In: International Conference on Machine Learning (ICML), pp. 6458–6467 (2019)

[157] Nguyen, T.H., Şimşekli, U., Gürbüzbalaban, M., Richard, G.: First exit time analysis of stochastic gradient descent under heavy-tailed gradient noise. In: Neural Information Processing Systems (NeurIPS), vol. 32 (2019)

[158] Hodgkinson, L., Mahoney, M.W.: Multiplicative noise and heavy tails in stochastic optimization. In: International Conference on Machine Learning (ICML), vol. 139, pp. 4262–4274 (2021)

[159] Zhang, J., Karimireddy, S.P., Veit, A., Kim, S., Reddi, S.J., Kumar, S., Sra, S.: Why are adaptive methods good for attention models? In: Neural Information Processing Systems (NeurIPS), vol. 33, pp. 15,383–15,393 (2020)

[160] Orvieto, A., Lucchi, A.: Continuous-time models for stochastic optimization algorithms. In: Neural Information Processing Systems (NeurIPS), pp. 12,589–12,601 (2019)

[161] Bousquet, O., Elisseeff, A.: Stability and generalization. Journal of Machine Learning Research (JMLR) **2**, 499–526 (2002)

[162] Elisseeff, A., Evgeniou, T., Pontil, M.: Stability of randomized learning algorithms. Journal of Machine Learning Research (JMLR) **6**, 55–79 (2005)

[163] Hardt, M., Recht, B., Singer, Y.: Train faster, generalize better: Stability of stochastic gradient descent. In: International Conference on Machine Learning (ICML), vol. 48, pp. 1225–1234 (2016)

[164] Mou, W., Wang, L., Zhai, X., Zheng, K.: Generalization bounds of SGLD for non-convex learning: Two theoretical viewpoints. In: Conference on Learning Theory (COLT), vol. 75, pp. 605–638 (2018)

[165] Shalev-Shwartz, S., Shamir, O., Srebro, N., Sridharan, K.: Learnability, stability and uniform convergence. Journal of Machine Learning Research (JMLR) **11**, 2635–2670 (2010)

[166] Feldman, V., Vondrak, J.: High probability generalization bounds for uniformly stable algorithms with nearly optimal rate. In: Conference on Learning Theory (COLT), vol. 99, pp. 1270–1279 (2019)

[167] Bousquet, O., Klochkov, Y., Zhivotovskiy, N.: Sharper bounds for uniformly stable algorithms. In: Conference on Learning Theory (COLT), vol. 125, pp. 610–626 (2020)

[168] Arjevani, Y., Carmon, Y., Duchi, J.C., Foster, D.J., Srebro, N., Woodworth, B.: Lower bounds for non-convex stochastic optimization. arXiv preprint arXiv:1912.02365 (2019)

[169] Freedman, D.A.: On tail probabilities for martingales. The Annals of Probability pp. 100–118 (1975)

[170] Mohri, M., Rostamizadeh, A., Talwalkar, A.: Foundations of Machine Learning, 2 edn. MIT press (2018)

[171] Devroye, L., Wagner, T.: Distribution-free inequalities for the deleted and holdout error estimates. IEEE Trans. on Information Theory **25**(2), 202–207 (1979)

[172] Petersen, K.B., Pedersen, M.: The Matrix Cookbook. `https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf` (2008)

[173] Billingsley, P.: Probability and Measure, 3 edn. John Wiley & Sons (1995)

[174] Jin, C., Netrapalli, P., Ge, R., Kakade, S.M., Jordan, M.I.: A short note on concentration inequalities for random vectors with subgaussian norm. arXiv preprint arXiv:1902.03736 (2019)

[175] Hunter, J.K., Nachtergaele, B.: Applied Analysis. World Scientific Publishing Company (2001)

[176] Ghadimi, S., Lan, G., Zhang, H.: Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. Mathematical Programming **155**(1-2), 267–305 (2016)

[177] Reddi, S.J., Sra, S., Poczos, B., Smola, A.J.: Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. In: Neural Information Processing Systems (NeurIPS), pp. 1153–1161 (2016)

Appendix A

Supplementary Material for Chapter 4

A.1 Standard Optimization Results

We state a few useful definitions and results from optimization. More details can be found in [15] and [17]. All our definitions are with respect to the Euclidean norm, denoted $\|\cdot\|$, and we let $f : \mathbb{R}^d \to \mathbb{R}$. We write $C^1(\mathbb{R}^d)$ or just C^1 to denote functionals on $\mathbb{R}^d$ that are continuously differentiable.

Definition 7 (Lipschitz continuity)**.** *We say $f : \mathbb{R}^d \to \mathbb{R}$ is ρ-Lipschitz continuous if $|f(x)-f(y)| \leq \rho\|x-y\|$ for all $x, y \in \mathbb{R}^d$. Likewise, $g : \mathbb{R}^d \to \mathbb{R}^m$ is ρ-Lipschitz continuous if $\|g(x)-g(y)\| \leq \rho\|x-y\|$ for all $x, y \in \mathbb{R}^d$.*

Note that averages of Lipschitz continuous functions are Lipschitz continuous with the same constant by the triangle inequality. Also, if $f \in C^1$, then f is ρ-Lipschitz if and only if $\|\nabla f(x)\| \leq \rho$ for all $x \in \mathbb{R}^d$.

Definition 8 (L-smoothness)**.** *We say f is L-smooth if $f \in C^1$ and its gradient is L-Lipschitz continuous.*

If f is L-smooth, then a standard result is

$$(\forall x, y \in \mathbb{R}^d)\ f(y) \leq f(x) + \langle \nabla f(x), y - x\rangle + \frac{L}{2}\|x - y\|^2. \tag{A.1}$$

Applying this result to $x - \frac{1}{L}\nabla f(x)$ and x and rearranging terms gives

$$\|\nabla f(x)\|^2 \leq 2L(f(x) - f^\star) \tag{A.2}$$

and taking $x \in \operatorname{argmin}_{x'} f(x')$ gives

$$f(y) - f^\star \leq \frac{L}{2}\|y - x^\star\|^2.$$

If f is both μ-PL and L-smooth, then combining the PL inequality and Eq. (A.2) gives

$$f(x) - f^\star \leq \frac{L}{\mu}\left(f(x) - f^\star\right)$$

which implies $L \geq \mu$.

The following inequality is known as the **quadratic growth** (QG) condition:

$$\exists \mu > 0 \quad \text{s.t. } (\forall x \in \mathbb{R}^d)\ \|x - \mathcal{P}_{X^\star}(x)\| \leq \frac{2}{\mu}\left(f(x) - f^\star\right) \tag{A.3}$$

where $\mathcal{P}_{X^\star}$ denotes the (set-valued) Euclidean projection onto the set of global minimizers, $X^\star = \operatorname{argmin}_x f(x)$. The PL inequality implies the QG condition [52].

Strong convexity implies the PL inequality but the PL inequality only implies invexity, not convexity. Furthermore, unlike the convexity and strong convexity properties, the PL property is not preserved under nonnegative sums. The PL inequality implies the quadratic growth property (Eq. A.3), which can be used to bound how far away the nearest minimizer is in terms of the optimality gap (which can be bounded above by simply evaluating the cost function, if the cost function is nonnegative).

A.2 Convergence of Random Variables

See [173, Sec. 20] for details. A sequence of random variables X_t converges to X :

- in probability if, for all $\epsilon > 0$, $\lim_t P(|X_t - X| > \epsilon) = 0$, denoted $X_t \xrightarrow{p} X$;
- in mean if $\lim_t \mathbb{E}|X_t - X| = 0$, denoted $X_t \xrightarrow{L^1} X$;
- almost surely if $P(\lim_t X_t = X) = 1$, denoted $X_t \xrightarrow{a.s.} X$.

When the *rate* of convergence is of interest, we say a sequence of random variables X_t converges to X:

- with mean convergence rate $r(t)$ if, for all t, $\mathbb{E}|X_t - X| \leq r(t)$;
- with convergence rate $\tilde{r}(t, \delta)$ if, for all t and δ, $P\left(|X_t - X| \leq \tilde{r}(t, \delta)\right) \geq 1 - \delta$.

In particular, when $\tilde{r}(t, \delta)$ is separable, i.e. $\tilde{r}(t, \delta) = r(t)p(\delta)$, we use "high probability" to refer to $p(\delta) = \text{poly}(\log(1/\delta))$ and "low probability" to refer to $p(\delta) = \text{poly}(1/\delta)$. All five kinds of convergence are interrelated:

- Convergence in mean and convergence almost surely both imply convergence in probability.
- Mean convergence with rate $r(t)$ such that $r(t) \to 0$ as $t \to \infty$ implies convergence in mean.
- By Markov's inequality, mean convergence with rate $r(t)$ implies convergence with rate $r(t)/\delta$.
- By the Borel-Cantelli lemma [173, Thm. 4.3], convergence with rate $r(t)p(\delta)$ implies almost sure convergence if $\sum_{t=0}^{\infty} \min\{1, p^{-1}(a/r(t))\} < \infty$ for all $a > 0$. If $p(\delta) = 1/\delta^c$ for some $c > 0$, then $r(t) = o(1/t^c)$ is required. If $p(\delta) = \log(1/\delta)$, then only $r(t) = O(1/t^c)$ for some $c > 0$ is required.

