Understanding and Overcoming Optimization Barriers in Non-convex and Non-smooth Machine Learning

by

Khashayar Gatmiry

S.B., Sharif University of Technology (2019)
S.M., Massachusetts Institute of Technology (2022)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

DOCTORAL OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2025

Authored by:	Khashayar Gatmiry Department of Electrical Engineering and Computer Science August 15, 2025
Certified by:	Stefanie Jegelka Professor of Electrical Engineering and Computer Science, Thesis Supervisor
Certified by:	Jonathan A. Kelner Professor of Applied Mathematics, Thesis Supervisor
Accepted by:	Leslie A. Kolodziejski Professor of Electrical Engineering and Computer Science Chair, Department Committee on Graduate Students

THESIS COMMITTEE

Advisors

Stefanie Jegelka
Professor of Electrical Engineering and Computer Science
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Jonathan A. Kelner
Professor of Applied Mathematics
Department of Mathematics
Massachusetts Institute of Technology

Thesis Readers

Constantinos Daskalakis
Professor of Electrical Engineering and Computer Science
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology

Understanding and Overcoming Optimization Barriers in Non-convex and Non-smooth Machine Learning

by

Khashayar Gatmiry

Submitted to the Department of Electrical Engineering and Computer Science
on August 15, 2025 in partial fulfillment of the requirements for the degree of

DOCTORAL OF PHILOSOPHY

ABSTRACT

At their core, our machine learning systems are trained by solving an optimization problem, where the goal is to minimize a predefined objective function by adjusting model parameters based on the data. Despite the wealth of structure and prior knowledge present in the data and feedback, our training methods remain relatively simple and independent of this structure. In spite of, or perhaps because of, this simplicity, these methods are often lacking in theoretical guarantees. To design machine learning algorithms that are less data-hungry while ensuring theoretical guarantees on both computational efficiency and output validity, it is essential to better understand and leverage the rich structure within the learning setup and the data distribution, e.g. by altering the geometry of the solution space or adjusting the objective function to induce a more effective learning procedure. This approach moves beyond classical algorithm design, which focuses primarily on handling worst-case instances. This thesis investigates the optimization landscape of central learning problems and develops geometric and analytic schemes adapted to their structure, leading to algorithms with superior computational and statistical performance. In addition, it seeks to advance our mathematical understanding of the principles underlying the success of deep learning.

Thesis supervisor: Stefanie Jegelka

Title: Professor of Electrical Engineering and Computer Science

Thesis supervisor: Jonathan A. Kelner

Title: Professor of Applied Mathematics

Acknowledgments

I am deeply grateful to my parents, whose unwavering love, encouragement, and sacrifices have been the foundation of all my endeavors. Their belief in me have shaped not only my work but also the person I have become. I also wish to express my deepest appreciation to my advisors, Stefanie and Jon, for their invaluable guidance, mentorship, and inspiration throughout my PhD. Their insight, generosity, and unwavering support have profoundly shaped my research and academic growth.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A signi cant portion of AI's current success relies on deep learning which has shown striking and counterintuitive learning capabilities in the face of challenging multi-modal data distributions, by tackling a highly non-convex optimization problem. Although this empirical success shows that our current training algorithms can exploit the structure in data to some extent, these systems still have major issues in reliability and safety, which is mainly due to our lack of a systematic understanding of how deep learning works. In this thesis, we start with studying and resolving important optimization challenges in well-de ned learning problems. In parallel to optimization, we will also study sampling problems in machine learning, including sampling with di usion models. Finally we will focus on obtaining a better mathematical understanding of the generalization ability and the empirical phenomena observed in neural networks and their modern architectures transformers and di usion models, and their training optimization landscape. In particular, we will devise new methods to characterize the properties of convergence points of gradient-based optimization algorithms in notoriously di cult-to-analyze non-convex landscapes, which arise in training neural networks.

1.1 Optimization

A fundamental idea in optimization for minimizing an objective function f is to obtain a sequence of points $\hat{x}_i \bullet_{i=1}^{n}$ in the domain of f by iteratively computing x_i from x_{i-1} using the local information of f around x_{i-1}, including the derivatives of f around x_{i-1}. Due to the computational cost of computing higher derivatives, in machine learning most algorithms only use the gradient, i.e. the rst derivative, and sometimes they also exploit the hessian which is the matrix of second order derivatives. For this reason, such rst and second order algorithms will be the focus of this thesis.

The simplest optimization algorithm is gradient descent, where x_i is obtained from x_{i-1} by taking a step toward the gradient at x_{i-1}, namely $x_i \quad x_{i-1} \quad \copyright f \hat{} x_{i-1} \bullet$, with step size . This step can be realized as the minimizer of the linear approximation of f around x_{i-1}, given by $f_{\hat{} x_{i-1} \bullet} \hat{} x \bullet \quad f \hat{} x_{i-1} \bullet \quad \grave{} \copyright f \hat{} x_{i-1} \bullet ; x \quad x_{i-1}$ e within the Euclidean ball of radius around x_{i-1}. This

trick enables us to optimize a linear function at every step instead of f, which is a simpler task. However, note that as we get far from x_{i-1}, the error of the linear approximation grows, so we only allow x_i to go at most $\quad$ unit in Euclidean distance far from x_{i-1} to minimize $f(x_{i-1})$. The key point to grasp here is that the choice of Euclidean ball for the restricting set is not necessary. In fact, a more general strategy is to choose ellipsoids instead of balls. For example, if we are in the plane, one might restrict the minimization of the linear approximation to an ellipsoid which allows a step of two units in the vertical direction but only a step of one unit in the horizontal direction. Note that the choice of ellipsoids could be different for each x_i; in particular, every location in the domain of f can be assigned to a unique ellipsoid. If the choice of such ellipsoids is smooth in the domain of f, then it induces a differentiable manifold M with a unique geometry. Then, minimizing the linear approximation of f at every x_i restricted to the ellipsoid of x_i results in a unique generalization of gradient-descent algorithm based on the geometry of M. A close relative of this family of generalized gradient descents is the family of mirror descent algorithms. For a given strongly convex regularizer or barrier $\quad$ on the domain of f, mirror descent updates x_{i-1} via the gradient $\nabla f(x_{i-1})$ after applying the mirror map $\nabla \quad \mathbb{R}^d \quad \mathbb{R}^d$:

$$\nabla(x_i) \quad \nabla(x_{i-1}) \quad \nabla f(x_{i-1}): \tag{1.1}$$

For sufficiently small step size, mirror descent roughly takes the same step as generalized gradient descent under the geometry given by the ellipsoids that correspond to the hessian of $\quad$. This geometry corresponds to a differentiable manifold whose metric is given by the hessian of $\quad$ in the standard coordinate system.

In optimization, the behavior of the derivatives of f plays an important role in the convergence speed. The simplest assumption on the derivatives that enables a fast convergence speed to the minimizer of f is the smoothness property, which restrict the amount of change the gradient of $\nabla f(x)$ can sustain by changing x. Even without smoothness, under a weaker Lipschitz assumption, for a given objective f and its domain X, it turns out that the average point of the sequence of iterates $(x_i)_{i=1}^n$ for gradient descent and more generally mirror descent still converges to the minimizer of f. For mirror descent, the Lipschitz assumption can be generalized to a bound on the size of the gradients of f in the geometry of $\quad$ around each point in X.

In this setting, The choice of the regularizer $\quad$ with a suitable geometry has an important role in the speed of convergence of mirror descent in minimizing f. It turns out that the convergence speed depends on two major factors that corresponds to this geometry: (1) The range of $\quad$ on this domain, and (2) the length of mirror descent steps when measured in the geometry of $\quad$. Note that range of $\quad$ on the domain is closely related to diameter of the domain in the geometry of $\quad$. For example, one can see that for $X \subseteq Y \subseteq \mathbb{R}^2$, the range of $\quad$ is within a constant factor of the square of the Euclidean diameter. Having a dependence on the diameter of D is intuitively expected as the algorithm is searching in X for the minimizer of f. While taking longer steps enables us to traverse the domain faster, it is riskier as the objective f drastically change compared to its linear approximation once we get far from x_{i-1}. It turns out that the length of our steps in the geometry of $\quad$ has a direct linear effect in the speed of convergence for mirror descent. Therefore, for a given f and domain X, one needs to design a regularizer $\quad$, which balances properties (1) and (2).

The optimization setting discussed above, namely optimizing a function f over domain X is known as o ine optimization. Another important class of optimization problems in machine learning are online, in which at each time $1 \le t \le T$ the learner's action is penalized by a time-dependent loss function $f_t(x)$, where T is called the time horizon. Of particular importance in the online setting is the notion of regret, which is the di erence between the overall incured loss by the learner over time and the smallest possible loss achievable by a xed action x. More precisely, regret is de ned as

$$\mathrm{reg}(x_1, \ldots, x_T) = \sum_{t=1}^{T} f_t(x_t) - \min_{x \in X} \sum_{t=1}^{T} f_t(x).$$

The goal in online optimization is to desgin an algorithm that achieves a sublinear (in T) regret. It turns out that even in the online world, gradient descent and more generally mirror descent are still e ective towards this goal, by using the gradient information of f_t at time t instead of a xed objective as in o ine optimization. In particular, under mild assumptions, the regret of mirror descent could be bounded by the same quantities (1) and (2) above, namely the range of regularizer, and the length of the steps of mirror descent in the geometry of , multiplied by $\sqrt{T}$, which is a sublinear factor in the time horizon.

In this thesis, we design an algorithm that substitutes the need for a projection or linear optimization oracle at each round by the cost of a few (constant) number of linear system solves [1] when f_t's are convex and X is a polytope. Additionally, we improved the regret by shaving o a dimension factor or a condition number factor from the previous best algorithms. To obtain these results for online optimization, in parallel to the interior point methods in o ine optimization, we constructed regularizers for mirror descent that satis es properties (1) and (2) and is also self-concordant.

Self-concordance is a condition that roughly means the local geometry induced by changes slowly in location, and is essential in breaking the previous running time for online convex optimization over polytopes. While Lipschitzness and Smoothness were conditions can be interpreted as boundedness conditions on rst and second derivatives of, conventional self-concordance bound the third derivative of by its second derivative. Note that the computational complexity of computing mirror descent steps matters is the overall runtime, which in turn depends on the choice of the regularizer. This is where the self-concordance becomes useful; since the local geometry of changes slowly, mirror descent's ith step does not change drastically from the $i-1$th step. Hence, one can accelerate the computation of the ith step by using the $i-1$th step as a warm start, which has already been computed in the previous round. Moreover, the notion of self-concordance of leads to an induced geometry by the Hessian of which squeezes more and more as one gets close to the boundaries of X; this forces mirror descent to take smaller and smaller steps measured in the Euclidean distance, since as we mentioned, mirror descent is actually take x size steps in the geometry of ; Crucially, this avoids the algorithm from ever stepping outside of X, which liberates the algorithm for the need of projecting back onto X after every step as mirror descent generally does.

On the one hand, the geometry of has to adapt to the shape of X around each point in the domain, and the gradients of the objective f need to satisfy properties (1) and (2) above;

This might require picking a more complicated R which results in computationally more expensive mirror descent steps as each step involves computing ∇R. Balancing these two forces is essential in our approach in Section 5 in obtaining an online algorithm with both low regret and fast running time.

Self-concordance and Interior Point Theory. Steps of mirror descent are closely related to the Newton steps in interior point methods for minimizing a linear objective in specific domains such as in polytopes, which corresponds to solving a Linear Programming (LP). Our fastest known algorithm for solving LPs are famously based on interior point theory [2]. There, one picks a barrier function R inside the polytope which is self-concordant, and forms an auxilliary optimization problem by adding R to the linear objective to turn the optimization over a non-smooth domain to have a smoother behavior. At each round, one then slightly changes the auxiliary loss by increasing the coefficient of the original linear objective compared to the barrier function, and takes a Newton step to go to the proximity of the new optimal solution. This is similar to the mirror descent step for online linear optimization, except that in online optimization we observe a new linear loss at every round. In section 3, we will illustrate how one can use a self-concordant barrier to obtain faster online optimization algorithms which avoids the need of projections onto the domain set.

Exp-concave optimization. The idea of mirror descent is also useful in tackling the class of online exp-concave optimization, in which the online loss f_t at each round is exp-concave. My work uses mirror descent with a suitable regularizer to improve the regret and running time for online exp-concave optimization. See [3] for more details.

Achieving the optimal minimax regret. As we will see, the regret bounds that we obtain in online convex optimization heavily depends the shape of the action domain X as well as the loss functions f_t that we at every step, and in particular the shape of their gradeints $\nabla f_t(x)$. While most online convex optimization algorithms that we have achieve regret $O(\sqrt{T})$ in the time horizon which is optimal, the rate is not necessarily optimal in other parameters such as in dimension. A curious question is, given a pair of action set X and gradient set C which characterizes the shapes of the gradeints $\nabla f_t(x)$ for all $1 \le t \le T$, whether we can design an algorithm that can acheive the minimax optimal regret albeit with potential compromise on computational efficiency. We will answer this question in Section 2 when C and X are symmetric convex sets, by designing a suitable regularizer for the pair (X, C). It turns out even though this regularizer does not have a closed form, we can design an efficient separation oracle to compute it as a pre-processing step. While our online algorithm is fully polynomial time, the pre-processing and memory are exponential in the dimension of the space.

Partial information Another setting of online optimization or learning that is of importance is the case when the player does not receive full information about the history of the losses in the previous rounds. Note that this is different from the bandit setting in which the learner observes each loss function only partially but does not miss any information from its history of observations. In this model, the player at time t plays over the simplex and can

observe the loss function on a subset of the previous rounds. We form a graph S by defining a node for each time step t, $1 \le t \le T$, and we draw an out-going edge from the nodes whose loss function is observable at time t to node t. This framework subsumes many important cases such as Batched learning, Learning with delayed feedback, Learning with bounded recall. If S was a clique, we could simply use mirror descent with the entropy regularizer over the simplex to obtain an optimal regret. In Section 4, we propose an algorithm based on partitioning S into fractional cliques and running mirror descent in each clique, then combining the resulting action vectors over the fractional cliques. We further prove a novel lower bound which shows the regret bound obtained by this approach is tight, at least in the case when S is transitive.

1.2 Sampling

As we discussed, regularizers or barrier functions are extremely useful in optimization for creating a suitable geometry on the space for taking the optimization steps, and potentially transform a non-smooth problem into a smooth one. It turns out that barriers are not only useful for optimization, but also sampling. Sampling and optimization are interrelated. For example, in order to roughly optimize function f, one can instead sample from the Gibbs distribution e^{f} with potential f. In generative AI usually we hope to sample from complex data distributions, e.g. distribution over image or sequences of tokens.

In parallel to the golden condition in optimization that leads to our algorithmic capability is convexity of the objective, an exact rendering of this condition for sampling translates into the concavity of the log density (or log-concavity). However, researchers have found milder assumptions on the target density e^{f} which still leads to novel sampling algorithms for those targets. Such milder conditions are often characterized as functional inequalities where each side of the inequality involves the integration of a rather arbitrary function or its derivatives with respect to e^{f}. Famous known functional inequalities that lead to fast algorithms are the Poincare and log-sobolev inequalities. We will talk about these inequalities as we need in every section.

An important class of algorithms for sampling are the class of Markov Chain Monte Carlo (MCMC) methods, in which one defines a markov chain whose stationary distribution is the one that we wish to sample from. Then, one starts from an initial distribution and run the steps of the markov chain until the distribution of the random point converges to the desired one. Markov chain can either be in discrete time or continuous time.

Langevin Diffusion. An important class of continuous-time markov chains are Langevin diffusions, whose trajectory behavior can be captured by a stochastic differential equation. As an example, for sampling from the Gibbs distribution e^{f}, one can use the standard Langevin diffusion given by the following SDE:

$$dX_t = \nabla f(X_t)\,dt + dB_t. \tag{1.2}$$

While the Langevin diffusion converges to the desired distribution e^{f}, it does not immediately lead to a runnable algorithm as it is in continuous time. To get an algorithm, however, one

can discretize the SDE as

$$X_{t+1} \leftarrow X_t - \eta \nabla f(X_t) + B_t, \tag{1.3}$$

where $B_t \sim N(0, I)$ is an independent normal variable at each time. The recovered algorithm from this discretization is known as the Unadjusted Langevin Diffusion. Note that the update equation of this algorithm looks similar to gradient descent, except that we also add Gaussian noise at each step. While the continuous time Langevin Diffusion has the correct stationary distribution e^{-f}, this discretized version has a bias and does not exactly converge to the right distribution as time goes to infinity. However, we can still use it as an efficient sampler as long as the bias is not too large. Another approach that is standard is to adjust the discretized algorithm by adding a metropolis filter to remove the bias. While this approach is effective in correcting the bias, the rejection probability induced by the metropolis filter often gets large fast with the step size η. Hence, picking a large step size leads to large rejection probability, which slows down the algorithm. On the other hand, picking a small step size could also result in slow mixing time. Because of the growth of the rejection probability issue, sometimes the unadjusted algorithm is preferred.

Hamiltonian Monte Carlo. Another way to resolve the rejection issue is to use a different markov chain based on the Hamiltonian Monte Carlo (HMC). The HMC markov chain is defined based on a second order differential equation. Intuitively, if one think of the state of the markov chain as a particle, then HMC equation for sampling from e^{-f} defines the gradient of f (log density) as an accelerating force on the particle:

$$\frac{dX_t}{dt} = V_t, \tag{1.4}$$

$$\frac{d}{dt}V_t = -\nabla f(X_t). \tag{1.5}$$

Then, picking the initial velocity as a random Gaussian, HMC equation 1.5 defines a reversible markov chain without the need for a metropolis filter. HMC can more generally be defined on a differentiable manifold, in which case the movement of the particle under the force field is with respect to the distance on the manifold. We will define this more general version, known as Riemannian HMC (RHMC) rigorously in section 5 using the differentiation operator known as "covariant derivative" on the manifold. In particular, geometric HMC provides a framework for sampling from constraint sets $X \subset \mathbb{R}^d$ by considering a barrier function inside X and running HMC on the geometry induced by the Hessian of. For example, if a unit distance according to the geometry of squeeze close to the boundary of X, then RHMC naturally travels shorter in the direction of the boundary (measure in the noermal Euclidean distance). In particular, if is self-concordant and goes to infinity on the boundary of X, RHMC never exits the domain X. This geometry induced by was also discussed and studied in the previous section for optimization with mirror descent.

It turns out that HMC-based samplers converge to the stationary distribution very fast in practice and this speed highly depends on the choice of the geometry on induced by In particular, if the geometry of is adapted to the local geometry around each in the domain X, and if it is aligned with the measure e^{-f} inside X, then RHMC mixes fast.

As we elaborate , the properties within a barrier function that are e ective for obtaining a fast mixing time with RHMC are di erent from those for the purpose of optimization. In this thesis, we will see how to design barrier functions that are useful for sampling as well as optimization. First, note that while optimization is a search problem for which the range of the barrier function or the diameter of X in the geometry induced by is important, sampling only cares about the regions of X where the e ^{f} is concentrated on. Rouhgly, in sampling we care about how much measure with respect to the Gibbs distribution e ^{f} do we gain as we take a unit step in the geometry of. This concept is rigorously captured by the notion of isoperimetry on the manifold in X obtained from . On the other hand, to have a fast sampling algorithm with RHMC under , we also care about the stability of the ellipsoids of induced on the domain. For optimization, we argued that the standard notion of self-concordance captures this notion very well. However, in sampling, it turns out that we need di erent, more sophisticated form of self-concordance so that RHMC mixes fast. For one condition which is also shared with optimization, we want the ellipsoid of at each point $p > X$ to be a good approximator of the local geometry inside X around p. We call this property well-roundedness and will de ne it rigorously.

Mirror Langevin The source of randomness in the stochastic di erential equation that corresponds to the Langevin Di usion (equation 1.3) comes from the Brownian motion term dB_t, which is spherically symmetric. This is because in every time interval, the brownian motion term adds up an isotropic Gaussian noise. However, to obtain a faster mixing time, one might want to use a non-spherically-symmetric noise in which the covariance of the noise depends on the location of the particle. It turns out one can generalize the Langevin Di usion to the Riemannian case on a manifold so that the covariance of the noise aligns with the local geometry around each point on the manifold. In other words, the isotropic noise variable in the standard Langevin di usion would expand or squeeze in di erent directions propostional to the local distances on the manifold. Similar to RHMC, this is useful for sampling from a distribution constrained to a set X b R^{d} by using the induced geometry from a barrier function on X that goes to in nity on the boundary. A curious question is now what properties of the barrier results in a faster mixing time for the discretization of the Riemannian Langevin di usion (RLD). In Section 6 we answer this question by driving a new self-concordance property which we call second-order self-concordance for the barrier function de ned on X b R^{d}, that is suitable for sampling with RLD. Combining it with a log-sobolev inequality for the target density e ^{f}, we obtain a fast mixing for the discretization of RLD for sampling from e ^{f} constrained to the set X.

1.3 Di usion models

A major problem with MCMC techniques such as algorithms based on Langevin di usion or HMC is that such samplers are inherently built for log-concave sampling, or for sampling from distributions that are close to log-concave. Another important issue with these samplers is that unless the target density satis es a strong functional inequality such as the log-sobolev inequality, they require a warm start, meaning the starting distribution needs to have a bounded density ratio with the target distribution, otherwise their mixing time could be very

slow. Di usion models can potentially solve both of these issues by smoothing out the target distribution and start the sampling from easier smoothed distributions, then gradually turn them into a sample from the original distribution. This idea is similar to the what interior point methods do in optimization, using a smooth barrier function de ned on the domain set. With di usion models, this smoothing procedure is done by convolving the target distribution with the Gaussian density. Note that convolving the distribution with a very high-variance isotropic Gaussian results in is almost a Gaussian with the same variance, which is close to being log-concave and easy to sample from. However, as we gradually decrease the level of noise (speci ed by the Gaussian variance) in the Gaussian convolution, the distribution could get harder to sample from, especially if the original distribution is highly multi-model, aka far from being log-concave. However, the hope here is that even though these distributions get harder and more complex as we decrease the level of noise, the task of transforming a sample from a higher noise distribution to a slightly lower noisy one is still easy algorithmically. This seems to be the case in practice as di usions have proven to be highly successful in generative AI, e.g. in sampling from highly multi-modal distributions of images in computer vision or proteins in bio-informatics. One way that di usion models do these transformations is via discretizing a stochastic process that is the reverse of the Ornstein-Uhlenbeck (OU) process. The OU process is a standard Langevin di usion as described in equation 1.2 that transforms an arbitrary distribution to Gaussian, given by the following SDE:

$$dX_t \quad X_t dt \quad dB_t:$$

If we start this process from X_0 distributed according to our target distribution p_0 that we hope to sample from, then at some large time T of the OU process, the distribution of X_T, denoted by p_T, is almost Gaussian (as the limit in the in nity is exactly Gaussian). Now if we start the reverse stochastic process from Gaussian, it would roughly transform it to our target p_0. Therefore, discretizing this reverse process provides us with an algorithm for iteratively transforming Gaussian noise to a sample from the target. It turns out the reverse process can also be described by a stochastic di erential equation, albeit one where the bias (dt) term is time dependent:

$$dY_t \quad \copyright \ln p_{T\ t}\hat{}Y_t\bullet \quad dW_t; \qquad 0 B t B T;$$

where p_t is the density of X_t at time t (based on the OU process or the forward process) and W_t is another Brownian motion. Hence, to simulate the backward process, we need to estimate the gradient of the log density, $\log p_t$'s, known as the score function, for all t's between 0 and T, where each is the convolution of di erent amounts of Gaussian noise with the original distribution.

In Section 7, we will see the underlying mechanisms behind the success of di usion models in sampling from multi-modal distributions by showing how one can provably sample from continuous mixtures of Gaussians using di usion models. A continuous mixture of Gaussians with k-components is simply the convolution of a base-measure whose support can be covered by k unit Euclidean balls with isotropic Gaussians. In particular, we show that this approach leads to a running time that is quasi-polynomial in the dimension and the number of clusters of the mixture, which almost matches the state of the art algorithm by [4] for sampling from

Isotropic discrete mixtures of Gaussians. This is while our approach covers the substantially larger class of continuous mixtures of Gaussians that, despite the class of discrete mixtures of Gaussians, is a non-parameteric class.

Guidance. The science of Machine learning is beyond just tting a model to a dataset to obtain predictive capability on test data, but also includes exploring ways that we can incorporate our prior knowledge in the learning procedure. This could be in form of a prior knowledge about the data distribution: for example, the fact that we observe the distribution over images has rich structures in the space of pixels that are local has inspired us to train convolutional architectures of neural networks. But what if this prior knowledge is given to us in the form of another machine learning system? Consider the senario where we have a mixed dataset of pictures of cats and dogs, and we hope to generate a new image of a cat which is almost independent of the dataset (otherwise we could just return a cat picture in our dataset!). In addition, we are given a classi er that has been trained to distinguish between cats and dogs. One approach for this goal is that we can rst use the classi er to separate the pictures of cats and dogs, then we can train a di usion model on the pictures of cats that can generate a fresh image. But this might not work so well if we our mixed dataset is small. The question is wether there is a better way to exploit this classi er for sampling a cat? Researchers have empirically discovered a method for this task with the di usion framework, called guidance, that turns out to be e ective in practice: recall that for simulating the backward process in di usion models one requires an estimate of the score function Given an estimate for the score of the mixture distribution of cats and dogs and a classi er for distinguishing cats and dogs, it turns out one can design an estimate for the score for the conditional density of distribution of cats, which then enables us to sample from cats by running the di usion model for the conditional distribution. The idea is to approximate the gradient of the likelihood function for cats, then add it to the score of the mixed distribution, which according to the Bayes rule would give the score for the conditional distribution. In particular, if we model the image space with variable x and the cat or dog label by discrete variable $y \in \{1, \ldots\}$, then we have

$$\nabla_x \log p(x \mid y) = \nabla_x \log p(y \mid x) + \nabla_x \log p(x).$$

Moreover, researchers found that by consider a more general linear combination such as

$$\nabla_x \log p(y \mid x) + (1 - w)\nabla_x \log p(x),$$

with an arbitrary guidance parameter $w \geq 0$, that increasing $w > 0$ to some level would enhance the image generation quality. However, it is not clear where this improvement comes from. Before our work, it was believed that using $w > 0$ results in sampling from the tilted distribution $p^{y,w}(x) = p(x)p(y \mid x)^{w-1}$, for a xed label y. We discredit this false belief and show that the e ect of increasing w is in fact much more intricate: increasing w above zero results in gradually pushing the mass of $p(x \mid y)$ toward part of the support of $p(x \mid y)$ which is as far as possible from the support of $p(x \mid y)$ for the other label $y \neq y$. We rigorously prove this for the one-dimensional case and further show this e ect empirically in higher dimensions. We refer the reader to [5] for more details on our approach.