A.3 Proof of Proposition 1

We want to find requirements on a sequence (K_i) such that $\mathbb{E}\left[\exp(\lambda X_i)\right] \leq \exp(\lambda K_i)\ \forall \lambda \in [0, K_i^{-1}]$. Then, by Markov's inequality and taking $\lambda_n = K_n^{-1}$, $P\left(X_n \geq \log(e/\delta)K_n\right) = P\left(\exp(X_n/K_n) \geq e/\delta\right) \leq \delta$. Our proof is inductive. For the base case, we need

$$(1)\ K_0 \geq X_0.$$

For the induction step, assume $\mathbb{E}\left[\exp(\tilde{\lambda}X_i)\right] \leq \exp(\tilde{\lambda}K_i)\ \forall\tilde{\lambda} \in [0, K_i^{-1}]$. Let $\lambda \in [0, K_{i+1}^{-1}]$. Then

$$
\begin{aligned}
\mathbb{E}\left[\exp(\lambda X_{i+1})\right] &\leq \mathbb{E}\left[\exp(\lambda\alpha_i X_i + \lambda\hat{w}_i + \lambda\hat{v}_i)\right] \quad \text{by non-negativity} \\
&\overset{\text{TE}}{=} \mathbb{E}\left[\exp(\lambda\alpha_i X_i)\mathbb{E}\left[\exp(\lambda\hat{w}_i)\exp(\lambda\hat{v}_i) \mid \mathcal{F}_{i-1}\right]\right] \\
&\overset{\text{CS}}{\leq} \mathbb{E}\left[\exp(\lambda\alpha_i X_i)\mathbb{E}\left[\exp(2\lambda\hat{w}_i) \mid \mathcal{F}_{i-1}\right]^{1/2}\mathbb{E}\left[\exp(2\lambda\hat{v}_i) \mid \mathcal{F}_{i-1}\right]^{1/2}\right] \\
&\overset{(2)}{\leq} \mathbb{E}\left[\exp(\lambda\alpha_i X_i)\exp\left(\frac{(2\lambda)^2}{2}\beta_i^2 X_i\right)^{1/2}\exp(2\lambda\gamma_i)^{1/2}\right] \quad \text{if} \quad 2\lambda \in [0, \gamma_i^{-1}] \\
&= \mathbb{E}\left[\exp(\lambda\alpha_i X_i + \lambda^2\beta_i^2 X_i + \lambda\gamma_i)\right] \\
&= \mathbb{E}\left[\exp(\lambda(\alpha_i + \lambda\beta_i^2)X_i\right]\exp(\lambda\gamma_i) \\
&\overset{(3)}{\leq} \exp(\lambda((\alpha_i + \lambda\beta_i^2)K_i + \gamma_i)) \quad \text{via induction if } \tilde{\lambda} := \lambda(\alpha_i + \lambda\beta_i^2) \leq K_i^{-1} \\
&\overset{(4)}{\leq} \exp(\lambda K_{i+1})
\end{aligned}
$$

where TE denotes the law of total expectation [173, Thm. 34.3], CS denotes the Cauchy-Schwarz inequality, and $(2) - (4)$ are the requirements

$$
\begin{aligned}
&(2)\ 2\lambda \leq \gamma_i^{-1} \impliedby 2K_{i+1}^{-1} \leq \gamma_i^{-1} \iff K_{i+1} \geq 2\gamma_i \\
&(3)\ \lambda(\alpha_i + \lambda\beta_i^2) \leq K_i^{-1} \impliedby K_{i+1}^{-1}(\alpha_i + K_{i+1}^{-1}\beta_i^2) \leq K_i^{-1} \iff K_{i+1}^2 \geq \alpha_i K_i K_{i+1} + \beta_i^2 K_i \\
&(4)\ K_{i+1} \geq (\alpha_i + \lambda\beta_i^2)K_i + \gamma_i \impliedby K_{i+1} \geq (\alpha_i + K_{i+1}^{-1}\beta_i^2)K_i + \gamma_i \\
&\qquad\qquad \iff K_{i+1}^2 \geq (\alpha_i K_i + \gamma_i)K_{i+1} + \beta_i^2 K_i.
\end{aligned}
$$

The requirement of the proposition implies requirements $(2) - (4)$, completing the proof.

A.4 Proof of Theorem 11

We first note that η_t is decreasing in t for $\kappa \geq 1$ and $t \geq 0$, and so

$$
\eta_t = \frac{2t + 4\kappa - 1}{\mu(t + 2\kappa)^2} \leq \frac{4\kappa - 1}{\mu(2\kappa)^2} \leq \frac{4\kappa}{\mu 4\kappa^2} = \frac{1}{L}\,.
$$

Thus, we can use Eq. (4.3):

$$
f(x_{t+1}) - f^\star \leq (1 - \mu\eta_t)(f(x_t) - f^\star) + \eta_t(1 - L\eta_t)\langle\nabla f(x_t), e_t\rangle + \frac{L\eta_t^2}{2}\|e_t\|^2. \tag{A.4}
$$

Defining

$$X_t = (t+2\kappa-1)^2(f(x_t)-f^\star)$$

$$\hat{w}_t = \eta_t(1-L\eta_t)(t+2\kappa)^2\langle \nabla f(x_t), e_t\rangle$$

$$\hat{v}_t = \frac{L\eta_t^2(t+2\kappa)^2}{2}\|e_t\|^2$$

then multiplying both sides of Eq. (A.4) by $(t+2\kappa)^2$ and noting $1-\mu\eta_t = \frac{(t+2\kappa-1)^2}{(t+2\kappa)^2}$ gives the recursion

$$X_{t+1} \le X_t + \hat{w}_t + \hat{v}_t.$$

Observe that

$$\begin{aligned}
\mathbb{E}\left[\exp\left(\frac{\hat{w}_t^2}{\frac{18L}{\mu^2}X_t\sigma^2}\right)\,\middle|\,\mathcal{F}_{t-1}\right] &\le \mathbb{E}\left[\exp\left(\frac{\eta_t^2(1-L\eta_t)^2(t+2\kappa)^4\|\nabla f(x_t)\|^2\|e_t\|^2}{\frac{18L}{\mu^2}X_t\sigma^2}\right)\,\middle|\,\mathcal{F}_{t-1}\right] \\
&\le \mathbb{E}\left[\exp\left(\frac{\eta_t^2(1-L\eta_t)^2\frac{(t+2\kappa)^4}{(t+2\kappa-1)^2}2LX_t\|e_t\|^2}{\frac{18L}{\mu^2}X_t\sigma^2}\right)\,\middle|\,\mathcal{F}_{t-1}\right] \quad \text{via (A.2)} \\
&\le \mathbb{E}\left[\exp\left(\frac{\frac{18L}{\mu^2}X_t\|e_t\|^2}{\frac{18L}{\mu^2}X_t\sigma^2}\right)\,\middle|\,\mathcal{F}_{t-1}\right] \\
&\le 2
\end{aligned}$$

since $\|e_t\|$ is assumed to be σ-sub-Gaussian. Similarly,

$$\begin{aligned}
\mathbb{E}\left[\exp\left(\frac{\hat{v}_t}{\frac{2L}{\mu^2}\sigma^2}\right)\,\middle|\,\mathcal{F}_{t-1}\right] &\le \mathbb{E}\left[\exp\left(\frac{\frac{2L}{\mu^2}\|e_t\|^2}{\frac{2L}{\mu^2}\sigma^2}\right)\,\middle|\,\mathcal{F}_{t-1}\right] \\
&\le 2.
\end{aligned}$$

Thus, conditioned on $\mathcal{F}_{t-1}$, $\hat{w}_t$ is centered and $\frac{18L}{\mu^2}X_t\sigma^2$-sub-Gaussian and $\hat{v}_t$ is $\frac{2L}{\mu^2}\sigma^2$-sub-exponential. Thus, for absolute constants a and b, by Prop. 2.5.2 and Prop. 2.7.1 of [155], we can set $\beta_t^2 = \frac{18aL\sigma^2}{\mu^2}$ and $\gamma_t = \frac{2bL\sigma^2}{\mu^2}$ and use Proposition 1. In particular, we can set

$$\begin{aligned}
K_T &= X_0 + \sum_{t=0}^{T-1}(2\gamma_t + \beta_t^2) \\
&= X_0 + \frac{(18a+2b)L\sigma^2 T}{\mu^2}.
\end{aligned}$$

Dividing by $(T+2\kappa)^2$ completes the proof.

A.5 PL Projected SGD

When working with constraints to a set $x \in X \subsetneq \mathbb{R}^d$, if f is strongly convex, then so is f added to the indicator function of X, and results for gradient descent methods easily extend to projected gradient descent. On the other hand, if f is PL, then f plus the indicator function is not KL (Kurdyka-Łojasiewicz). This has real impacts on gradient descent algorithms, where gradient descent might converge but projected gradient descent does not. For example, there is a smooth function, a mollified version of $f(x,y) = \left(a(x)_+^2 - b\left(|y| - c\right)_+\right)_+$, such that the PL inequality is satisfied but projected gradient descent does *not* converge to a minimizer; we formalize this in the remark below.