1.4 Deep Learning

A signi cant portion of AI's current success relies on deep learning which has shown striking and counterintuitive learning capabilities in the face of challenging multi-modal data distributions, by tackling a highly non-convex optimization problem. Although this empirical success shows that our current training algorithms can exploit the structure in data to some extent, these systems still have major issues in reliability and safety, which is mainly due to our lack of a systematic understanding of how training for deep learning works and how these systems generalize well on test data. Given the limitations of traditional generalization bounds based on uniform convergence for deep networks, researchers have increasingly recognized that understanding the remarkable generalization of neural networks requires a deeper exploration of training algorithm's trajectory. In this thesis, we take a closer look to the trajectory of the training algorithms and the critical points of training over-parameterized neural networks. This training can be regarded as a stochastic optimization problem in which one obtains estimates for the loss function and its gradient. In particular, if we denote the parameterized neural net function with parameters W by $f_W \quad R^d \quad R^m$, and given loss function $` \quad R \quad C \quad R$ with C being the output (label) space, the objective function on W that we wish to minimize is de ned as

$$L\hat{}W\bullet < E_{\hat{}x;y\bullet}.``\hat{}f_W\hat{}x\bullet;y\bullet;$$

where the expectation is over the population distribution on the pair of input and labels $\hat{}x;y\bullet$. In reality, we do not have direct access to this distribution but only observe the dataset $\tilde{\hat{}}\,x_i;y_i\bullet\bullet_{i\ 1}^{m}, \hat{}x_i;y_i\bullet \quad D$ and form the empirical loss based on that:

$$L\hat{}W\bullet < \frac{1}{m}\,Q_{i\ 1}^{m}\ ``\hat{}f_W\hat{}x_i\bullet;y_i\bullet:$$

It is easy to check that $L\hat{}W\bullet$ is an unbiased estimate of $L\hat{}W\bullet$, and its gradient $\copyright L\hat{}W\bullet$ is an unbiased estimate of $\copyright L\hat{}W\bullet$. Because of this reason, we can assume that training neural networks is a stochastic optimization problem over the population loss L for which we have access to an unbiased estimate of its gradients.

Unfortunately, this problem is not convex even for very simple neural networks. Therefore, most of the mathematical tools that have been developed in convex optimization does not directly apply to deep learning. We saw in the case of sampling from the class of continuous mixtures of Gaussians how di usion models overcome the non-concavity of the log density by rst sampling from simpler distributions that are convolved with Gaussian. There, we use a regression over the basis of low-degree piecewise polynomials to obtain an estimate for the score function. In practice people use a neural networks to learn the score function and it often works quite well. But the problem of learning the score with a neural network is again a highly non-convex stochastic optimization problem.

In this thesis, we seek to create new mathematical tools to better understand the optimization landscape of neural networks. Our methodology for this study can be divided into two categories: (1) benign landscape analysis and (2) implicit bias.

1.4.1 Benign Landscape Analysis

The idea is to show that the landscape is benign for optimization even though it is not convex. The benign aspect at high level can be characterized as if there is a gap in loss to the optimal minimum value, then there is a direction in which SGD can progress. There are two general benign conditions with which we work (1) The first condition is that if we are at a point that is considerably suboptimal, then its gradient should be large, in which case SGD can make progress in the direction of the gradient. Conditions of this type are sometimes denoted as gradient dominance. The second type of condition, which is more complicated, is the nonexistence of spurious saddle points. In high level, this means (2) if we are at a point with suboptimality gap which is close to being a saddle point (i.e. it has a small gradient norm), then there should be a negative enough direction in the hessian. In the second case, it turns out the noise in SGD enables the algorithm to escape the non-spurious saddle point. Although these conditions were recognized before in nonconvex optimization, the main contribution of this thesis is to explore the cases where the landscape of neural networks satisfies these conditions. We start by characterizing a novel generalization bound for SGD-trained three-layer neural networks beyond the Neural Tangent Kernel (NTK).

Generalization of neural nets beyond kernels The neural tangent kernel is a regime of training neural networks in which the weights remain close to the random initialization. In this regime, the output is normalized with the square root of the number of neurons. When the width of the network large enough, the weights remain in a neighborhood of the initialized weights in which the linear approximation of the neural net remains accurate, and as a result, training the network essentially acts as fitting a kernel to data. This can be seen as embedding the data in a high-dimensional feature space and then fitting a linear model in that space. While this leads to novel generalization bounds, neural nets are known to be capable of extracting intermediate features from data adaptively and further building upon those features to describe the output, which cannot be described by a fixed kernel. In this thesis, we see how a three-layer neural net trained by a variant of SGD can extract data-adaptive features with its first layer, then use them in the next layers to fit the labels. As a result, we show a generalization bound in this regime which is as small as the best kernel in a family of kernels corresponding to the learned intermediate features in the first layer. For more details on our approach, please see [6].

Transformers learn how to do in-context learning We have seen the emergence of large language models (LLMs) with even more surprising abilities in reasoning and few-shot learning; provided a few examples from a new learning task in the prompt, the pre-trained model is able to learn the task and generalize to the test query without any fine-tuning [7, 8]. This ability of LLMs is conjectured to be due to the fact that transformers are able to internally implement iterative algorithms such as gradient descent [9, 10]. An important question here is *How does a transformer learn to internally implement gradient descent during training?* While answering this question in general for the LLMs that are pre-trained on the internet seems elusive at the moment, when conducting theoretical studies, it is often best to start with simplified cases. Following this strategy, this thesis focuses on understanding the optimization landscape of in-context learning for linear regression. In particular, we assume

that we train the transformer on random instances of linear regression, and we want to see how it becomes capable of solving a new instance as its input. In particular, we study this problem for loop transformers in which the weights of the attention blocks between di erent layers are shared. In this case, it turns out that the loss landscape satis es a form of gradient dominance condition.

1.4.2 Implicit bias

It might be di cult or impossible to show a benign feature of the landscape of population loss for arbitrary weights, or we might not have enough data, resulting in our study of the population loss being inaccurate and the gradients having high variance. A perhaps di erent approach is to focus on the empirical loss, despite the population loss, and see where the training algorithm tends to converge to, as that is the loss that are directly optimizing over. Since we are considering the overparameterized case, which means that we have more parameters in the neural net than we have data, it is often the case that the network can perfectly t the training data. However, while there are possibly an in nite number of networks that can perfectly t the data, we see that in practice the network, trained by SGD, performs well on test data. This is attributed to the fact that SGD does not pick just any weights that can t the data but chooses one that is simple enough to be able to generalize on test data. The tendency of SGD to pick zero-loss solutions that can generalize well on test data is known as the implicit bias phenomenon, whose understanding requires us to study the trajectory of the training algorithm.

Recent ndings have shown thatafter reaching zero loss,a speci c variant of SGD, i.e. label-noise SGD, is locally biased toward minimizing the trace of Hessian of the loss during training [11]. However, this provides limited insight into the*global* implicit bias of label-noise SGD. In this thesis, we study this implicit bias by characterizing the global minimizer of the trace of Hessian on the manifold of neural networks with zero loss, demonstrating that this global minimizer can generalize well to test data. Furthermore, a key open question in the deep learning community has been:Can we prove that (label-noise) SGD converges to this global implicit bias? Here, we also show that label-noise SGD converges to the global minimizer of trace of Hessian on the zero-loss manifold, for a broad class of nonlinear two-layer neural networks. Notably, our analysis takes a new di erential-geometric approach toward a non-convex optimization problems on a sub-manifold of the Euclidean space.

Chapter 2

Computing Optimal Regularizers for Online Linear Optimization

Abstract

Follow-the-Regularized-Leader (FTRL) algorithms are a popular class of learning algorithms for online linear optimization (OLO) that guarantee sub-linear regret. However, the choice of regularizer can significantly impact dimension-dependent factors in the regret bound. We present an algorithm that takes as input convex and symmetric action sets and loss sets for a specific OLO instance, and outputs a regularizer such that running FTRL with this regularizer guarantees regret within a universal constant factor of the best possible regret bound. In particular, for any choice of (convex, symmetric) action set and loss set we prove that there exists an instantiation of FTRL that achieves regret within a constant factor of the best possible learning algorithm, strengthening the universality result of Srebro et al., 2011.

Our algorithm requires preprocessing time and space exponential in the dimension d of the OLO instance, but can be run efficiently online assuming a membership and linear optimization oracle for the action and loss sets, respectively (and is fully polynomial time for the case of constant dimension d). We complement this with a lower bound showing that even deciding whether a given regularizer is 1-strongly-convex with respect to a given norm is NP-hard.

2.1 Introduction

Online Linear Optimization (OLO) is one of the most fundamental problems in the theory of online learning. Here, a learner must repeatedly (for T rounds) select an action x_t from some bounded convex action set X. Simultaneously, an adversary selects a linear loss function ℓ_t from a bounded convex loss set L, and the learner receives loss $\ell_t(x_t) = \langle \ell_t, x_t \rangle$. The learner would like to minimize their total loss, and more specifically minimize their *regret*: the gap between their total loss and the loss of the best fixed action $x^\ddagger \in X$ in hindsight.

By choosing the action set X and loss set L appropriately, online linear optimization captures many other learning-theoretic problems of interest. For example, when $X = \Delta_d$ (distributions over $\{1, 2, \ldots, d\}$) and $L = [0, 1]^d$, this captures the classical problem of learning with experts. Similarly, when the loss set L is the ℓ_2 unit ball, this variant of OLO is the core subproblem involved in online convex optimization (specifically, of a Lipschitz function with domain X). Even more generally, the works of [12] and [13] demonstrate how to reduce the problems of linear Φ-regret minimization (including swap regret minimization) and Blackwell approachability to different instances of OLO. These problems in turn have many applications extending past learning theory, from designing algorithms for computing correlated equilibria in repeated games, to producing calibrated forecasts, to constructing classifiers satisfying a variety of fairness criteria [14–16].

For this reason, it is an extremely relevant problem to understand the best possible regret bounds achievable for different instances of OLO. Here, the state-of-the-art leaves something to be desired. It is well-known that learning algorithms such as Follow-The-Regularized-Leader (FTRL) achieve regret that scales with $O(\sqrt{T})$, and that this dependence on T is tight. However, the dependence of the optimal regret on the sets X and L (e.g., how the constant factor in the above regret bound depends on the dimension d of these sets) is in general poorly understood.

Moreover, FTRL is not a single algorithm, but a family of algorithms parametrized by a convex function $f : X \to \mathbb{R}$ called the *regularizer*. The actual regret bounds achieved by FTRL can vary greatly depending how the choice of regularizer interacts with the geometry of X and L. For example, running FTRL with the quadratic regularizer results in an $O(\sqrt{dT})$ regret algorithm for the learning with experts problem; however, running FTRL with the negative entropy regularizer results in an algorithm with a tight $O(\sqrt{T \log d})$ regret bound, with an exponential improvement in dimension over the quadratic choice of regularizer. On the other hand, there exist other instances (choices of X and L) where the quadratic regularizer is optimal. Understanding what the optimal choice of regularizer is for a given instance of OLO is a major open problem.

2.1.1 Our contributions

For any action set X and loss set L, the optimal possible regret bound (as T goes to infinity) scales as $\mathrm{Rate}(X, L) \cdot \sqrt{T} \pm o(\sqrt{T})$, for some constant $\mathrm{Rate}(X, L)$. Our goal in this paper is to design learning algorithms which approximately achieve this optimal regret bound. Specifically, we want to algorithmically construct learning algorithms with worst-case regret at most $C \cdot \mathrm{Rate}(X, L) \cdot \sqrt{T}$ for some universal constant C that holds for any choice of action set and loss set in any dimension. For technical reasons, we restrict our attention in the following results to action sets X and loss sets L that are centrally symmetric – it is an interesting open direction to extend these results to fully general choices of X and L.

We begin by showing that the optimal regret bound is achieved by some instantiation of Follow-The-Regularized-Leader. We do so by extending earlier work of [17] who, by analyzing the martingale types of Banach spaces, demonstrated that there is always an instance of FTRL which achieves regret $O(\mathrm{Rate}(X, L) \cdot \log T \cdot \sqrt{T})$. In Theorem 38, we show that a more

careful analysis of these martingale types allows us to remove this $\log T$ factor and prove that some variant of FTRL is within a universal constant of optimal.

Although the above argument proves the existence of a near-optimal instance of FTRL, the regularizer for this FTRL instance is not computable. The reason is that the de nition of the barrier in [17] is a supremum of an in nite sum over in nite sequences of martingales that are de ned by the action and loss set. Even worse, it is not clear if any truncation of this sum can approximate the value of the supremum with any given accuracy. In the remainder of the paper we study the following algorithmic question: given sets X and L (e.g., via oracle access), how can we compute a regularizer for these sets that guarantees the optimal regret up to universal constants when used with FTRL? Ultimately, we provide an algorithm that takes as input X and L (via standard oracle access to both sets), runs in time $\exp(O(d^2 \log d))$, and outputs a regularizer f with the property that the worst-case regret of FTRL with f is at most a universal constant times $\mathrm{Rate}(X, L)\sqrt{T}$ (Theorem 1). Note that this regularizer is not an approximation of the regularizer of [17], but rather is the solution of an optimization problem that we design.

The main technical ingredient in this algorithm is a new method for optimizing over the set of convex functions that are α-strongly convex with respect to a given norm. This is important for the above problem because one can show that for any regularizer f, the regret of running FTRL with that regularizer is bounded by $O(\sqrt{D/\alpha}\sqrt{T})$ if the range of f over X (the maximum value of f minus the minimum value of f) is at most D and if f is α-strongly-convex with respect to the norm induced by the dual set of the loss set L. We can show that this regret-bound is constant-factor-optimal for the near-optimal variant of FTRL in Theorem 38, and hence it su ces to try to minimize D/α over all convex functions f.

To do this, we rst show that we can approximate any smooth convex function h as a maximum of several quasi-quadratic functions: quadratic functions g_{x_0} centered at some point x_0 with a small cubic term which guarantee that that the contribution of g_{x_0} to the Hessian of h decays far from x_0. Note that these are not just approximations of the values of h, but also the gradients and Hessians of h; in particular, if the original function was α-strongly-convex with respect to some norm, our approximation will be similarly strongly-convex.

By restricting our quasi-quadratic functions to be centered at points belonging to a (large but) nite discretization of X, we demonstrate how to optimize over this set of approximations by solving a large convex program with variables for the values, gradients, and Hessians of the quasi-quadratic functions at each point in the discretization. Solving this convex program involves implementing a separation oracle to verify whether a speci c approximation is α-strongly-convex with respect to an arbitrary norm.

As stated earlier, this approach takes time exponential in the dimension of the action and loss sets (although is completely independent of the time horizon T), and thus e cient for constant dimension d). We complement this with a lower bound showing that even verifying whether a regularizer f is α-strongly-convex at a speci c point $x \in X$ requires exponentially many oracle queries to L.

2.2 Related Work

Applications of Online Linear Optimization. The problem of Online Linear Optimization (and its generalization, Online Convex Optimization) are central problems in the eld of online learning we refer the reader to [18] for a general-purpose introduction. Traditionally OLO is studied in the case where the action sets and loss sets are unit balls in a standard norm (e.g. the ℓ_1, ℓ_2, or ℓ_∞ norms). However, there are many motivating settings where we wish to minimize regret with less standard sets. Several authors [19 22] study variants of OLO where the action space has some combinatorial structure for example, $\mathcal{X}$ could be the spanning tree polytope, or the polytope formed by all s-t paths in a graph. Minimizing external regret in extensive form games one standard method for computing coarse correlated equilibria [23] involves solving an instance of OLO where $\mathcal{X}$ is the sequence form polytope. Finally, as mentioned earlier, the work of [13] and [12] allows us to translate any instance of Blackwell approachability or α-regret minimziation to a (usually non-standard) instance of OLO.

Follow-The-Regularized-Leader and Mirror Descent. The Follow-The-Regularized-Leader algorithm can be thought of as a form of mirror descent, a family of rst-order optimization algorithms that generalize gradient descent by using arbitrary distance-generating functions. Originally, mirror descent was proposed by [24] as an o ine optimization algorithm with ℓ_p norm constraints and ℓ_q Lipschitz assumptions, and was shown to have minimax optimal query complexity. [25] studied the optimality of mirror descent for online linear optimization when the action and loss vectors are in the unit ball of two Banach spaces dual to each other, proving the existence of a regularizer for mirror descent that almost achieves the minimax rate under an adaptive adversary. Later, [17] extended this approach to cases where the action and loss vectors come from independent convex balls in primal and dual Banach spaces. The existence of such strongly convex regularizers is also linked to the Burkholder method introduced by [26] for more general online learning problems. In particular, the authors propose that given an online learning instance and a target regret bound, the existence of a Burkholder function for that instance guarantees the existence of a prediction strategy that achieves the desired regret. Notably, taking the dual of this Burkholder function for the online linear optimization (OLO) problem results in a strongly convex regularizer that can be used e ectively with FTRL [26]. We survey additional related work in Appendix C.1.

2.3 Preliminaries

2.3.1 Online linear optimization

We begin by de ning the problem ofonline linear optimization (OLO). In this problem, every round t (for a total of T rounds) the learner must pick an action x_t from a convex action set $\mathcal{X} \subseteq \mathbb{R}^d$. The adversary then picks a loss vector ℓ_t from a convex loss set $\mathcal{L}$, after which the learner su ers loss $\ell(x_t, \ell_t)$ and observes the loss vector ℓ_t. The learner would like to minimize their total loss, and more speci cally minimize their total regret: the gap between their loss and the loss of the best action in hindsight. Formally, given a sequence of learner actions

$x = (x_1, x_2, \ldots, x_T)$ and losses $(\ell_1, \ell_2, \ldots, \ell_T)$, the regret of the learner is given by

$$\mathrm{Reg}(x, \ell) = \sum_{t=1}^{T} \ell(x_t, \ell_t) - \sum_{t=1}^{T} \min_{x^* \in X} \ell(x^*, \ell_t).$$

The learner chooses their actions according to some learning algorithm A, which can be thought of as a function A mapping a sequence of losses $(\ell_1, \ell_2, \ldots, \ell_T)$ to a sequence of actions $x = (x_1, x_2, \ldots, x_T)$ in such a way that x_t depends only on the history of losses $\ell_1, \ell_2, \ldots, \ell_{t-1}$ until round t. We define the T-round regret $\mathrm{Reg}_T(A)$ to be the worst-case regret suffered by algorithm A against an adversarially chosen sequence of losses, i.e., $\mathrm{Reg}_T(A) = \sup_{\ell \in L^T} \mathrm{Reg}(A(\ell), \ell)$.

One of the fundamental results in online learning is that there exist algorithms A that guarantee $O(\sqrt{T})$ regret (e.g., online gradient descent), which is the best possible dependency one can hope for in terms of T. However, the optimal scaling factor in front of the $\sqrt{T}$ depends on the geometry of the action and loss sets X and L and is the primary focus of interest in this paper. To this end, define $\mathrm{Rate}(A) = \limsup_{T \to \infty} \frac{1}{\sqrt{T}} \mathrm{Reg}_T(A)$ to be the worst-case scaling factor achieved by the algorithm A, and $\mathrm{Rate}(X, L) = \inf_A \mathrm{Rate}(A)$ to be the best possible scaling factor achieved by any algorithm for this action set and loss set. Our goal is to understand how to approximate $\mathrm{Rate}(X, L)$ and design corresponding optimal algorithms for any choice of action set and loss set.

2.3.2 Regularizers and Follow-The-Regularized-Leader

One of the most popular classes of learning algorithms for online linear optimization is the class of follow-the-regularized-leader algorithms. Follow-The-Regularized-Leader (FTRL) is an algorithm parameterized by a convex function $f : X \to \mathbb{R}$ (the regularizer) and a learning rate $\eta > 0$ (which we will generally set equal to $1/\sqrt{T}$). At round t, it plays the action x_t given by

$$x_t = \arg\min_{x \in X} \left\{ \frac{1}{\eta} f(x) + \sum_{s=1}^{t-1} \ell(x, \ell_s) \right\}. \tag{2.1}$$

Intuitively, FTRL always plays an action that is approximately the best response to the current empirical loss (with the regularizer preventing this action from overfitting too rapidly to the actions of the adversary). The class of FTRL algorithms contains many popular algorithms for special cases of online linear optimization, including online gradient descent and multiplicative weights.

It can be shown that as long as f is strongly convex, FTRL will incur $O(\sqrt{T})$ regret and thus have non-infinite rate; however, the value of $\mathrm{Rate}(X, L)$ can depend significantly on the choice of f. For example, when $X = B_d$ and $L = (0, 1)^d$ (the classic setting for learning from experts), it is known that:

- If we use the quadratic regularizer $f(x) = \|x\|_2^2$, the resulting rate of the FTRL algorithm is $\mathrm{Rate}(A) = \sqrt{d}$. (This corresponds to running online gradient descent).

- If we use the negative entropy regularizer $f(x) = \sum_i x_i \log x_i$, the resulting rate of the FTRL algorithm is $\text{Rate}(A) = \sqrt{\log d}$. (This corresponds to running multiplicative weights / Hedge).

We will soon see that the optimal rate is achieved by some instantiation of FTRL (Theorem 43), and therefore much of our focus will be on computing a suitable regularizer f for a given pair of action set and loss set (X, L). To this end, it is useful to understand the guarantees the standard analysis of FTRL grants us for a specific choice of regularizer. Before we can state these, we will need to introduce some terminology regarding convex sets and their associated norms.

First, we will make the standard assumption in convex optimization that all of our convex sets are bounded and contain an open ball. In particular, we have the following assumption:

Assumption 1. We assume the action and loss sets are symmetric[1]. We further assume they both include a ball of radius r and are included in a ball of radius R: $B(0, r) \subseteq X, L \subseteq B(0, R)$.

The symmetry assumption allows us to define norms corresponding to X and L. In general, the norm provided by a bounded symmetric convex set C is defined as follows:

Definition 1. Given a bounded symmetric convex subset $C \subseteq \mathbb{R}^d$, we define the natural norm $\|\cdot\|_C$ corresponding to C as

$$\{v \in \mathbb{R}^d, \|v\|_C < \inf\{\alpha \geq 0, \frac{v}{\alpha} \in C\}\}. \tag{2.2}$$

It is easy to check that $\|\cdot\|_C$ defined in equation 2.2 is a norm [27].

Given a symmetric convex set C, we can also define a norm on linear functionals over C by constructing the appropriate dual convex set.

Definition 2. Given a symmetric convex set $C \subseteq \mathbb{R}^d$, the dual set C^* is defined as $C^* = \{x \in \mathbb{R}^d \mid \forall y \in C, \langle x, y\rangle \leq 1\}$. Note that if C is symmetric, bounded, and full-dimensional, the dual set C^* is symmetric, bounded, and full-dimensional. The dual norm $\|\cdot\|_{C^*}$ is the norm corresponding to the dual set.

We also need to define the notion of strong convexity with respect to an arbitrary norm $\|\cdot\|_C$:

Definition 3. A convex function $f : X \to \mathbb{R}$ is strongly-convex with respect to norm $\|\cdot\|_C$ if for every $x, y \in X$ and every sub-gradient g of f at x: $f(y) - f(x) - \langle y - x, g\rangle \geq \frac{1}{2}\|y - x\|_C^2$.

Now we are ready to state the standard regret bound for FTRL with regularizer f. As we can see, the regret bound depends on both the strong convexity of f with respect to the dual norm of L, and the range of f over X:

Fact 1. [Theorem 5.2 in [18]] Let FTRL(f) be the FTRL algorithm initialized with regularizer f and learning rate $\eta = 1/\sqrt{T}$. If $0 \leq f(x) \leq D^2$ for all $x \in X$ and f is σ-strongly-convex with respect to L^* on X (see Definition 3), then $\text{Reg}(\text{FTRL}(f)) \leq O(D/\sigma\sqrt{T})$.

2.3.3 Convex Optimization and Oracles

We will in general assume that we have oracle access (i.e., access to membership oracles, separation oracles, linear optimization oracles) to the sets X and L. For a more comprehensive

[1] A convex set $S \subseteq \mathbb{R}^d$ is (centrally) symmetric if $x \in S$ implies that $-x \in S$.

de nition of these oracles, see Appendix C.4.

2.4 Main Result and Overview

Our main contribution is to propose an algorithm for computing a regularizer g such that running FTRL with g achieves the optimal regret of $O(\mathrm{Rate}(X; L, \bar{T}))$ for the online linear optimization problem, as de ned in Section 2.3.1. In particular, we state our main result in the following theorem.

Theorem 1 (Algorithmic optimal online linear optimization). Given access to a linear optimization oracle for L, which can minimize any linear function $\langle c, x\rangle$ over L up to accuracy $\varepsilon_{\mathrm{lin}}$ in time $\mathrm{Lin}(L, \varepsilon_{\mathrm{lin}})$, there is a cutting-plane algorithm that runs in time $O\left(\frac{dR}{\varepsilon}\right)^{O(d^2)} \cdot \mathrm{Lin}(L, \Omega(\frac{\varepsilon}{dR})) \cdot d$ and calculates a regularizer g which satis es

1. $\sup_{x \in X} g(x) \le O(\mathrm{Rate}(X; L, \bar{T}))^2$,
2. g is 1-strongly convex w.r.t. $\|\cdot\|_{L,c}$.

Furthermore, given access to a membership oracle for X and the regularizer g (which can be precomputed and summarized via an $\exp(O(d^2))$-dimensional vector as described in Section 2.8) there is a cutting-plane algorithm that runs FTRL with regularizer g with running time $O(d^2 \ln^{O(1)}(d) \cdot dRT)$ per round and which guarantees regret $O(\mathrm{Rate}(X; L, \bar{T}))$.

The starting point of our proof of the above theorem is to demonstrate the existence of a regularizer that enables FTRL to achieve the optimal minimax regret, up to a constant factor.

Theorem 2. There exists a regularizer f_0 so that running FTRL with f_0 yields a regret of $\mathrm{Reg}(x, \ell) \le O(\mathrm{Rate}(X; L, \bar{T}))$.

We prove Theorem 2 in Appendix C.2, where we eliminate the additional $\log(T)$ factor from the regret analysis of the regularizer in [17], proving that it achieves the optimal regret bound of $O(\mathrm{Rate}(X; L, \bar{T}))$, up to universal constants. This improvement is made possible by a novel analytic estimate for the norm growth of certain martingales. In particular, we prove in Theorem 38 that the regularizer from [17] can be chosen to be 1-strongly convex with respect to $\|\cdot\|_{L,c}$ while being bounded by $O(\mathrm{Rate}(X; L, \bar{T}))^2$ on the domain X. Theorem 2 then follows from Theorem 38 and Fact 1.

This allows us to restrict our attention to the problem of nding the optimal regularizer over X which is 1-strongly-convex with respect to $\|\cdot\|_{L,c}$. To e ectively do this optimization, it is important that the resulting regularizer has not only bounded values, but also bounded gradients. Note that this is not a priori achieved by the regularizers guaranteed to exist by Theorem 38, and in fact several optimal regularizers used in practice (e.g. the negative entropy regularizer) do have unbounded gradients. Nonetheless, in Section 2.5 and Appendix C.3, we demonstrate how to use Gaussian smoothing to obtain a new regularizer that (1) achieves the same optimal regret when used in FTRL, and (2) has smooth derivatives (Theorem 3).

Our next step is to show that we can e ectively optimize over the space of smooth convex functions de ned over X. To do so, we show that given a near-optimal smooth regularizer f, we can approximate it using quasi-quadratic functions such that the resulting regularizer $\tilde{f}$

remains (1) $\tilde{\sigma}$-2 strongly convex with respect to $\|\cdot\|_{Y,Y,L_c}$, and (2) bounded by $O(\text{Rate}(X,L)\cdot\epsilon^{-2}\cdot)$ on X. Notably, the set of quasi-quadratic functions (with a discretized set of centers) is finite-dimensional, and so the optimal regularizer can be encoded by a finite-dimensional vector $\tilde{\Gamma}$. We carry this out in Section 2.6.

Finally, in Section C.5, we demonstrate how to optimize over this set by writing an explicit convex program such that $\tilde{f}$ is a feasible solution to this program, but also such that any feasible solution so that any feasible solution from this set yields a regularizer $g(\cdot)$ with near optimal regret. Solving this convex program can be done via standard cutting-plane methods, except for one of the constraints that involves checking whether a candidate regularizer is σ-strongly-convex with respect to $\|\cdot\|_{Y,Y,L_c}$. In Section C.6, we demonstrate how to construct a separation oracle for this constraint, and finally establish the existence of this algorithm.