Remark 9. *Consider* $f(x,y) = \left(a(x)_+^2 - b\left(|y| - c\right)_+\right)_+$ *where* $(\cdot)_+$ *denotes* $\max(\cdot, 0)$ *and* $a, b > 0, c \geq 0$. *The minimum of* f *is 0 and* $X^\star = \{(x,y) \mid x \leq 0 \text{ or } |y| \geq \frac{a}{b}x^2 + c\}$. *If we use* $\varphi(x) = \mathcal{X}_{B_1(0)} \cdot \exp\left(-1/(1 - \|x\|^2)\right)/\Phi$*—where* $\mathcal{X}$ *denotes the indicator function,* $B_1(0)$ *denotes the ball of radius 1 centered at 0, and* Φ *is the normalization constant—to mollify* f*, then, for* $\epsilon < c$*,* f_ϵ *has PL constant* $2a$ *and smoothness constant* $2a$*. Consider the starting point* $(d, 0)$*. For* a, b, c, d *chosen appropriately, the distance from* $(d, 0)$ *to its projection onto* $X^\star$ *is strictly less than* d*. Thus, if we let* X *be the ball centered at* $(d, 0)$ *with radius equal to exactly that distance, then the constrained problem and the unconstrained problem have the same minimum. However, projected gradient flow, starting from* $(d, 0)$*, ends up stuck at a non-minimizer: the point of* X *closest to* $(0, 0)$*. See Figure A.1 for the contour plot when* $a = 1/10$*,* $b = 1 = c$*, and* $d = 10$*.*

In order to generalize gradient methods to projected gradient methods under PL-like assumptions, the proper generalization is that the function should satisfy a **proximal** PL inequality [52]. However, such an assumption is quite restrictive compared to the PL inequality. We leave the problem of convergence with just the PL inequality, via added noise or a Frank-Wolfe type construction, as a future research direction.

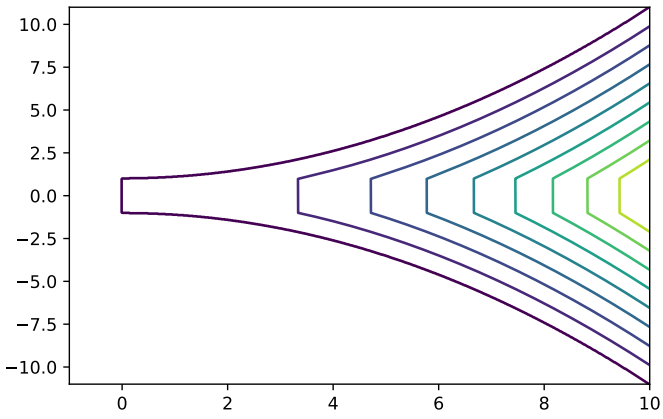


Figure A.1: Contour plot of the PL function counter-example to projected gradient flow

A.6 Generalization Results

A random vector X is σ-sub-Gaussian if $\langle u, X\rangle$ is σ-sub-gaussian for all unit vectors u [155, Def. 3.4.1]. Assume $\nabla f(x) - g(x,1)$ is $\sigma/\sqrt{d}$-sub-Gaussian. Then $\nabla f(x) - g(x,b)$ is $C\sigma/\sqrt{bd}$-sub-Gaussian for some universal constant C by Hoeffding's inequality [155, Prop. 2.6.1]. Furthmore, by Lemma 1 of [174], $\|\nabla f(x) - g(x,1)\|$ is $C\sigma/\sqrt{b}$-sub-Gaussian. So, $\|\nabla f(x) - g(x,1)\|^2$ is $C^2\sigma^2/b$-sub-exponential.

Definition 9. *Let S and S' differ in at most one sample. Let g and g' be the respective gradient estimators. Define $x_0 = x_0'$ and, for $t \geq 0$, $x_{t+1} = x_t - \eta_t g(x_t, b_t)$ and $x'_{t+1} = x'_t - \eta_t g'(x'_t, b_t)$. We say SGD with T iterations is α-uniformly stable in expectation if for all S and S', $\sup_s |\mathbb{E}[\ell(x_T, s) - \ell(x'_T, s)]| \leq \alpha$.*

Theorem 17. *[162, Thm. 12] Assume $\ell(x,s) \in [0, M]$ for all x and s. Let $\delta \in (0,1)$. If SGD with T steps is α-uniformly stable in expectation then,* w.p. $\geq 1 - \delta$ *over S and (i_t) for all SGD,*

$$\tilde{f}(x_T) - f(x_T) \leq \sqrt{\frac{6Mn\alpha + M^2}{2n\delta}}.$$

Note that Thm. 12 of [162] is actually for pointwise hypothesis stability in expectation, but uniform stability in expectation implies pointwise hypothesis stability in expectation. Furthermore, in the form we have it, Theorem 17 can be proved using Lem. 9 of [161] and Chebyshev's inequality.

Define $\delta_t := \|x_t - x'_t\|$. Note that $P(I_t = I'_t) = \left(1 - \frac{1}{n}\right) \cdots \left(1 - \frac{1}{n - b_t + 1}\right) = 1 - \frac{b_t}{n}$. Following [163], a uniform stability bound is sought by bounding the growth of δ_t; if $\ell(\cdot, s)$ is ρ-Lipschitz, this implies that $\sup_s |\ell(x_t, s) - \ell(x'_t, s)| \leq \rho\delta_t$.

Lemma 4. *[163, Lem. 2.5] Assume $\ell(\cdot, s)$ is ρ-Lipschitz and L-smooth for all s. Then,*

$$\delta_{t+1} \leq \begin{cases} (1 + \eta_t L)\delta_t & \text{if } I_t = I'_t \\ \delta_t + 2\eta_t \rho & \text{o.w.} \end{cases}.$$

Theorem 18. *Let $\eta_t \leq c/(t+1)$ and $b_t = b$. Assume $\ell(\cdot, s)$ is ρ-Lipschitz and L-smooth for all s. Then, SGD with T iterations is α-uniformly stable in expectation with*

$$\alpha \leq \frac{2\rho^2 T^{Lc}(c + 1/L)b}{n}.$$

Proof. Define $a_t = 0$ if $I_t = I'_t$ and $a_t = 1$ otherwise. Hence, $a_t \sim B(1, p)$ independently with $p = b/n$. By Lemma 4,

$$\begin{aligned}\delta_{t+1} &\leq (1 + (1 - a_t)\eta_t L)\delta_t + 2\eta_t \rho a_t \\ &\leq (1 + \eta_t L)\delta_t + 2\eta_t \rho a_t.\end{aligned}$$

Thus,

$$\begin{aligned}\delta_T &\leq 2\rho \sum_{t=0}^{T-1} \prod_{i=t+1}^{T-1} (1 + \eta_i L)\eta_t a_t \\ &\leq 2\rho \sum_{t=0}^{T-1} \exp\left(L \sum_{i=t+1}^{T-1} \eta_i\right) \eta_t a_t \\ &\leq 2\rho c \sum_{t=0}^{T-1} \exp\left(Lc \log\left(\frac{T}{t+1}\right)\right) \frac{a_t}{t+1} \\ &\leq 2\rho c T^{Lc} \sum_{t=0}^{T-1} \frac{a_t}{(t+1)^{Lc+1}}.\end{aligned}$$

Note that

$$\begin{aligned}\sum_{t=0}^{T-1} \frac{1}{(t+1)^{Lc+1}} &\leq 1 + \int_0^{T-1} \frac{dt}{(t+1)^{Lc+1}} \\ &= 1 + \frac{1}{Lc} - \frac{1}{LcT^{Lc}} \\ &\leq 1 + \frac{1}{Lc}.\end{aligned}$$

The result follows by taking expectation and using $\sup_s |\ell(x_t, s) - \ell(x'_t, s)| \leq \rho\delta_t$. □

Thm. 18 is essentially Thm. 3.12 of [163]. In fact, the two theorems consist of the same analysis with only two differences. First, we extend the analysis to mini-batch SGD, though this extension is not difficult. Second, [163] considers the first iteration where instability becomes non-zero. They use this to break up the bound and optimize over it. However, in so doing, they introduce the restriction $(2cL)^{1/(Lc+1)} T^{Lc/(Lc+1)} < n$. Thus, for simplicity, we left out this step. While it seems that makes our bound significantly larger, that is only when the restriction is not met (in which case their bound is vacuous).