As seen in Theorem 1, computing and storing this optimal regularizer takes time that is exponential in the dimension of the problem. In Section??, we establish a lower bound based on the result of [28] that even checking the strong convexity of the Euclidean norm squared regularizer with respect to $\|\cdot\|_{Y,Y,L_c}$ requires an exponential number of queries in the dimension.

2.5 A Smooth Optimal Regularizer

While Theorem 2 promises the existence of an ideal regularizer which achieves the optimal rate, this regularizer is not effective. To obtain a computable regularizer, our goal will be to search over a parametric family of functions. We use f_0 to prove the existence of a good regularizer in our search space. More accurately, we plan to accomplish this by approximating the regularizer f_0 as a maximum over several local approximations to the regularizer at a finite, discrete set S of N points in X. By Fact 1, doing this requires showing that there exists such an approximation that (1) preserves the strong convexity of f_0, (2) is bounded by $O(\text{Rate}(X,L)\cdot\epsilon^{-2}\cdot)$ on X, ensuring that the resulting regret matches the bound in Theorem 2.

To preserve strong convexity, local first-order approximations of f_0 are insufficient as they flatten the function's curvature (i.e., this would result in a piecewise-linear approximation of f_0 once we take the maximum over our local approximations). Therefore, we must produce local second-order approximations of f_0. In order for these approximations to remain close to f_0 locally around each $x_i \in S$, we need f_0 to have a Lipschitz-continuous Hessian. However, the regularizer from [17] does not necessarily even possess smooth derivatives. We side-step this issue by proposing an alternative regularizer that not only achieves the optimal rate of $O(\text{Rate}(X,L)\cdot\bar{T}\cdot)$ but also features smooth derivatives 2. This regularizer can then be approximated by our strategy.

Theorem 3 (Existence of a smooth regularizer) There exists a regularizer f so that running FTRL with f has regret bound $\text{Reg}(\text{FTRL}(f))\le B\,O(\text{Rate}(X,L)\cdot\bar{T}\cdot)$. In addition, the derivatives of f are bounded as $\|D^k f(x)[v,\dots,v]\| \le O(\text{Rate}(X,L)\cdot\epsilon^{2-k}\frac{d^{k-4}}{r^k}\cdot)$.

We construct the smooth regularizer f of Theorem 3 by adding Gaussian noise to f_0, and prove that (1) the Gaussian smoothing does not impact performance; running mirror descent with f achieves the same regret bound as running mirror descent with f_0, and (2) the derivatives of f are sufficiently smooth due to the Gaussian smoothing (see Theorem 39). For

the remainder of the paper we will let $\tilde{f}$ denote this smooth, optimal regularizer (in contrast to f_0, which we will use to denote the original possibly non-smooth regularizer guaranteed by Theorem 2).

2.6 Approximating the Smooth Regularizer

Now that we can focus on smooth regularizers, we can explore their approximation using the previously outlined approach; from the derivative bound in Theorem 3, it is easy to show that our smooth regularizer $\tilde{f}$ has an L-Lipschitz Hessian (i.e., satisfies $\|\nabla^2 \tilde{f}(x_0) - \nabla^2 \tilde{f}(x_1)\| \le L\|x_0 - x_1\|$ for all $x_0, x_1 \in \mathbb{R}^d$). From this property, we can show that the quadratic approximation of $\tilde{f}$ around x_0 remains close to $\tilde{f}$, at least locally around x_0.

However, since our final approximation of $\tilde{f}$ is constructed as the maximum over a collection of local approximators, the quadratic approximation at x_0 may be overshadowed by those centered at other points, potentially leading to significant deviations from $\tilde{f}$ in the vicinity of x_0. To address this, we must ensure that each local approximator is designed to maintain its dominance, at least within its own neighborhood. We accomplish this by adding a norm-cubic decay term to the quadratic approximation. This ensures that for any two points $x_0, x_1 \in \mathbb{R}^d$, if they are sufficiently distant, the local approximator around x_0 does not dominate the one at x_1 within its own neighborhood, and vice versa. Consequently, our final local approximation of $\tilde{f}$ around a point $x_0 \in \mathbb{R}^d$ takes the following form:

$$f_{x_0}(x) = \tilde{f}(x_0) + \langle \nabla \tilde{f}(x_0), x - x_0 \rangle + \frac{1}{2}(x - x_0)^\top \nabla^2 \tilde{f}(x_0)(x - x_0) - \frac{L}{3}\|x - x_0\|^3. \tag{2.3}$$

We refer to a function of the form in equation 2.3 as quasi-quadratic, centered at x_0. The intuition for this approximation is that the norm cubic term adds a decay to the Hessian of the function as we move away from x_0; this decay guarantees that $f_{x_0}(x)$ is always a lower bound for $\tilde{f}$, and in particular can be estimated by $\tilde{f}$ from above and below with margin $\frac{L}{6}\|x - x_0\|^3$. We show this in Lemma 1. On the other hand, this decay is slow enough so that from the L-Hessian smoothness of $\tilde{f}$ we can prove that the Hessian of the approximation remains almost the same as the Hessian of $\tilde{f}$ (at least locally around x_0) and therefore the strong convexity property can be preserved (see Lemma 5). In particular, even though the function in equation 2.3 is not strongly convex for all $x \in \mathbb{R}^d$ (or even all $x \in X$), since it is strongly convex locally, if we choose a sufficiently dense discretization set S, our resulting overall approximate regularizer will still be strongly convex (we discuss this more in Section 2.8).

Lemma 1 (Estimating $\tilde{f}$ by the approximator). We have the following relation between the value of $\tilde{f}$ and f_{x_0}:

$$f_{x_0}(x) - \frac{L}{6}\|x - x_0\|^3 \le \tilde{f}(x) \le f_{x_0}(x) + \frac{L}{2}\|x - x_0\|^3.$$

The proof of Lemma 1 is in Section C.7.1.

Finally, we combine these local approximations for each $x_i \in S$ by taking the maximum over all these functions and defining the piece-wise quasi-quadratic function $\hat{f}(x) = \max_{x_i \in S} f_{x_i}(x)$.

Importantly, while $\tilde{f}$ remains strongly convex and suitably bounded on $\mathcal{X}$, it is also efficiently encoded by $\hat{f}(x_i)$, $\hat{\nabla f}(x_i)$, and $\hat{\nabla^2 f}(x_i)$ at discretized points $S = \{x_i\}_{i=1}^N$, since each $f_{x_i}(x)$ does not use more than zeroth, first, and second order information of f at x_i's. Therefore, we can narrow our search for suitable regularizers from all convex functions $\mathbb{R}^d$ to the selection of the value, gradient, and Hessian of a piece-wise quasi-quadratic function at a finite set of points. In fact, in the next section we write a convex program to minimize the maximum value of these piecewise quasi-quadratic regularizers.

2.7 A Convex Program for Calculating an Ideal Regularizer

In the previous section, we showed that it is possible to approximate f with a set of quasi-quadratic approximators which only depend on the value, gradient, and Hessian of f at a finite set of points $S = \{x_i\}_{i=1}^N$. In this section, we describe how to search the space of such approximators by defining a convex program whose variables are the function's value, gradient and Hessian at S, denoted by $\{r_{x_i}, v_{x_i}, \gamma_{x_i}\}_{i=1}^N$.

Before rigorously defining the program, we first provide some motivation for its definition. In particular, we want the instance $I = \{\hat{r}_{x_i}\}_{i=1}^N, \{\hat{v}_{x_i}\}_{i=1}^N, \{\hat{\gamma}_{x_i}\}_{i=1}^N\}$ where $\hat{r}_{x_i} = \hat{f}(x_i)$, $\hat{v}_{x_i} = \hat{\nabla f}(x_i)$, $\hat{\gamma}_{x_i} = \hat{\nabla^2 f}(x_i)$, corresponding to the smoothed regularizer in Theorem 2, to be a feasible point. On the other hand, for any instance $I = \{\hat{r}, \hat{v}, \gamma\} = \{\{r_{x_i}\}_{i=1}^N, \{v_{x_i}\}_{i=1}^N, \{\gamma_{x_i}\}_{i=1}^N\}$, we can define a regularizer $g_{x_i}^I(\hat{x})$ as

$$g^I(\hat{x}) = \max_{i=1}^N g_{x_i}^I(\hat{x}), \tag{2.4}$$

where imitating the approximation that we derived for f in equation 2.3, $g_{x_i}^I(\hat{x})$ denotes a quasi-quadratic function:

$$g_{x_i}^I(\hat{x}) = r_{x_i} + \langle v_{x_i}, \hat{x} - x_i \rangle + \frac{1}{2}(\hat{x} - x_i)^T \gamma_{x_i}(\hat{x} - x_i) + \frac{L}{6}\|\hat{x} - x_i\|^3. \tag{2.5}$$

With this terminology, it is clear that $f = g^I$. Besides having I as a feasible point of the program, we also want to impose constraints so that for the optimal solution of the program, $I^\ddagger$, the regularizer $g^{I^\ddagger}$ is strongly convex and suitably bounded on $\mathcal{X}$. First, note that from Lemma 59, μ-strong convexity of f with respect to $\|\cdot\|_{L_c}$ is equivalent to the condition

$$v^T \nabla^2 f(x) v \geq \mu \tag{2.6}$$

for all $x \in \mathcal{X}$ and $v \in \mathbb{L}$. Hence, we also add the condition $v^T \gamma_{x_i} v \geq \mu, \forall v \in \mathbb{L}$ to the program. While this condition asserts strong convexity of g^I for all feasible instances I at the discretization points, it does not guarantee strong convexity elsewhere. The reason is that the approximator in equation 2.10 loses the strongly convexity property for points far from x_i. Therefore, in order to guarantee strong convexity for g^I everywhere, we need to make sure that at any point $x \in \mathcal{X}$, the maximum in equation F.5 is attained by a function $g_{x_i}^I$ where x_i is sufficiently close to x. Building on this observation, we introduce the concept of "locality" for an arbitrary instance I:

Definition 4. We define an instance $I = (r, v, \Sigma)$ as ϵ-local if, for every x, $\|x_{i^*(x)} - x\| = O(\epsilon)$ where $i^*(x) = \arg\max_{i \in [N]} g^{(I)}_{x_i}(x)$.

Note that ϵ-locality is guaranteed for $\tilde{f} = g^{(I^*)}$ by Lemma 1. Specifically, if there is a point $x_i \in S$ such that $\|x_i - x\| = O(\epsilon)$, then according to Lemma 1, the point $x_{i^*(x)}$, where $g_{x_{i^*(x)}}$ attains its maximum in equation F.5 at x, must also be within a distance of $O(\epsilon)$ from x. To ensure that the maximum equation F.5 is attained at an $x_{i^*(x)}$ that is close to x, we enforce a slightly relaxed version of the lower bound from Lemma 1 on $g^{(I)}$ at the discretization points:

$$g^{(I)}_{x_i}(x_j) + \frac{15L}{96}\|x_j - x_i\|^2 \le r_{x_j}, \quad i, j = 1, \dots, N. \tag{2.7}$$

As noted in Lemma 1, $\tilde{f}$ satisfies the inequality $f_{x_0}(x) + \frac{L}{6}\|x - x_0\|^2 \le f(x)$. The reason we apply a slightly weaker version of this inequality in equation 2.12 will become evident when we design a separation oracle for the feasibility set of the convex program. At a high level, this condition ensures that not only is $\tilde{f}$ a feasible instance for our program, but that a small neighborhood around it also remains feasible. As we will see, even after enforcing the condition in equation 2.12, an arbitrary feasible instance I does not achieve $O(\epsilon)$-locality like $\tilde{f}$. Instead, we can only prove that it is $O(\epsilon^{1/3})$-local (see Lemma 4). The reason is that equation 2.12 is only enforced at the discretization points, whereas $\tilde{f}$ satisfies it for any $x \in X$ as shown in Lemma 1.

Finally, we aim to minimize the maximum value of $g^{(I)}$ over X to obtain a suitable regularizer for FTRL. As mentioned earlier, we smooth the theoretical regularizer f_0 from [17] by adding Gaussian noise, resulting in $\tilde{f}$, which ensures bounded gradients and Hessians. To achieve a similar smoothness condition on the regularizer $g^{(I)}$ that correspond to a feasible instance of our program, we enforce the conditions $\|v_{x_i}\|_\infty \le c_0$ and $\Sigma_{x_i} \preceq c_2 I$ for constants c_0, c_2 (we use the infinity norm instead of the 2-norm to maintain a linear constraint.) With the discretization set $S = \{x_i\}_{i=1}^N$ fixed, the final program is as follows:

$$\text{minimize } r \tag{2.8}$$

$$\begin{aligned}
\text{subject to } & r_{x_i} - \langle v_{x_i}, \delta_{ij} \rangle + \tfrac{1}{2}\delta_{ij}^\top \Sigma_{x_i} \delta_{ij} + \tfrac{17L}{96}\|\delta_{ij}\|^2 \le r_{x_j} && \forall i, j \in [N],\ \delta_{ij} = x_j - x_i \\
& \|v_{x_i}\|_\infty \le c_0 && i \in [N] \\
& \Sigma_{x_i} \preceq c_2 I && \forall i \in [N] \\
& v^\top \Sigma_{x_i} v \ge && \forall v \in L;\ \forall i \in [N] \\
& r \ge r_{x_i} && \forall i \in [N] \\
& r, r_{x_i} \le C_0 && \forall i \in [N].
\end{aligned}$$

Next, to establish the locality property for feasible points of the program, we state in Lemma 4 that for any arbitrary $x \in X$, the maximum in equation F.5 is attained at a discretization point $x_i \in S$ that is not too far from x. Specifically, given that every point in X has a discretization point x_i within a distance of ϵ, we show that the maximum in equation F.5

is achieved by $x_{\hat{l}}$ which is no further than $O(\hat{} \ ^{1\sim3}\bullet)$ from x. Additionally, we prove that the value of $g(\hat{l}\bullet)$ at x is close to $g_{x_i}(\hat{l}\bullet\hat{}x\bullet)$.