To prove Theorem 13, we follow similar steps to Appendix A.4 but set

$$X_t = (t+1)^{\mu c}\left(f(x_t) - f^\star\right)$$

$$\hat{w}_t = \eta(1 - L\eta_t)(t+2)^{\mu c}\langle \nabla f(x_t), e_t\rangle$$

$$\hat{v}_t = \frac{L\eta_t^2(t+2)^{\mu c}}{2}\|e_t\|^2$$

and find that, conditioned on $\mathcal{F}_{t-1}$, $\hat{w}_t$ is centered and $\frac{6Lc^2}{(t+1)^{2-\mu c}}X_t\sigma^2$-sub-Gaussian and $\hat{v}_t$ is $\frac{Lc^2}{(t+1)^{2-\mu c}}\sigma^2$-sub-exponential. So, we can set

$$K_T = O\left(X_0 + \frac{Lc^2\sigma^2}{b}\right).$$

Dividing by $(T+1)^{\mu c}$ completes the proof.

Finally, note that the assumptions for generalization are interrelated with the assumptions for convergence:

Proposition 3. *Define f as in Eq. (4.4) and assume it is μ-PL. Assume $\ell(\cdot, s)$ is L-smooth for all s. Then f is L-smooth. Moreover, if $\nabla f(x) - g(x, 1)$ is sub-gaussian and $\ell(x, s)$ is nonnegative for all x and s, and assuming interpolation ($f^\star = 0$), then for (x_t) generated by SGD with a $\mathcal{O}(1/t)$ step-size sequence there exists a constant $R < \infty$ such that $P\left(\|x_t - x_0\| \leq R\ \forall t \in \mathbb{N}\right) = 1$. If s comes from a compact set (e.g., normalized data) and we assume $\ell(x, s)$ is jointly continuous in x and s, then there exist constants ρ and M such that, for all x in the ball of radius R around x_0 and for all s, $\ell(x, s)$ is ρ-Lipschitz with respect to x and is bounded above by M.*

Proof. The smoothness of f easily follows from that of ℓ using the definition of f in Eq. (4.4), the definition of smoothness, and the triangle inequality.

The rest of the proof will hinge on bounding the SGD sequence (x_t). Unrolling the SGD updates, we have $x_{T+1} = x_0 + \sum_{t=0}^{T}\eta_t g_t$. Define $\tilde{f}_t = \frac{1}{b_t}\sum_{i\in I_t}\ell(\cdot, s_i)$ so that $g(\cdot, b_t) = \nabla\tilde{f}_t$, and

observe $\tilde{f}$ is L-smooth for the same reason that f is L-smooth. Then,

$$\begin{aligned}\|g(x, b_t)\|^2 = \|\nabla \tilde{f}_t(x)\|^2 &\leq 2L(\tilde{f}_t(x) - \tilde{f}_t^\star) \\ &\leq 2L\tilde{f}_t(x) \\ &\leq 2L\frac{n}{b_t}f(x)\end{aligned}$$

where the first inequality follows from Eq. (A.2), the second from nonnegativity, and the third from nonnegativity as well. For simplicity, assume a fixed minibatch size $b_t = b$.

Thus, by the triangle inequality, $\|x_T - x_0\| \leq \sqrt{2L\frac{n}{b}} \sum_{t=0}^{T-1} \eta_t \sqrt{f(x_t)}$. For both Thm. 11 and Thm. 13, the step-size is $\eta_t = \mathcal{O}(1/t)$, and, by interpolation, $f(x_t) \leq \log(1/\delta_t)\frac{C}{t^c}$ for each t with probability $1 - \delta_t$, where C and $c > 0$ are constants (e.g., $c = 1$ for Thm. 11).[1] Choose $\delta_t = \mathcal{O}(1/t^2)$, then it holds that

$$\text{w.p.} \geq 1 - \delta, \quad (\forall T \in \mathbb{N}) \quad \|x_T - x_0\| \lesssim C' \sum_{t=1}^{T} \frac{\log(t)}{t^{1+c/2}} \leq C''$$

for constants C' and C'' since the series converges (verified by, e.g., the integral test) and with $\delta = \sum_{t=0}^{T-1} \delta_t \leq \sum_{t=0}^{\infty} \delta_t < 1$ coming from the union bound and the fact $\sum_{t=1}^{\infty} t^{-2} = \pi^2/6 < \infty$.

Let E_k be the event that $\|x_T - x_0\| \leq k$ for all T. Then the event $\bigcup_{k=1}^{\infty} E_k$ implies that $(x_t)_{t=0}^{\infty}$ is a bounded sequence. Since E_k is nested, by continuity of measure, $P(\bigcup_{k=1}^{\infty} E_k) = \lim_{k\to\infty} P(E_k)$, and the argument from the previous paragraph shows that $P(E_k)$ can be made arbitrarily close to 1 by choosing a sufficiently large radius k. Hence $(x_t)_{t=0}^{\infty}$ is a bounded sequence with probability 1.

Now that (x_t) is bounded with probability 1, we turn to ρ and M. First, $\ell(x, s)$ is jointly continuous in x and s, and since (x, s) comes from a compact set, we get that ℓ attains its maximum value [175, Thm. 1.68], which we call M. The same is true of $\|\nabla_x \ell(x, s)\|$ and we call its maximum value ρ. Thus, since $\ell(x, s)$ is C^1 with respect to x, we have that it is ρ-Lipschitz with respect to x. □

[1] These theorems and step-sizes are actually in terms of $t + 1$ due to the 0-based indexing convention, so we now use 1-based indexing to simplify presentation.

A.7 Sub-Weibull Properties

Lemma 5 provides a simple bound for $\mathbb{E}\left[|X|^p\right]$; the proof is included for completeness and to demonstrate the techniques. Lemma 6 is from Prop. 2.5.2 of [155]. Lemma 7 extends Prop. 2.7.1 of [155] to interpolate between the sub-Gaussian and sub-exponential regimes. Lemma 8 is an application of Thm. 1 of [153], Lma. 5 of [154], and the triangle inequality for L_p spaces (note that this is called the Minkowski inequality and requires $p \geq 1$).

To go from probability to expectation, the CDF formula is used:

$$\mathbb{E}\left[|Y|\right] = \int_0^\infty P\left(|Y| > t\right) dt.$$

To go from expectation to probability, Markov's inequality is used:

$$P\left(|Y| > t\right) \leq \frac{1}{t}\mathbb{E}\left[|Y|\right] \ \forall t > 0.$$

In both cases, the trick is choosing what Y should be, e.g. $Y = |X|^p$ or $Y = \exp((\lambda|X|)^{1/\theta})$.

Lemma 5. *If X is K-sub-Weibull(θ) then $\mathbb{E}\left[|X|^p\right] \leq 2\Gamma(\theta p + 1)K^p \ \forall p > 0$. In particular, $\mathbb{E}\left[X^2\right] \leq 2\Gamma(2\theta + 1)K^2$.*

Proof. First, for all $t \geq 0$,

$$\begin{aligned} P\left(|X| \geq t\right) &= P\left(\exp\left(\left(|X|/K\right)^{1/\theta}\right) \geq \exp\left(\left(t/K\right)^{1/\theta}\right)\right) \\ &\leq 2\exp\left(-\left(t/K\right)^{1/\theta}\right). \end{aligned}$$

Second,

$$\begin{aligned} \mathbb{E}\left[|X|^p\right] &= \int_0^\infty P\left(|X|^p \geq x\right) dx \\ &\leq 2\int_0^\infty \exp\left(-\left(x^{1/p}/K\right)^{1/\theta}\right) \\ &= 2\theta p K^p \int_0^\infty \exp(-u) u^{\theta p - 1} du \\ &= 2\theta p \Gamma(\theta p) K^p \\ &= 2\Gamma(\theta p + 1) K^p. \end{aligned}$$

□

Lemma 6. *[155, Prop. 2.5.2(e)] If X is centered and K-sub-Guassian then* $\mathbb{E}\left[\exp(\lambda X)\right] \leq \exp(\lambda^2 K^2)\ \forall \lambda \in \mathbb{R}$.

Lemma 7. *If X is centered and K-sub-Weibull(θ) with $\theta \in (1/2, 1]$, then* $\mathbb{E}\left[\exp(\lambda X)\right] \leq \exp\left(\frac{\lambda^2}{2}(4\theta)^{2\theta} e^2 K^2\right)$ *for all* $\lambda \in \left[0, \frac{1}{(4\theta)^\theta eK}\right]$.