Lemma 2 (Convex program feasibility $\Rightarrow$ Locality of regularizer g). Assume that $l(\hat{}r; v; \bullet)$ is feasible for LP 2.13, for $\ satisfying \ B \ _2 \min\left(\frac{\circ L}{\bar{d}c_0}; \frac{\circ L}{c_0 \ \bar{d}c_2^3}; \frac{L}{c_2}; \ \sqrt{\frac{\circ}{c_0 \ \bar{d}}}; \frac{c_0 \ \bar{d}}{c_2}\right)$: If for $x_i; x_j$ and $x > X$ we have $\|x_i \ x\| B$ and $\|x_j \ x\| C \ \check{S}\frac{c_0 \ \bar{d}}{L}\bullet^{1\sim3}$ for some universal constant , then

$$g_{x_i}(\hat{l}\bullet\hat{}x\bullet) \ A \ g_{x_j}(\hat{l}\bullet\hat{}x\bullet) \quad c_0 \ \sqrt{\circ}\bar{d};$$

and if $\|x_j \ x\| B \ \check{S}\frac{\bar{d}c_0}{L}\bullet^{1\sim3}$, then

$$\left|g_{x_j}(\hat{l}\bullet\hat{}x_i\bullet) \ g_{x_j}(\hat{l}\bullet\hat{}x\bullet)\right| B \ _2c_0 \ \sqrt{\circ}\bar{d};$$

for some constant $_2$.

The proof can be found in Section C.7.2. To prove strong convexity of $f(\hat{l}\bullet)$ for a feasible point l, we must first establish the strong convexity of the local approximators $g_{x_i}(\hat{l}\bullet)$, defined in equation 2.10. This is demonstrated in Lemma 5 below. Specifically, we prove that if the quadratic form of the Hessian variable $_{x_i}$ is lower bounded by the norm squared $\|\cdot\|_{L^c}^2$ in all directions, then $g_{x_i}(\hat{l}\bullet\hat{}x\bullet)$ is strongly convex locally around x_i.

Lemma 3. [Local strong convexity of the approximators] Suppose the PSD matrix is such that for all v, $v^{\top} \ v \ C \ \|v\|_{L^c}^2$. Then, the function

$$g(\hat{}x\bullet) \quad r \ `(v; x \ x_0)e^{\frac{1}{2}\hat{}x \ x_0\bullet^{\top}\hat{}x \ x_0\bullet \ \frac{L}{6}\|x \ x_0\|^3}$$

for arbitrary $x_0; v; r; L$ is $\sim$2-strongly convex with respect to $\|\cdot\|_{L^c}$ in the neighborhood $\|x \ x_0\| B \frac{}{2R^2L}$. Consequently, if f is -strongly convex with respect to $\|\cdot\|_L$, then $f_{x_0}(\hat{}x\bullet)$ is $\frac{}{2}$ strongly convex with respect to $\|\cdot\|_L$ for $\|x \ x_0\| B \frac{}{2R^2L}$.

The proof of Lemma 5 can be found in Section C.7.3. Finally, by combining Lemmas 64 and 4, we show that the barrier $g(\hat{l}\bullet)$ constructed from a feasible point of the matrix program has a suitable upper bound on X, satisfying the desired strong convexity. Additionally, we prove that the feasible region can be approximated both from the inside and outside by Euclidean balls, a key property necessary for constructing a separation oracle for the feasible set later.

Theorem 4 (Convex program solution $\Rightarrow$ optimal regularizer). Assume we are given a smooth barrier function $f \ R^d \ R$ with $\|\hat{}x\bullet\| B \ C^2; \ x > X$, which is c_1 Lipschitz, c_2 gradient Lipschitz, L Hessian Lipschitz, and $\|\cdot\|_{L^c}$-strongly convex in X. Additionally, if for every two points in the cover $x_i; x_j > X$ we have $\|x_i \ x_j\| C$, then the convex program in equation 2.13 with $c_0 \ c_1 \ L^{\ 3}; c_2 \ c_2 \ L^{\circ 3}, L \ L, C_0 \ C^2 \ L^{\ 3}$, and discretization parameter $B \ _3 \min\left(\frac{\circ L}{\bar{d}c_1}; \frac{\circ L}{c_1 \ \bar{d}c_2{}^3}; \frac{L}{c_2}; \ \sqrt{\frac{}{c_1 \ \bar{d}}}; \frac{c_1 \ \bar{d}}{c_2}; \frac{\sqrt[3]{\circ}}{512R^6L^2c_1 \ \bar{d}}\right)$ for all sufficiently small constant $_3$ is feasible. Furthermore, the function $g(\hat{l}^{\ddagger}\bullet)$, corresponding to the optimal solution $l^{\ddagger} \ (\hat{}r^{\ddagger}; v^{\ddagger}; \ ^{\ddagger}\bullet)$ is convex and satisfies the following properties:

1. $\left|g(\hat{l}^{\ddagger}\bullet\hat{}x\bullet) \ \right| B C^2 \ _2 \sqrt{\circ}\bar{d}c_0$ for constant $_2$.
2. For any feasible instance $l > P_l$, $g(\hat{l}\bullet\hat{}x\bullet)$ is $\frac{}{2}$ strongly convex with respect to $\|\cdot\|_{L^c}$.

3. $B_L \leq 288\hat{\Gamma} \bullet b P_l \; b B_2 \frac{1}{4} \frac{}{\hat{N} \; 1 \bullet C_0^2 \; N d \% \%_0^2 \; c_2^2 \tilde{Z}} \hat{\Gamma} \bullet$.

The proof of Theorem 5 can be found in Section C.7.4.

2.8 A Convex Program for Calculating an Ideal Regularizer

In the previous section, we showed that it is possible to approximate f with a set of quasi-quadratic approximators which only depend on the value, gradient, and Hessian of f at a finite set of points $S \; \tilde{x}_i \bullet_{i \; 1}^N$. In this section, we describe how to search the space of such approximators by defining a convex program whose variables are the function's value, gradient and Hessian at S, denoted by $\tilde{r}_{x_i}; v_{x_i}; \; _{x_i} \bullet_{i \; 1}^N$.

Before rigorously defining the program, we first provide some motivation for its definition. In particular, we want the instance $\tilde{\Gamma} \; \% \%_0 r_{x_i} \bullet_{i \; 1}^N; \tilde{} v_{x_i} \bullet_{i \; 1}^N; \tilde{} \tilde{} _{x_i} \bullet_{i \; 1}^N \check{Z}$ where $r_{x_i} < f \hat{} x_i \bullet; \forall_{x_i} < \copyright f \hat{} x_i \bullet; \tilde{}_{x_i} < \copyright^2 f \hat{} x_i \bullet$, corresponding to the smoothed regularizer f_Γ in Theorem 2, to be a feasible point. On the other hand, for any instance $I \; \hat{} r; v; \; \bullet \; \tilde{\tilde{}} r_{x_i} \bullet_{i \; 1}^N; \tilde{} v_{x_i} \bullet_{i \; 1}^N; \tilde{} \; _{x_i} \bullet_{i \; 1}^N \bullet$, we can define a regularizer $g_{x_i}^{\hat{I}} \bullet \hat{} x \bullet$ as

$$g^{\hat{I}} \bullet \hat{} x \bullet < \max_{i > N} g_{x_i}^{\hat{I}} \bullet \hat{} x \bullet; \tag{2.9}$$

where imitating the approximation that we derived for f in equation 2.3, $g_{x_i}^{\hat{I}} \bullet \hat{} x \bullet$ denotes a quasi-quadratic function:

$$g_{x_i}^{\hat{I}} \bullet \hat{} x \bullet \; r_{x_i} \; \grave{} v_{x_i}; x \; x_i e \; \tfrac{1}{2} \hat{} x \; x_i \bullet \bar{} \; _{x_i} \hat{} x \; x_i \bullet \; \tfrac{L}{6} Y x \; x_i Y^3 : \tag{2.10}$$

With this terminology, it is clear that $f \tilde{} \; g^{\tilde{\Gamma}}$. Besides having $\tilde{\Gamma}$ as a feasible point of the program, we also want to impose constraints so that for the optimal solution of the program, $I^{\ddagger}$, the regularizer $g^{\hat{I}^{\ddagger}} \bullet$ is strongly convex and suitably bounded on X. First, note that from Lemma 59, $\; -$strong convexity of f with respect to $Y \cdot Y_{L^c}$ is equivalent to the condition

$$v \bar{} \copyright^2 f \hat{} x \bullet v \; C \tag{2.11}$$

for all $x > X$ and $v > L$. Hence, we also add the condition $v \bar{} \; _{x_i} v \; C \; ; \; \vdots \; v > L$ to the program. While this condition asserts strong convexity of $g^{\hat{I}} \bullet$ for all feasible instances I at the discretization points, it does not guarantee strong convexity elsewhere. The reason is that the approximator in equation 2.10 loses the strongly convexity property for points far from x_i. Therefore, in order to guarantee strong convexity for $g^{\hat{I}} \bullet$ everywhere, we need to make sure that at any point $x > X$, the maximum in equation F.5 is attained by a function $g_{x_i}^{\hat{I}} \bullet$ where x_i is sufficiently close to x. Building on this observation, we introduce the concept of *locality* for an arbitrary instance I:

Definition 5. We define an instance $I \; \hat{} r; v; \; \bullet$ as $\; -$local if, for every x, $Z x_{\hat{I}} \hat{} x \bullet \; x Z \; O \hat{} \bullet$ where $\hat{I} \hat{} x \bullet < \arg\max_{i > N} \; g_{x_i}^{\hat{I}} \bullet \hat{} x \bullet$:

Note that -locality is guaranteed for $\tilde{f}$ by Lemma 1. Specifically, if there is a point $x_i \in S$ such that $\|x_i - x\| \le \hat{\delta}$, then according to Lemma 1, the point $x_{\hat{i}(x)}$, where $g_{x_{\hat{i}(x)}}$ attains its maximum in equation F.5 at x, must also be within a distance of $\hat{\delta}$ from x. To ensure that the maximum equation F.5 is attained at an $x_{\hat{i}(x)}$ that is close to x, we enforce a slightly relaxed version of the lower bound from Lemma 1 on $\hat{g}^{(1)}$ at the discretization points:

$$\hat{g}^{(1)}_{x_i}(x_j) \ge \frac{15L}{96}\|x_j - x_i\|^3 \quad \forall i, j = 1, \ldots, N. \tag{2.12}$$

As noted in Lemma 1, $\tilde{f}$ satisfies the inequality $f(x_0) + \frac{L}{6}\|x - x_0\|^3 \ge \tilde{f}(x)$. The reason we apply a slightly weaker version of this inequality in equation 2.12 will become evident when we design a separation oracle for the feasibility set of the convex program. At a high level, this condition ensures that not only is $\tilde{f}$ a feasible instance for our program, but that a small neighborhood around it also remains feasible. As we will see, even after enforcing the condition in equation 2.12, an arbitrary feasible instance h does not achieve $\hat{\delta}$-locality like $\tilde{f}$. Instead, we can only prove that it is $\hat{\delta}^{1/3}$-local (see Lemma 4). The reason is that equation 2.12 is only enforced at the discretization points, whereas $\tilde{f}$ satisfies it for any $x \in X$ as shown in Lemma 1.

Finally, we aim to minimize the maximum value of $\hat{g}^{(1)}$ over X to obtain a suitable regularizer for FTRL. As mentioned earlier, we smooth the theoretical regularizer f_0 from [17] by adding Gaussian noise, resulting in f, which ensures bounded gradients and Hessians. To achieve a similar smoothness condition on the regularizer g, $\hat{g}^{(1)}$ that correspond to a feasible instance of our program, we enforce the conditions $\|v_{x_i}\|_\infty \le c_0$ and $\gamma_{x_i} \ge c_2$ for constants c_0, c_2 (we use the infinity norm instead of the 2-norm to maintain a linear constraint.) With the discretization set $S = \{x_i\}_{i=1}^N$ fixed, the final program is as follows:

$$\text{minimize } r \tag{2.13}$$

$$\begin{aligned}
\text{subject to } & r_{x_i} \ge \langle v_{x_i}, \delta_{ij}\rangle + \tfrac{1}{2}\gamma_{ij}\|x_i - x_j\|^2 + \tfrac{17L}{96}\|\delta_{ij}\|^3 + r_{x_j} && \forall i, j \in N, \delta_{ij} = x_j - x_i \\
& \|v_{x_i}\|_\infty \le c_0 && \forall i \in N \\
& \gamma_{x_i} \ge c_2 && \forall i \in N \\
& \|v_{x_i}\| \le C && \forall i \in N, v \in L, \\
& r \ge r_{x_i} && \forall i \in N \\
& r, r_{x_i} \ge C_0 && \forall i \in N.
\end{aligned}$$

Next, to establish the locality property for feasible points of the program, we state in Lemma 4 that for any arbitrary $x \in X$, the maximum in equation F.5 is attained at a discretization point $x_i \in S$ that is not too far from x. Specifically, given that every point in X has a discretization point x_i within a distance of , we show that the maximum in equation F.5 is achieved by $x_{\hat{i}}$ which is no further than $\hat{\delta}^{1/3}$ from x. Additionally, we prove that the value of $\hat{g}^{(1)}$ at x is close to $g_{x_i}(x)$.

Lemma 4 (Convex program feasibility – Locality of regularizer). Assume that (r, v, τ) is feasible for LP 2.13, for τ satisfying $\tau \le \min\left(\frac{\alpha}{d c_0}, \frac{\alpha}{c_0 d^3}, \frac{1}{c_2}, \sqrt{\frac{\alpha}{c_0 d}}, \frac{c_0 d}{c_2}\right)$. If for x_i, x_j and $x \in \mathcal{X}$ we have $\|x_i - x\| \le \beta$ and $\|x_j - x\| \le \left(\frac{c_0 d}{L}\right)^{1/3}$ for some universal constant γ, then

$$g_{x_i}(x) \le g_{x_j}(x) + c_0 d,$$

and if $\|x_j - x\| \le \left(\frac{\alpha d c_0}{L}\right)^{1/3}$, then

$$|g_{x_j}(x_i) - g_{x_j}(x)| \le \gamma_2 c_0 d,$$

for some constant γ_2.

The proof can be found in Section C.7.2. To prove strong convexity of $f(\cdot)$ for a feasible point τ, we must first establish the strong convexity of the local approximators $g_{x_i}(\cdot)$, defined in equation 2.10. This is demonstrated in Lemma 5 below. Specifically, we prove that if the quadratic form of the Hessian variable $\nabla^2 f(x_i)$ is lower bounded by the norm squared $\|\cdot\|_{L_c}^2$ in all directions, then $g_{x_i}(x)$ is strongly convex locally around x_i.

Lemma 5. [Local strong convexity of the approximators] Suppose the PSD matrix $\nabla^2 f(x)$ is such that for all v, $v^\top \nabla^2 f(x) v \ge \|v\|_{L_c}^2$. Then, the function

$$g(x) = f(x_0) + \langle \nabla f(x_0), x - x_0 \rangle + \frac{1}{2}(x - x_0)^\top \nabla^2 f(x_0)(x - x_0) - \frac{L}{6}\|x - x_0\|^3$$

for arbitrary x_0 is $\frac{1}{2}$-strongly convex with respect to $\|\cdot\|_{L_c}$ in the neighborhood $\|x - x_0\| \le \frac{1}{2R^2 L}$. Consequently, if f is α-strongly convex with respect to $\|\cdot\|_L$, then $f_{x_0}(x)$ is $\frac{\alpha}{2}$ strongly convex with respect to $\|\cdot\|_L$ for $\|x - x_0\| \le \frac{1}{2R^2 L}$.

The proof of Lemma 5 can be found in Section C.7.3. Finally, by combining Lemmas 64 and 4, we show that the barrier $g(\cdot)$ constructed from a feasible point of the matrix program has a suitable upper bound on $\mathcal{X}$, satisfying the desired strong convexity. Additionally, we prove that the feasible region can be approximated both from the inside and outside by Euclidean balls, a key property necessary for constructing a separation oracle for the feasible set later.

Theorem 5 (Convex program solution – optimal regularizer). Assume we are given a smooth barrier function $f : \mathbb{R}^d \to \mathbb{R}$ with $f(x) \in C^2$, $\forall x \in \mathcal{X}$, which is c_1 Lipschitz, c_2 gradient Lipschitz, L Hessian Lipschitz, and $\|\cdot\|_{L_c}$-strongly convex in $\mathcal{X}$. Additionally, if for every two points in the cover $x_i, x_j \in \mathcal{X}$ we have $\|x_i - x_j\| \le \beta$, then the convex program in equation 2.13 with $c_0 = c_1$, $L = \gamma_3 c_2^3$, $L = L$, $C_0 = C^2 = L^3$, and discretization parameter $\beta = \gamma_3 \min\left(\frac{\alpha}{d c_1}, \frac{\alpha}{c_1 d c_2^3}, \frac{1}{c_2}, \sqrt{\frac{c_1 d}{c_2}}, \frac{\alpha^3}{512 R^6 L^2 c_1 d}\right)$ for all sufficiently small constant γ_3 is feasible. Furthermore, the function $g(\cdot)$, corresponding to the optimal solution $(\tau^\ddagger, r^\ddagger, v^\ddagger, \alpha^\ddagger)$ is convex and satisfies the following properties:

1. $|g(\tau^\ddagger)(x) - f(x)| \le \gamma_2 c_0 d$ for constant γ_2.
2. For any feasible instance $(\tau > \tau^\ddagger)$, $g(\tau)(x)$ is $\frac{\alpha}{2}$ strongly convex with respect to $\|\cdot\|_{L_c}$.
3. $\gamma_3 \cdot 288\,(\tau - \tau^\ddagger) \le \tau^\ddagger \le \tau$ for $\gamma_2 \le \sqrt{\frac{\alpha}{N(1 + C_0^2 N d)}}\,\frac{1}{c_2^2}(\tau - \tau^\ddagger)$.

The proof of Theorem 5 can be found in Section C.7.4.

Chapter 3

Projection-Free Online Convex Optimization via E cient Newton Iterations

Abstract

This paper presents new projection-free algorithms for Online Convex Optimization (OCO) over a convex domain $\mathcal{K} \subseteq \mathbb{R}^d$. Classical OCO algorithms (such as Online Gradient Descent) typically need to perform Euclidean projections onto the convex set $\mathcal{K}$ to ensure feasibility of their iterates. Alternative algorithms, such as those based on the Frank-Wolfe method, swap potentially-expensive Euclidean projections onto $\mathcal{K}$ for linear optimization over $\mathcal{K}$. However, such algorithms have a sub-optimal regret in OCO compared to projection-based algorithms. In this paper, we look at a third type of algorithms that output approximate Newton iterates using a self-concordant barrier for the set of interest. The use of a self-concordant barrier automatically ensures feasibility without the need of projections. However, the computation of the Newton iterates requires a matrix inverse, which can still be expensive. As our main contribution, we show how the stability of the Newton iterates can be leveraged to only compute the inverse Hessian a vanishing fractions of the rounds, leading to a new e cient projection-free OCO algorithm with a state-of-the-art regret bound.

3.1 Introduction

We consider the Online Convex Optimization (OCO) problem over a convex set $\mathcal{K} \subseteq \mathbb{R}^d$, in which a learner (algorithm) plays a game against an adaptive adversary for T rounds. At each round $t = 1, \ldots, T$, the learner picks $\mathbf{w}_t \in \mathcal{K}$ given knowledge of the history $H_{t-1} = \{\ell_1, \mathbf{w}_1, \ldots, \ell_{t-1}, \mathbf{w}_{t-1}\}$. Then, the adversary picks a convex loss function $\ell_t : \mathcal{K} \to \mathbb{R}$ with the knowledge of H_{t-1} and the iterate $\mathbf{w}_t$, and the learner su ers loss $\ell_t(\mathbf{w}_t)$ and proceeds to the next round. The goal

of the learner is to minimize the regret after T rounds:

$$R_T(\mathbf{w}) = \sum_{t=1}^{T} \ell_t(\mathbf{w}_t) - \sum_{t=1}^{T} \ell_t(\mathbf{w}),$$

against any comparator $\mathbf{w} \in \mathcal{K}$. The aim of this paper is to design computationally-efficient (projection-free) algorithms for OCO that enjoy the optimal (up to log-factor in T) $O(\sqrt{T})$ regret.

The OCO framework captures many optimization settings relevant to machine learning applications. For example, OCO algorithms can be used in offline convex optimization as more computationally- and memory-efficient alternatives to interior-point and cutting plane methods whenever the dimension d is large [29, 30]. OCO algorithms are also often used in stochastic convex optimization, where the standard $O(\sqrt{T})$ regret (achieved by, e.g. Online Gradient Descent) translates into the optimal $O(1/\sqrt{T})$ rate[1] via the classical online-to-batch conversion technique [31, 32]. It has been shown that OCO algorithms can also achieve state-of-the-art accelerated rates in both the offline and stochastic optimization settings despite being designed for the more general OCO framework [33, 34]. What is more, it has recently been shown that even non-convex (stochastic) optimization can be reduced to online linear optimization (a special case of OCO), where it is then possible to recover the best-known convergence rates for the setting [35].

Given the prevalent use of OCO algorithms in machine learning applications, it is important to have computationally-efficient algorithms that scale well with the dimension d of the ambient space. However, most OCO algorithms fall short of being efficient because of the need of performing (Euclidean) projections onto $\mathcal{K}$ (potentially at each iteration) to ensure that the iterates are feasible. These projections are often inefficient, especially in high-dimensional settings with complex feasible sets. Existing projection-free OCO algorithms address this computational challenge by swapping potentially-expensive Euclidean projections for often much cheaper linear optimization or separation over the feasible set $\mathcal{K}$. However, existing projection-free algorithms have sub-optimal regret guarantees in terms of their dependence in T, or have potentially unbounded condition numbers for the feasible set multiplying their regret guarantee.

Contributions. In this paper, we address these computational and performance challenges by revisiting an existing (but somewhat overlooked) type of projection-free OCO algorithms. Unlike existing algorithms, our proposed method does not require linear optimization or separation over the feasible set $\mathcal{K}$. Instead, the algorithm, Barrier-Regularized Online Newton Step (BARONS),[2] uses a self-concordant barrier for the set $\mathcal{K}$ to always output iterates that are guaranteed to be within $\mathcal{K}$; much like interior point methods for offline optimization. In particular, our algorithm outputs Newton iterates with respect to time-varying, translated versions of . The main novelty of our work is in devising a new efficient way of computing the Newton iterates without having to evaluate the inverse of the Hessian of the barrier at

[1] This is the optimal rate when no further assumptions are made.

[2] We credit the name BARONS to [36] who used barrier-regularized Newton steps for the portfolio selection problem.

every iteration, which can be computationally expensive in high-dimensional settings. Our algorithm only needs to compute a full inverse of the Hessian a vanishing $O(1/\sqrt{T})$ fraction of the rounds. For the rest of the rounds, the computational cost is dominated by that of evaluating the gradient of the barrier , which can be much cheaper than evaluating the inverse of its Hessian in many cases.

For the special case of a polytope with m constraints, we show that there is a choice of a barrier (e.g. the Lee-Sidford barrier) that when used within our algorithm, reduces the per-round computational cost to essentially $O(m)$ linear-system-solves of size $m \times d$. We show that this is often cheaper than performing linear optimization over $\mathcal{K}$, which other projection-free algorithms require. More importantly, our algorithm achieves a dimension-free $O(\sqrt{T})$ regret bound. This improves over the existing regret bounds of projection-free algorithms over polytopes. For example, among projection-free algorithms that achieve a $O(\sqrt{T})$ regret, the algorithms by [34, 37, 38], which require a separation/membership Oracle for $\mathcal{K}$, have a multiplicative R/r factor multiplying their regret bounds, where $r, R > 0$ are such that $B(r) \subseteq \mathcal{K} \subseteq B(R)$. The constant , known as the *asphericity* [39], can in principle be arbitrarily large. Even after applying a potentially expensive pre-processing step, which would typically involve putting the set $\mathcal{K}$ into (near-) isotropic position [40, 41], can still be as large as $\sqrt{d}$ in the worst-case, and so the regret bounds achieved by the algorithms of [34, 37, 38] can be of order $O(d\sqrt{T})$; this is worse than ours by a $\sqrt{d}$ factor. Other projection-free algorithms based on the Frank-Wolfe method, e.g. those in [42 44], also have multiplicative condition numbers that are even less benign that the asphericity. In fact, the condition numbers in the regret bounds for polytopes appearing in, e.g. [42], can in principle be arbitrarily large regardless of any pre-processing.

Finally, another advantage of our algorithm is that it can guarantee a sublinear regret even for non-Lipschitz losses (i.e. where the norm of the sub-gradients may be unbounded). In particular, we show that the general guarantee of BARONI implies a $O(d\sqrt{T})$ regret bound for the portfolio selection problem [45] and a problem of linear prediction with log-loss [46], all while keeping the per-round computational cost under $O(d^2)$, when $T \geq d$. The losses in both of these problems are neither bounded or Lispchitz.

Related works. In the past decade, many projection-free OCO algorithms have been developed to address the computational shortcoming of their projection-based counter parts [29, 30, 34, 37, 47, 48]. Most projection-free algorithms are based on the Frank-Wolfe method and perform linear optimization (typically once per round) over $\mathcal{K}$ instead of Euclidean projection. Under no additional assumptions other than convexity and lipschitzness of the losses, the best-known regret bound for such algorithms scales as $O(T^{3/4})$ [47]. While this bound is still sublinear in T and has no dependence in the dimension d, it is sub-optimal compared to the $O(\sqrt{T})$ regret bound achievable with projection-based algorithms. In the recent years, there have been improvements to this bound under additional assumptions such as when the functions are smooth and/or strongly convex [47, 48], or when the convex set is smooth and/or strongly convex [49 52]. For the case where $\mathcal{K}$ is a polytope, [42] presented a linear-optimization-based algorithm that enjoys a $O(d\sqrt{T})$ regret bound, where is a conditioning number for the set $\mathcal{K}$. Unfortunately, can be large for many sets of interests

as it essentially scales inversely with the minimum distance between the vertices of $\mathcal{K}$. In this work, we achieve a dimension-free $\tilde{O}(\sqrt{T})$ regret bound without the ρ factor.

More recently a new type of projection-free algorithms have emerged which use membership/separation oracle calls instead of linear optimization [34, 37, 38, 52]. From a computational perspective, separation-based and linear optimization-based algorithms are not really comparable, since there are sets over which separation is cheaper than linear optimization, and vice-versa. On the regret side, separation-based algorithms have been show to achieve a $O(\rho\sqrt{T})$ regret bound, where ρ is the asphericity of the set $\mathcal{K}$. Separation-based algorithms are simple, often easy to analyze, and achieve the optimal $\sqrt{T}$-regret bound, unlike linear optimization-based algorithms. However, the multiplicative factor ρ in their regret bounds means that a pre-conditioning step may be required to ensure it is appropriately bounded. This precondition step would involve putting the set into (near-) isotropic position [40]; an operation, that can cost $\tilde{O}(nd^4)$ arithmetic operations [41]; and even after such a pre-processing step, ρ can still be as large as d in the worst-case. Our algorithm has the benefit of not requiring any pre-processing step.