Proof. First, using Lemma 5 and $\Gamma(x+1) \leq x^x\ \forall x \geq 1$, we can get $\|X\|_p \leq (2\theta)^\theta K p^\theta$ for all $p \geq 1/\theta$, and so, in particular, for all $p \geq 2$. Thus, if $\lambda \in \left[0, \frac{1}{(4\theta)^\theta eK}\right]$, then

$$\begin{aligned}
\mathbb{E}\left[\exp(\lambda X)\right] &= \mathbb{E}\left[1 + \lambda X + \sum_{p=2}^{\infty} \frac{(\lambda X)^p}{p!}\right] \\
&= 1 + \sum_{p=2}^{\infty} \frac{\lambda^p \|X\|_p^p}{p!} \\
&\leq 1 + \sum_{p=2}^{\infty} \frac{\lambda^p (2\theta)^{\theta p} K^p p^{\theta p}}{p!} \\
&\leq 1 + \sum_{p=2}^{\infty} \left(\frac{\lambda (2\theta)^\theta eK}{p^{1-\theta}}\right)^p \\
&\leq 1 + \sum_{p=2}^{\infty} \left(\lambda (4\theta)^\theta (e/2) K\right)^p \\
&\leq 1 + \frac{\left(\lambda (4\theta)^\theta (e/2) K\right)^2}{1 - \lambda (4\theta)^\theta (e/2) K} \\
&\leq 1 + 2\left(\lambda (4\theta)^\theta (e/2) K\right)^2 \\
&\leq \exp\left(\frac{\lambda^2}{2}(4\theta)^{2\theta} e^2 K^2\right),
\end{aligned}$$

completing the proof. □

Lemma 8. *[153, Thm. 1] [154, Lma. 5] Suppose $X_1, \ldots, X_n$ are sub-Weibull(θ) with respective parameters $K_1, \ldots, K_n$. Then, for all $t \geq 0$,*

$$P\left(\left|\sum_{i=1}^{n} X_i\right| \geq t\right) \leq 2\exp\left(-\left(\frac{t}{v(\theta)\sum_{i=1}^{n} K_i}\right)^{1/\theta}\right),$$

where $v(\theta) = (4e)^\theta$ for $\theta \leq 1$ and $v(\theta) = 2(2e\theta)^\theta$ for $\theta \geq 1$.

A.8 Proof of Proposition 2

The novel lemmas used in the proof are Lemma 10 and Lemma 14.

Observe that:

$$\begin{aligned} P\left(|\xi_i| \geq t \mid \mathcal{F}_{i-1}\right) &= P\left(\exp\left(\left(|\xi_i|/K_{i-1}\right)^{1/\theta}\right) \geq \exp\left(\left(t/K_{i-1}\right)^{1/\theta}\right) \mid \mathcal{F}_{i-1}\right) \\ &\leq 2\exp\left(-\left(t/K_{i-1}\right)^{1/\theta}\right). \end{aligned}$$

Moreover, observe that:

$$\begin{aligned} P\left(|\xi_i| \geq cK_{i-1}\right) &= P\left(\exp\left(\left(|\xi_i|/K_{i-1}\right)^{1/\theta}\right) \geq \exp\left(c^{1/\theta}\right)\right) \\ &\leq \exp\left(-c^{1/\theta}\right)\mathbb{E}\left[\exp\left(\left(|\xi_i|/K_{i-1}\right)^{1/\theta}\right)\right] \\ &= \exp\left(-c^{1/\theta}\right)\mathbb{E}\left[\mathbb{E}\left[\exp\left(\left(|\xi_i|/K_{i-1}\right)^{1/\theta}\right) \mid \mathcal{F}_{i-1}\right]\right] \\ &\leq 2\exp\left(-c^{1/\theta}\right). \end{aligned}$$

We call the first result tail bound 1:

$$P\left(|\xi_i| \geq t \mid \mathcal{F}_{i-1}\right) \leq 2\exp\left(-\left(t/K_{i-1}\right)^{1/\theta}\right) \tag{TB1}$$

and the second result tail bound 2:

$$P\left(|\xi_i| \geq cK_{i-1}\right) \leq 2\exp\left(-c^{1/\theta}\right) \tag{TB2}$$

The following lemma is not novel, but distills the proof technique of [169], as found in the proof of Thm. 2.1 of [140], so that we can use it more readily.

Lemma 9. *[140, Pf. of Thm. 2.1] Let $(\Omega, \mathcal{F}, (\mathcal{F}_i), P)$ be a filtered probability space. If (d_i) is adapted to $(\mathcal{F}_i)$ and (A_k) is a sequence of events; then*

$$P\left(\bigcup_{k\in[n]} A_k\right) \leq \sup_{k\in[n]} \mathbb{I}_{A_k} \prod_{i=1}^{k} \frac{\mathbb{E}\left[d_i \mid \mathcal{F}_{i-1}\right]}{d_i}.$$

Proof. Define $Z_k = \prod_{i=1}^{k} \frac{d_i}{\mathbb{E}[d_i|\mathcal{F}_{i-1}]}$. Then (Z_k) is a martingale.

Let T be a stopping time. Then the stopped process $(Z_{k\wedge T})$ is a martingale (where $a \wedge b$ denotes $\min\{a, b\}$). Moreover, $Z_{k\wedge T}$ is a probability density so define the conjugate probability measure $dP' = Z_{n\wedge T} dP$.

Define the stopping time $T(\omega) = \min\{k \in [n] \mid \omega \in A_k\}$. Then $\mathbb{I}_{\cup_{k\in[n]} A_k} = \sum_{i=1}^{n} \mathbb{I}_{\{T=k\}}$.

Observe,

$$\begin{aligned} P\left(\bigcup_{k\in[n]} A_k\right) &= \mathbb{E}'\left[Z_{n\wedge T}^{-1} \sum_{k=1}^{n} \mathbb{I}_{\{T=k\}}\right] \\ &= \sum_{k=1}^{n} \mathbb{E}'\left[\prod_{i=1}^{k} \frac{\mathbb{E}\left[d_i \mid \mathcal{F}_{i-1}\right]}{d_i} \mathbb{I}_{\{T=k\}}\right] \\ &\leq \left(\sup_{k\in[n]} \mathbb{I}_{A_k} \prod_{i=1}^{k} \frac{\mathbb{E}\left[d_i \mid \mathcal{F}_{i-1}\right]}{d_i}\right) \sum_{k=1}^{n} \mathbb{E}'\left[\mathbb{I}_{\{T=k\}}\right] \\ &= \sup_{k\in[n]} \mathbb{I}_{A_k} \prod_{i=1}^{k} \frac{\mathbb{E}\left[d_i \mid \mathcal{F}_{i-1}\right]}{d_i}. \end{aligned}$$

□

We apply Lemma 9 to a particular MGF bound to prove a generalization of the Generalized Freedman inequality of [149] to allow for up-to sub-exponential random variables. This is a novel contribution (although not a difficult extension). The extension beyond sub-exponential random variables is the more technical contribution.

Lemma 10. *Let $(\Omega, \mathcal{F}, (\mathcal{F}_i), P)$ be a filtered probability space. Let $(\widetilde{\xi}_i)$ and (K_i) be adapted to $(\mathcal{F}_i)$. Assume $0 \leq K_{i-1} \leq m_i$ $\forall i \in [n]$ and*

$$\mathbb{E}\left[\exp(\lambda\widetilde{\xi}_i) \mid \mathcal{F}_{i-1}\right] \leq \exp\left(\frac{\lambda^2}{2} a K_{i-1}^2\right) \ \forall \lambda \in \left[0, \frac{1}{bK_{i-1}}\right].$$

Then, for all $x, \beta \geq 0$, and $\alpha \geq b \max_{i\in[n]} m_i$, and $\lambda \in \left[0, \frac{1}{2\alpha}\right]$,

$$P\left(\bigcup_{k\in[n]} \left\{\sum_{i=1}^{k} \widetilde{\xi}_i \geq x \text{ and } a\sum_{i=1}^{k} K_{i-1}^2 \leq \alpha \sum_{i=1}^{k} \widetilde{\xi}_i + \beta\right\}\right) \leq \exp(-\lambda x + 2\lambda^2\beta).$$

Proof. By Claim C.2 of [149], if $0 \leq \lambda \leq \frac{1}{2\alpha}$, then $\exists\ c \in [0, 2]$ such that $\frac{1}{2}(\lambda + \alpha c\lambda^2)^2 = c\lambda^2$. Define

$$d_i = \exp\left((\lambda + \alpha c\lambda^2)\widetilde{\xi}_i\right).$$

With $0 \leq \lambda \leq \frac{1}{2\alpha}$, we want $\lambda + \alpha c\lambda^2 \leq \frac{1}{bK_{i-1}}$. This is ensured by $\alpha \geq b \max_{i\in[n]} m_i$.

Define

$$A_k = \left\{ \sum_{i=1}^{k} \widetilde{\xi}_i \geq x \text{ and } a\sum_{i=1}^{k} K_{i-1}^2 \leq \alpha \sum_{i=1}^{k} \widetilde{\xi}_i + \beta \right\}.$$

Then $\omega \in A_k$ implies

$$\begin{aligned}
\prod_{i=1}^{k} \frac{\mathbb{E}\left[d_i \mid \mathcal{F}_{i-1}\right]}{d_i} &\leq \exp\left(-(\lambda + \alpha c\lambda^2) \sum_{i=1}^{k} \widetilde{\xi}_i + \frac{(\lambda + \alpha c\lambda^2)^2}{2} a \sum_{i=1}^{k} K_{i-1}^2 \right) \\
&\leq \exp(-\lambda x + c\lambda^2 \beta) \\
&\leq \exp(-\lambda x + 2\lambda^2 \beta).
\end{aligned}$$

□

The next three lemmas allow us to include previous results as special cases of the theorem.

Lemma 11. *[140, Thm. 2.6] Let $(\Omega, \mathcal{F}, (\mathcal{F}_i), P)$ be a filtered probability space. Let $(\widetilde{\xi}_i)$ and (K_i) be adapted to $(\mathcal{F}_i)$. Assume $0 \leq K_{i-1} \leq m_i \ \forall i \in [n]$ and*

$$\mathbb{E}\left[\exp(\lambda \widetilde{\xi}_i) \mid \mathcal{F}_{i-1}\right] \leq \exp\left(\frac{\lambda^2}{2} a K_{i-1}^2\right) \ \forall \lambda \in \left[0, \frac{1}{bK_{i-1}}\right].$$

Then, for all $x, \beta \geq 0$, and $\lambda \in \left[0, \frac{1}{b \max_{i\in[r]} m_i}\right]$,

$$P\left(\bigcup_{k\in[n]} \left\{ \sum_{i=1}^{k} \widetilde{\xi}_i \geq x \text{ and } a\sum_{i=1}^{k} K_{i-1}^2 \leq \beta \right\} \right) \leq \exp\left(-\lambda x + \frac{\lambda^2}{2}\beta\right).$$

Proof. Define

$$d_i = \exp\left(\lambda \widetilde{\xi}_i\right)$$

and

$$A_k = \left\{ \sum_{i=1}^{k} \widetilde{\xi}_i \geq x \text{ and } a\sum_{i=1}^{k} K_{i-1}^2 \leq \beta \right\}.$$

Then $\omega \in A_k$ implies

$$\begin{aligned}
\prod_{i=1}^{k} \frac{\mathbb{E}\left[d_i \mid \mathcal{F}_{i-1}\right]}{d_i} &\leq \exp\left(-\lambda \sum_{i=1}^{k} \widetilde{\xi}_i + \frac{\lambda^2}{2} a \sum_{i=1}^{k} K_{i-1}^2 \right) \\
&\leq \exp\left(-\lambda x + \frac{\lambda^2}{2}\beta\right).
\end{aligned}$$

□

Lemma 12. *[149, Thm. 3.3] Let $(\Omega, \mathcal{F}, (\mathcal{F}_i), P)$ be a filtered probability space. Let $(\widetilde{\xi}_i)$ and (K_i) be adapted to $(\mathcal{F}_i)$. Assume $K_{i-1} \geq 0 \ \forall i \in [n]$ and*

$$\mathbb{E}\left[\exp(\lambda\widetilde{\xi}_i) \mid \mathcal{F}_{i-1}\right] \leq \exp\left(\frac{\lambda^2}{2} a K_{i-1}^2\right) \ \forall \lambda \geq 0.$$

Then, for all $x, \beta, \alpha \geq 0$, and $\lambda \in \left[0, \frac{1}{2\alpha}\right]$,

$$P\left(\bigcup_{k\in[n]} \left\{ \sum_{i=1}^{k} \widetilde{\xi}_i \geq x \text{ and } a\sum_{i=1}^{k} K_{i-1}^2 \leq \alpha \sum_{i=1}^{k} \widetilde{\xi}_i + \beta \right\}\right) \leq \exp(-\lambda x + 2\lambda^2\beta).$$

Lemma 13. *[169] Let $(\Omega, \mathcal{F}, (\mathcal{F}_i), P)$ be a filtered probability space. Let $(\widetilde{\xi}_i)$ and (K_i) be adapted to $(\mathcal{F}_i)$. Assume $K_{i-1} \geq 0 \ \forall i \in [n]$ and*

$$\mathbb{E}\left[\exp(\lambda\widetilde{\xi}_i) \mid \mathcal{F}_{i-1}\right] \leq \exp\left(\frac{\lambda^2}{2} a K_{i-1}^2\right) \ \forall \lambda \geq 0.$$

Then, for all $x, \beta \geq 0$, and $\lambda \geq 0$,

$$P\left(\bigcup_{k\in[n]} \left\{ \sum_{i=1}^{k} \widetilde{\xi}_i \geq x \text{ and } a\sum_{i=1}^{k} K_{i-1}^2 \leq \beta \right\}\right) \leq \exp\left(-\lambda x + \frac{\lambda^2}{2}\beta\right).$$

If the ξ_i's are sub-Gaussian, that is, if $\theta = 1/2$, then, from Lemma 6,

$$\mathbb{E}\left[\exp(\lambda\xi_i) \mid \mathcal{F}_{i-1}\right] \leq \exp\left(\frac{\lambda^2}{2} 2K_{i-1}^2\right) \ \forall \lambda \in \mathbb{R},$$

so we can apply Lemma 12 if $\alpha > 0$ or Lemma 13 if $\alpha = 0$ and we are finished.

If the ξ_i's are at most sub-exponential, that is, if $1/2 < \theta \leq 1$, then, from Lemma 7,

$$\mathbb{E}\left[\exp(\lambda\xi_i) \mid \mathcal{F}_{i-1}\right] \leq \exp\left(\frac{\lambda^2}{2}(4\theta)^{2\theta} e^2 K_{i-1}^2\right) \ \forall \lambda \in \left[0, \frac{1}{(4\theta)^\theta e K_{i-1}}\right],$$

so we can apply Lemma 10 if $\alpha > 0$ or Lemma 11 if $\alpha = 0$ and we are finished.

On the other hand, if $\theta > 1$, then it may be the case that

$$\mathbb{E}\left[\exp(\lambda\xi_i) \mid \mathcal{F}_{i-1}\right] = \infty \ \forall \lambda > 0.$$

Thus, to allow for $\theta > 1$, we will truncate the ξ_i's. The following lemma is the main contribution of this proof as it allows us to go beyond the sub-exponential threshold. Note that the result/proof is essentially an extension of Corollary 2 of [141] from i.i.d. random variables to martingale difference sequences.

Lemma 14. *Let $(\Omega, \mathcal{F}, (\mathcal{F}_i), P)$ be a filtered probability space. Assume (ξ_i) is a martingale difference sequence. Let (K_i) be adapted to $(\mathcal{F}_i)$. Assume $0 \leq K_{i-1} \leq m_i \ \forall i \in [n]$ and*

$$\mathbb{E}\left[\exp\left((|\xi_i|/K_{i-1})^{1/\theta}\right) \mid \mathcal{F}_{i-1}\right] \leq 2 \ \forall i \in [n]$$

where $\theta > 1$. Let $\delta \in (0,1)$. Define

$$\widetilde{\xi}_i = \xi_i \mathbb{I}_{\{\xi_i \leq L_{i-1}\}}$$

with

$$L_{i-1} = \log(n/\delta)^{\theta} K_{i-1}.$$

Then

$$\mathbb{E}\left[\exp(\lambda\widetilde{\xi}_i) \mid \mathcal{F}_{i-1}\right] \leq \exp\left(\frac{\lambda^2}{2} a K_{i-1}^2\right) \ \forall \lambda \in \left[0, \frac{1}{bK_{i-1}}\right]$$

where

$$a = (2^{2\theta+1}+2)\Gamma(2\theta+1) + \frac{2^{3\theta}\Gamma(3\theta+1)}{3\log(n/\delta)^{\theta-1}}$$

$$b = 2\log(n/\delta)^{\theta-1}.$$

Proof. Let $\lambda \in \left[0, \frac{1}{bK_{i-1}}\right]$. That is, $\lambda \geq 0$ and $\lambda L_{i-1}^{1-1/\theta} K_{i-1}^{1/\theta} \leq \frac{1}{2}$. Since $\lambda \geq 0$, we have, by Lemma 4 of [141],

$$\mathbb{E}\left[\exp(\lambda\widetilde{\xi}_i) \mid \mathcal{F}_{i-1}\right] \leq \exp\left(\frac{\lambda^2}{2}\left(\mathbb{E}\left[\xi_i^2 \mathbb{I}_{\{\xi_i<0\}} \mid \mathcal{F}_{i-1}\right] + \mathbb{E}\left[\xi_i^2 \exp(\lambda\xi_i)\mathbb{I}_{\{0\leq\xi_i\leq L_{i-1}\}} \mid \mathcal{F}_{i-1}\right]\right)\right).$$

Observe

$$\begin{aligned}
\mathbb{E}\left[\xi_i^2 \mathbb{I}_{\{\xi_i<0\}} \mid \mathcal{F}_{i-1}\right] &= \int_0^\infty P\left(\xi_i^2 \mathbb{I}_{\{\xi_i<0\}} > x \mid \mathcal{F}_{i-1}\right) dx \\
&= \int_0^\infty P\left(\xi_i^2 > t^2, \xi_i < 0 \mid \mathcal{F}_{i-1}\right) 2t dt \\
&\leq \int_0^\infty P\left(|\xi_i| > t \mid \mathcal{F}_{i-1}\right) 2t dt \\
&\overset{TB1}{\leq} 2\int_0^\infty \exp\left(-(t/K_{i-1})^{1/\theta}\right) 2t dt \\
&= 2\Gamma(2\theta+1)K_{i-1}^2
\end{aligned}$$

and

$$
\begin{aligned}
&\mathbb{E}\left[\xi_i^2 \exp(\lambda\xi_i)\mathbb{I}_{\{0\le\xi_i\le L_{i-1}\}} \mid \mathcal{F}_{i-1}\right] \\
&= \int_0^\infty P\left(\xi_i^2 \exp(\lambda\xi_i)\mathbb{I}_{\{0\le\xi_i\le L_{i-1}\}} > x \mid \mathcal{F}_{i-1}\right) dx \\
&= \int_0^\infty P\left(\xi_i^2 \exp(\lambda\xi_i) > t^2\exp(\lambda t), 0 \le \xi_i \le L_{i-1} \mid \mathcal{F}_{i-1}\right)(2t+\lambda t^2)\exp(\lambda t)dt \\
&= \int_0^\infty P\left(|\xi_i| > t, 0 \le \xi_i \le L_{i-1} \mid \mathcal{F}_{i-1}\right)(2t+\lambda t^2)\exp(\lambda t)dt \\
&\le \int_0^{L_{i-1}} P\left(|\xi_i| > t \mid \mathcal{F}_{i-1}\right)(2t+\lambda t^2)\exp(\lambda t)dt \\
&\overset{TB1}{\le} 2\int_0^{L_{i-1}} \exp\left(-\left(1-\lambda t^{1-1/\theta}K_{i-1}^{1/\theta}\right)(t/K_{i-1})^{1/\theta}\right)(2t+\lambda t^2)dt \\
&\le 2\int_0^{L_{i-1}} \exp\left(-\left(1-\lambda L_{i-1}^{1-1/\theta}K_{i-1}^{1/\theta}\right)(t/K_{i-1})^{1/\theta}\right)(2t+\lambda t^2)dt \\
&= 2\int_0^{L_{i-1}} \exp(-u)\left(\frac{2K_{i-1}^2\theta u^{2\theta-1}}{\left(1-\lambda L_{i-1}^{1-1/\theta}K_{i-1}^{1/\theta}\right)^{2\theta}} + \frac{\lambda K_{i-1}^3\theta u^{3\theta-1}}{\left(1-\lambda L_{i-1}^{1-1/\theta}K_{i-1}^{1/\theta}\right)^{3\theta}}\right)du \\
&= \frac{2K_{i-1}^2\Gamma(2\theta+1)}{\left(1-\lambda L_{i-1}^{1-1/\theta}K_{i-1}^{1/\theta}\right)^{2\theta}} + \frac{2\lambda K_{i-1}^3\Gamma(3\theta+1)}{3\left(1-\lambda L_{i-1}^{1-1/\theta}K_{i-1}^{1/\theta}\right)^{3\theta}} \\
&\le \left(2^{2\theta+1}\Gamma(2\theta+1) + \frac{2^{3\theta}K_{i-1}\Gamma(3\theta+1)}{3L_{i-1}^{1-1/\theta}K_{i-1}^{1/\theta}}\right)K_{i-1}^2 \\
&= \left(2^{2\theta+1}\Gamma(2\theta+1) + \frac{2^{3\theta}\Gamma(3\theta+1)}{3\log(n/\delta)^{\theta-1}}\right)K_{i-1}^2.
\end{aligned}
$$

□

We can bound

$$
P\left(\bigcup_{k\in[n]}\left\{\sum_{i=1}^k \xi_i \ge x \text{ and } a\sum_{i=1}^k K_{i-1}^2 \le \alpha\sum_{i=1}^k \xi_i + \beta\right\}\right)
$$

by

$$
P\left(\bigcup_{k\in[n]}\left\{\sum_{i=1}^k \widetilde{\xi}_i \ge x \text{ and } a\sum_{i=1}^k K_{i-1}^2 \le \alpha\sum_{i=1}^k \widetilde{\xi}_i + \beta\right\}\right)
$$

and

$$P\left(\bigcup_{i\in[n]}\left\{\xi_i > \log(n/\delta)^\theta K_{i-1}\right\}\right) \leq \sum_{i=1}^{n} P\left(\xi_i > \log(n/\delta)^\theta K_{i-1}\right) \overset{TB2}{\leq} 2\delta$$

completing the proof.

A.9 Full Version and Proof of Theorem 15 and Proof of Theorem 16

Theorem 19 (Non-convex case, full version)**.** *Assume f is L-smooth and that, conditioned on the previous iterates, e_t is centered and $\|e_t\|$ is K-sub-Weibull(θ) with $\theta \geq 1/2$. If $\theta > 1/2$, assume f is ρ-Lipschitz. Let $\delta_1, \delta_2, \delta_3 \in (0,1)$ and $\delta = \max\{\delta_1, \delta_2, \delta_3\}$. Let*

$$c_\theta = (2^{2\theta+1}+2)\Gamma(2\theta+1) + \frac{2^{3\theta}\Gamma(3\theta+1)}{3}, \quad \textit{and}$$
$$c^\star = \frac{\sqrt{f(x_0)-f^\star}}{K\left(4e\theta\log(2/\delta_1)\right)^\theta\sqrt{L\log(T+1)}}.$$

Then, for T iterations of SGD with $\eta_t = c/\sqrt{t+1}$ where $c \leq 1/L$, we have:

if $\theta = 1/2$, then, w.p. $\geq 1-\delta_1-\delta_3$,

$$\frac{1}{\sqrt{T}}\sum_{t=0}^{T-1}\frac{1}{\sqrt{t+1}}\|\nabla f(x_t)\|^2 \leq \frac{4(f(x_0)-f^\star)}{c\sqrt{T}} + \frac{64K^2\log(1/\delta_3)}{\sqrt{T}} + \frac{8eLK^2\log(2/\delta_1)\log(T+1)}{\sqrt{T}/c}$$

and, in particular, if $c^\star \leq 1/L$ and $c = c^\star$, the right-hand side equals

$$\begin{aligned}&\frac{4\sqrt{2eLK^2\log(2/\delta_1)(f(x_0)-f^\star)\log(T+1)}}{\sqrt{T}} + \frac{64K^2\log(1/\delta_3)}{\sqrt{T}}\\ &= O\left(\frac{\sqrt{\log(T)\log(2/\delta)}+\log(1/\delta)}{\sqrt{T}}\right);\end{aligned} \tag{A.5}$$

if $1/2 < \theta \leq 1$, then w.p. $\geq 1-\delta_1-\delta_3$

$$\begin{aligned}\frac{1}{\sqrt{T}}\sum_{t=0}^{T-1}\frac{1}{\sqrt{t+1}}\|\nabla f(x_t)\|^2 \leq \frac{4(f(x_0)-f^\star)}{c\sqrt{T}} &+ \frac{8\max\{(4\theta)^\theta eK\rho, 4(4\theta)^{2\theta}e^2K^2\}\log(1/\delta_3)}{\sqrt{T}}\\ &+ \frac{4LK^2(4e\theta\log(2/\delta_1))^{2\theta}\log(T+1)}{\sqrt{T}/c}\end{aligned} \tag{A.6}$$

and, in particular, if $c^\star \le 1/L$ *and* $c = c^\star$*, the right-hand side equals*

$$\begin{aligned}&\frac{4K(4e\theta\log(2/\delta_1))^\theta\sqrt{2L(f(x_0)-f^\star)\log(T+1)}}{\sqrt{T}} + \frac{8\max\{(4\theta)^\theta eK\rho, 4(4\theta)^{2\theta}e^2K^2\}\log(1/\delta_3)}{\sqrt{T}}\\ &\qquad = O\left(\frac{\sqrt{\log(T)}\log(2/\delta)^\theta + \log(1/\delta)}{\sqrt{T}}\right);\end{aligned} \tag{A.7}$$

if $\theta > 1$*, then w.p.* $\ge 1-\delta_1-\delta_2-\delta_3$*,*

$$\begin{aligned}\frac{1}{\sqrt{T}}\sum_{t=0}^{T-1}\frac{1}{\sqrt{t+1}}\|\nabla f(x_t)\|^2 \le \frac{4(f(x_0)-f^\star)}{c\sqrt{T}} &+ \frac{8\max\{2\log(2T/\delta_2)^{\theta-1}K\rho, 4c_\theta K^2\}\log(1/\delta_3)}{\sqrt{T}}\\ &+ \frac{4LK^2(4e\theta\log(2/\delta_1))^{2\theta}\log(T+1)}{\sqrt{T}/c}\end{aligned} \tag{A.8}$$

and, in particular, for $c^\star \le 1/L$ *and* $c = c^\star$*, the right-hand side equals*

$$\begin{aligned}&\frac{2\sqrt{2}K(4e\theta\log(2/\delta_1))^\theta\sqrt{2L(f(x_0)-f^\star)\log(T+1)}}{\sqrt{T}} + \frac{8\max\{2\log(2T/\delta_2)^{\theta-1}K\rho, 4c_\theta K^2\}\log(1/\delta_3)}{\sqrt{T}}\\ &\qquad = O\left(\frac{\sqrt{\log(T)}\log(2/\delta)^\theta + \log(2T/\delta)^{\theta-1}\log(1/\delta)}{\sqrt{T}}\right).\end{aligned}$$

Proof. We have

$$\sum_{t=0}^{T-1}\eta_t\left(1-\frac{L\eta_t}{2}\right)\|\nabla f(x_t)\|^2 \le f(x_0)-f^\star + \sum_{t=0}^{T-1}\eta_t(1-L\eta_t)\langle\nabla f(x_t), e_t\rangle + \frac{L}{2}\sum_{t=0}^{T-1}\eta_t^2\|e_t\|^2.$$

Using the law of total expectation,

$$\mathbb{E}\left[\exp\left(\left(\frac{\eta_t^2\|e_t\|^2}{\eta_t^2K^2}\right)^{1/2\theta}\right)\right] \le 2.$$

Applying Lemma 8, we have, w.p. $\ge 1-\delta_1$,

$$\begin{aligned}\frac{L}{2}\sum_{t=0}^{T-1}\eta_t^2\|e_t\|^2 &\le \frac{LK^2v(2\theta)\log(2/\delta_1)^{2\theta}}{2}\sum_{t=0}^{T-1}\eta_t^2\\ &= \frac{LK^2c^2v(2\theta)\log(2/\delta_1)^{2\theta}\log(T+1)}{2}.\end{aligned}$$

Applying Proposition 2 with

$$\xi_t = \eta_t(1 - L\eta_t)\langle \nabla f(x_t), e_t\rangle$$

$$K_{t-1} = \eta_t(1 - L\eta_t)K\|\nabla f(x_t)\|$$

$$m_t = \eta_t(1 - L\eta_t)K\rho$$

$$\delta = \delta_2$$

$$\beta = 0$$

$$\lambda = \frac{1}{2\alpha}$$

$$x = 2\alpha \log(1/\delta_3)$$

(where we set $\rho = 1$ if $\theta = 1/2$), we have, for all $\alpha \geq bK\rho c$, w.p. $\geq 1 - 2\delta_2 - \delta_3$,

$$\sum_{t=0}^{T-1} \eta_t(1 - L\eta_t)\langle \nabla f(x_t), e_t\rangle \leq 2\alpha \log(1/\delta_3) + \frac{aK^2}{\alpha}\sum_{t=0}^{T-1} \eta_t^2(1 - L\eta_t)^2\|\nabla f(x_t)\|^2.$$

So, for all $\alpha \geq bK\rho c$, we have, w.p. $\geq 1 - \delta_1 - 2\delta_2 - \delta_3$,

$$\sum_{t=0}^{T-1} \eta_t\nu_t\|\nabla f(x_t)\|^2 \leq f(x_0) - f^\star + 2\alpha \log(1/\delta_3) + \frac{LK^2c^2v(2\theta)\log(2/\delta_1)^{2\theta}\log(T+1)}{2}$$

where

$$\nu_t = 1 - \frac{L\eta_t}{2} - \frac{aK^2}{\alpha}\eta_t(1 - L\eta_t)^2.$$

We want to bound ν_t away from zero. To do so, assume $c \leq \frac{1}{L}$ and $\alpha \geq 4aK^2c$. Then $\nu_t \geq \frac{1}{4}$. Setting

$$\alpha = \max\{bK\rho, 4aK^2\}c$$

and plugging in a and b completes the proof. □

To prove Theorem 16, we need the following lemma.

Lemma 15. *Let $Z = t \in [T]$ with probability p_t where $\sum_{t=1}^T p_t = 1$. Let $Z_1, \ldots, Z_n$ be independent copies of Z. Let $Y = (Z_1, \ldots, Z_n)$. Let X be an $\mathbb{R}_+^T$-valued random variable independent of Z. Then*

$$P\left(\min_{t\in Y} X_t > e\eta\right) \leq \exp(-n) + P\left(\sum_{t=1}^T p_tX_t > \eta\right) \ \forall \eta > 0.$$

Proof. First, letting $\eta > 0$ and $x \in \mathbb{R}_+^T$,

$$\begin{aligned} P\left(\min_{t\in Y} x_t > \eta\right) &= P\left(\bigcap_{i=1}^{n}\{x_{Z_i} > \eta\}\right) \\ &\overset{(i)}{=} \prod_{i=1}^{n} P\left(x_{Z_i} > \eta\right) \\ &= P\left(x_Z > \eta\right)^n \\ &\overset{(ii)}{\leq} \left(\frac{1}{\eta}\mathbb{E}\left[x_Z\right]\right)^n \\ &= \left(\frac{1}{\eta}\sum_{t=1}^{T} p_t x_t\right)^n, \end{aligned}$$

where (i) follows by the independence of the Z_i and (ii) follows by Markov's inequality since x_Z is non-negative almost surely. Next, define

$$\begin{aligned} A &= \left\{ x \in \mathbb{R}_+^T \;\middle|\; \sum_{t=1}^{T} p_t x_t \leq \eta \right\} \\ B &= \left\{ (x,y) \in \mathbb{R}_+^T \times [T]^n \;\middle|\; x_{y_i} > e\eta \;\forall i \in [n] \right\}. \end{aligned}$$

Observe,

$$\begin{aligned} P\left((X,Y) \in B\right) &\overset{(i)}{=} P\left(X \in A, (X,Y) \in B\right) + P\left(X \in A^c, (X,Y) \in B\right) \\ &\overset{(ii)}{=} \int_A P\left((x,Y) \in B\right)\mu(dx) + \int_{A^c} P\left((x,Y) \in B\right)\mu(dx) \\ &\leq \int_A \left(\frac{1}{e\eta}\sum_{t=1}^{T} p_t x_t\right)^n \mu(dx) + \int_{A^c} \mu(dx) \\ &\leq \exp(-n)\int_A \mu(dx) + P\left(X \in A^c\right) \\ &\leq \exp(-n) + P\left(\sum_{t=1}^{T} p_t X_t > \eta\right) \end{aligned}$$

where (i) follows from the law of total probability and (ii) follows from Thm. 20.3 of [173] since X and Y are independent. □

Remark 10. *Under the conditions of Lemma 15, [135] uses that* $\mathbb{E}\left[X_Z\right] = \mathbb{E}\left[\sum_{t=1}^{T} p_t X_t\right]$. *Lemma 15 essentially extends this to high probability.*

Now, to prove Theorem 16, we apply Lemma 15 with $X_t = \|\nabla f(x_t)\|^2$ and

$$p_t = \frac{1/\sqrt{t+1}}{\sum_{t=0}^{T-1} 1/\sqrt{t+1}};$$

and use that

$$\frac{\sum_{t=0}^{T-1} 1/\sqrt{t+1}}{\sqrt{T}} \geq 1.$$

A.10 Non-convex Projected SGD

Consider $x_{t+1} = P(x_t - \eta_t g_t) = P(x_t - \eta_t(\nabla f(x_t) - e_t))$. Define

$$G_t = \frac{x_t - P(x_t - \eta_t \nabla f(x_t))}{\eta_t}$$
$$E_t = \frac{x_{t+1} - P(x_t - \eta_t \nabla f(x_t))}{\eta_t}.$$

Note that if $P = I$, then $G_t = \nabla f(x_t)$ and $E_t = e_t$. Moreover, $x_t = Px_t$ and $x_{t+1} = Px_{t+1}$ so, by the non-expansiveness of P, $\|G_t\| \leq \|\nabla f(x_t)\|$ and $\|E_t\| \leq \|e_t\|$. We can get even tighter bounds using the second prox theorem [12, Thm. 6.39]: $\|G_t\|^2 \leq \langle \nabla f(x_t), G_t \rangle$ and $\langle G_t, E_t \rangle \leq \langle \nabla f(x_t), E_t \rangle$.

It is easy to come up with an example where $\|\nabla f(x_t)\|$ does not go to zero, so we would like to bound $\|G_t\|$ instead. We start as usual:

$$f(x_{t+1}) \leq f(x_t) + \langle \nabla f(x_t), x_{t+1} - x_t \rangle + \frac{L}{2}\|x_{t+1} - x_t\|^2.$$

Focusing on the norm term,

$$\frac{L}{2}\|x_{t+1} - x_t\|^2 = \frac{L\eta_t^2}{2}\|G_t\|^2 - L\eta_t^2 \langle G_t, E_t \rangle + \frac{L\eta_t^2}{2}\|E_t\|^2.$$

Focusing on the inner product term,

$$\begin{aligned}
\langle \nabla f(x_t), x_{t+1} - x_t \rangle &= \eta_t \langle \nabla f(x_t), E_t - G_t \rangle \\
&= \eta_t \langle \nabla f(x_t), E_t \rangle - \eta_t \langle \nabla f(x_t), G_t \rangle \\
&\not\leq \eta_t \langle G_t, E_t \rangle - \eta_t \|G_t\|^2.
\end{aligned}$$

Unfortuantely, we cannot procede any further. [176] is able to get around this but at the cost of getting $\sum_{t=0}^{T-1} \eta_t \|e_t\|^2 = O(\sqrt{T})$ instead of $\sum_{t=0}^{T-1} \eta_t^2 \|e_t\|^2 = O(\log(T))$. To mitigate this, they require an *increasing* batch-size. [177] is able to remove this requirement, but only for non-convex projected SVRG *not* non-convex projected SGD. Thus, we leave the analysis of non-convex projected SGD as an open problem.