A third type of algorithms avoid projections by outputting Newton iterates that are guaranteed to be feasible thanks to the use of a self-concordant barrier. The first such algorithm in the context of online learning was introduced by [53]. They presented a general recipe for using self-concordant barriers with Newton steps in online linear optimization. However, their approach falls short of being computationally-efficient as their algorithm needs to compute the inverse of the Hessian of the barrier at every iteration. Inspired by the work of [53], [54] used damped Newton steps with quadratic terms added to the barrier to design an efficient algorithm for the classical portfolio selection problem. Closer to our work is that of [55] who used a similar barrier for designing an algorithm for exp-concave optimization that can be viewed as a computationally-efficient version of the Online Newton Step [56]. Similar to our work, [55] also leverage the stability of the Newton iterates to avoid computing the inverse of the Hessian of the barrier at every step. However, their approach and analysis, which are tailored to the exp-concave setting do not necessarily lead to improved regret bounds in the general OCO setting we consider. In particular, their algorithm does not lead to a $\tilde{O}(\sqrt{T})$ regret bound over polytopes.

Finally, for our application to polytopes, we make use of recent tools and techniques developed for solving linear programs efficiently. In particular, we make use of the Lee-Sidford barrier [2, 57, 58], which can be computed efficiently and, when used to compute Newton iterates, leads to the state-of-the-art $\tilde{O}(d)$ iteration upper-bound for solving a linear program. For the OCO setting, we show that using the Lee-Sidford barrier within our algorithm leads to a $\tilde{O}(\sqrt{T})$ regret bound. We also note that ideas similar to the ones we use to avoid computing the inverse of the Hessian of the barrier at every round were used to amortize computations in the context of solving linear programs (see e.g. [59, 61]).

Outline. In section F.1, we present our notation and relevant definitions. In Section 3.3, we present our algorithm and guarantees. In Section 3.4, we apply our results to the case of a polytope. All the proof are differed to the appendix.

3.2 Preliminaries

Throughout the paper, we let K be a closed convex subset of $\mathbb{R}^d$. We denote by $\|\cdot\|$ the Euclidean norm and by $B(R) = \{x \in \mathbb{R}^d : \|x\| \le R\}$ the Euclidean ball of radius $R > 0$. We let $\operatorname{int} K$ denote the interior of K.

Our main algorithm, which can be viewed as an online counter-part to the Newton iterations [62], uses a self-concordant barrier over the set of interest to avoid the need of performing Euclidean projections onto K. Next, we present the definition of a self-concordant barrier.

Self-concordant barriers. For the rest of this section, we let K be a convex compact set with non-empty interior $\operatorname{int} K$. For a twice [resp. thrice] differentiable function, we let $\nabla^2 f(u)$ [resp. $\nabla^3 f(u)$] be the Hessian [resp. third derivative tensor] of f at u.

Definition 6 (Self-concordant function) *A convex function $f : \operatorname{int} K \to \mathbb{R}$ is called self-concordant with constant $M_f \ge 0$, if f is C^3 and satisfies*

- $f(x_k) \to a$ *for* $x_k \to x \in \partial K$; *and*
- *For all $x \in \operatorname{int} K$ and $u \in \mathbb{R}^d$, $\nabla^3 f(x)[u,u,u] \le 2M_f \|u\|^3_{\nabla^2 f(x)}$.*

Definition 7 (Self-concordant barrier) *For $M_f \ge 0$, we say that $f : \operatorname{int} K \to \mathbb{R}$ is a (M_f, ν)-self-concordant barrier for K if f is a self-concordant function over K with constant M_f and*

$$\forall w \in \operatorname{int} K, \quad \nabla f(w) \in \nabla^2 f(w)\, B(\nu).$$

Computational Oracles. We will assume that our algorithm has access to a self-concordant function over the set K through the following gradient and Hessian Oracles.

Definition 8 (Gradient Oracle). *Given a point $w \in \operatorname{int} K$ and a tolerance $\varepsilon > 0$, the gradient Oracle $O^{\mathrm{grad}}_\varepsilon(\cdot)$ returns an ε-approximate vector $\hat{g}_w$ of the gradient $\nabla f(w)$ in the dual local norm of the Hessian:*

$$\|\hat{g}_w - \nabla f(w)\|_{\nabla^2 f(w)} \le \varepsilon.$$

We denote by $C^{\mathrm{grad}}_\varepsilon(\cdot)$ the computational cost of one call to $O^{\mathrm{grad}}_\varepsilon(\cdot)$.

When clear from the context, we will simply write C^{grad} and $O^{\mathrm{grad}}_\varepsilon$ for $C^{\mathrm{grad}}(\cdot)$ and $O^{\mathrm{grad}}_\varepsilon(\cdot)$, respectively.

Definition 9 (Hessian Oracle) *Given a point $w \in \operatorname{int} K$ and a tolerance $\varepsilon > 0$, the Hessian Oracle $O^{\mathrm{hess}}_\varepsilon(\cdot)$ returns a matrix H and its inverse H^{-1} which are $(1 \pm \varepsilon)$ spectral approximations of the Hessian and inverse Hessian of at w:*

$$(1-\varepsilon)\nabla^2 f(w) \preceq H \preceq (1+\varepsilon)\nabla^2 f(w) \quad \text{and} \quad (1-\varepsilon)\nabla^2 f(w)^{-1} \preceq H^{-1} \preceq (1+\varepsilon)\nabla^2 f(w)^{-1}.$$

We denote by $C^{\mathrm{hess}}(\cdot)$ the computational cost of one call to $O^{\mathrm{hess}}_\varepsilon(\cdot)$.

When clear from the context, we will simply write C^{hess} and $O^{\mathrm{hess}}_\varepsilon$ for $C^{\mathrm{hess}}(\cdot)$ and $O^{\mathrm{hess}}_\varepsilon(\cdot)$, respectively.

Additional notation. We use the notation $f \ß g$ to mean $f \ B \ C g$ for some universal constant $C \ A \ 0$. We also write $f \ B \ \tilde{O} g$ to mean $f \ B \ \text{polylog}(T; d\bullet) g$. We let $\© \ ^2 \ \hat{}\©^{2}\bullet \ ^1$ and $\© \ ^{1=2}$ refer to the inverse of the Hessian and the inverse of the square root of the Hessian, respectively.

3.3 Algorithm and Regret Guarantees

In this section, we construct a projection-free algorithm for Online Convex Optimization. The algorithm in question (Alg. 1) outputs approximate Newton iterates with respect to potential functions $\hat{} \ _t\bullet$ that take the following form:

$$_t\hat{}w\bullet \quad \hat{}w\bullet \quad w\bar{} \sum_{s \ 1}^{t \ 1} g_s;$$

where $\hat{}g_s > @_s\grave{}\hat{}w_s\bullet\bullet$ are the sub-gradients of the losses $\grave{}s_s\bullet$ at the iterates $\hat{}w_s\bullet$ of Algorithm 1, and $$ is a self-concordant function over K. Algorithm 1 uses the the approximate gradient and Hessian Oracles of $$ (see F.1) to output iterates $\hat{}w_t\bullet$ approximate Newton iterates in the following sense:

$$¦ t > T ; \quad w_{t \ 1} \quad w_t \quad \© \ ^2 \ _{t \ 1}\hat{}w_t\bullet\© \ _{t \ 1}\hat{}w_t\bullet: \tag{3.1}$$

As is by now somewhat standard in the analyses of online Newton iterates of the form in equation 3.1, we will bound the regret of Algorithm 1 by showing that:

- The iterates $\hat{}w_t\bullet$ are close (in the norm induced by the Hessian $@^2 \ \hat{}w_t\bullet$) to the FTRL iterates, which are given by

$$w_t^{\dagger} > \arg\min_{w > K} \ _t\hat{}w\bullet: \tag{3.2}$$

- The regret of FTRL is bounded by $O\hat{}^{o} \ \bar{T}\bullet$.

Our main contribution is an algorithm that outputs iterates $\hat{}w_t\bullet$ that satisfy the first bullet point (i.e. iterates that satisfy equation 3.1), while only calling a Hessian Oracle (which is potentially computationally expensive) at $O\hat{}1\tilde{} \ \bar{T}\bullet$ fraction of the rounds after T rounds. As we show in Section 3.4, for the case where K is a polytope with $m > N$ constraints, the algorithm achieves a $\tilde{O}\hat{} \ \bar{T}\bullet$ regret bound, where the per-iteration computational cost essentially reduces to a linear-system-solve involving a $d \ m$ matrix. Among existing OCO algorithms that achieve a $\tilde{O}\hat{} \ \bar{T}\bullet$ regret bound, none can achieve this computational complexity for general polytopes with m constraints (see Section 3.4 for more details).

3.3.1 Efficient Computation of the Newton Iterates with BARONS

The key feature of BARONS (Algorithm 1) is that is uses an amortized computation of the Hessians. Namely, BARONS computes the inverse of the Hessian of the barrier only for a small fractions of the iterates $\hat{}w_t\bullet$. Henceforth, we refer to the iterates where the algorithm

Algorithm 1 BARONS: Barrier-Regularized Online Newton Step

Require:

- Parameters $\eta, \varepsilon > 0$, and $m_{\text{Newton}} \geq 1$.
- Gradient/Hessian Oracles $O^{\text{grad}}, O^{\text{hess}}$ for self-concordant function ϕ with constant $M > 0$.
- $w^\dagger = \arg\min_{w \in K} \phi(w)$.

1: Set $u_1 = w_1 = w^\dagger$, $H_1 = O^{\text{hess}}(w^\dagger)$, and $s_0 = 0$.
2: **for** $t = 1, \ldots, T$ **do**
3: Play w_t and observe $g_t = \nabla \ell_t(w_t)$.
4: Set $s_t = s_{t-1} + g_t$.
5: Set $w_{t-1}^1 = w_t$.
// Perform intermediate Newton steps to decrease the Newton decrement
6: **for** $m = 1, \ldots, m_{\text{Newton}} - 1$ **do**
7: Set $\hat{\theta}_{t-1}^m = O^{\text{grad}}(w_{t-1}^m)$.
8: Set $\hat{\theta}_{t-1}^m = \hat{\theta}_{t-1}^m + s_t$. // $\hat{\theta}_{t-1}^m = \theta_{t-1} + \nabla \phi(w_{t-1}^m)$
9: Set $w_{t-1}^{m+1} = w_{t-1}^m - H_{t-1}^{-1} \hat{\theta}_{t-1}^m$.
10: **end for**
11: Set $w_{t-1} = w_{t-1}^{m+1}$.
// Check if the landmark needs updating
12: **if** $\|w_{t-1} - u_t\|_{H_t} \leq \frac{1}{4} \cdot \frac{1}{M}$ **then**
13: Set $u_{t-1} = u_t$. // Update landmark
14: $H_{t-1} = H_t$ and $H_{t-1}^{-1} = H_t^{-1}$.
15: **else**
16: Set $u_{t-1} = w_{t-1}$. // No landmark update
17: Set $H_{t-1} = O^{\text{hess}}(u_{t-1})$ and compute H_{t-1}^{-1}.
18: **end if**
19: **end for**

computes the full inverse of the Hessian at landmark iterates; these are the iterates $\phi(u_t)$ in Lines 13 and 16 of Algorithm 1. The idea behind this is that for a sufficiently curved[3] barrier ϕ, the Newton iterates with respect to ϕ are stable enough that it suffices to compute the inverse of the Hessian of ϕ at the closest landmark iterate. For example, this is what was done in [55] to design an efficient algorithm for exp-concave optimization.

Unlike the setting of [55], where it is possible to add quadratic terms to the barrier for additional stability, in our setting we cannot do that without sacrificing performance in terms of regret. Without the quadratic terms, the Newton iterates are not stable enough for our desired guarantee. Instead of adding regularization terms, BARONS takes $O(1)$ Newton steps per round to get closer to the Newton iterate with the true Hessian matrix. This simple approach is key to the success of our approach.

In the next subsection, we give a generic guarantee for BARONS.

[3] Informally, the curvature of a convex function is high when the rate of change of its gradients is high.

3.3.2 Generic Regret Guarantee of BARONS

In this subsection, we present a general regret and computational guarantee for BARONS under minimal assumptions on the sequence of losses and without turning the step size. In the next subsection, we will instantiate the regret guarantee when additional assumptions on the sequence of losses are available. We now state the main guarantee for BARONS (the proof in Appendix D.3.1).

Theorem 6 (Master theorem). Let ϕ be a self-concordant function over $\mathcal{K}$ with constant $M \geq 1$, and let $b, \gamma, \varepsilon, G \geq 0$ and $m_{\text{Newton}} \in \mathbb{N}$ be such that $B \leq \frac{1}{1000bM}$, $\varepsilon \leq \frac{1}{20000M}$, $\gamma \leq 0.001$, and $m_{\text{Newton}} \geq \log \frac{1}{\varepsilon M}$. Further, let w_t be the iterates of Algorithm 1 with input $(\phi, \varepsilon, \gamma, m_{\text{Newton}})$ and suppose that the corresponding sub-gradients g_t satisfy $\|g_t\|_{\nabla^{-2}\phi(w_t)} \leq B$, for all $t \geq 1$. Then, the regret of Algorithm 1 is bounded as:

$$\sum_{t=1}^{T} \ell_t(w_t) - \ell_t(w) \leq \frac{1}{\gamma}\phi(w) + \sum_{t=1}^{T} \gamma \|g_t\|^2_{\nabla^{-2}\phi(w_t)} + \varepsilon \sum_{t=1}^{T} \|g_t\|_{\nabla^{-2}\phi(w_t)}, \quad \forall w \in \operatorname{int} \mathcal{K}. \tag{3.3}$$

Furthermore, the computational cost of the algorithm is bounded by

$$O\left(C^{\text{grad}}(d^2) \cdot T \log \frac{1}{\varepsilon M} + C^{\text{hess}}(d) \cdot (M \cdot T\varepsilon) \cdot M \sum_{t=1}^{T} \|g_t\|_{\nabla^{-2}\phi(w_t)}\right).$$

Theorem 6 essentially shows that it is possible to achieve the same regret as FTRL, while only computing the inverse of the Hessian of ϕ at most $O(M \cdot T)$ number of times.

3.3.3 Regret Guarantee Under Local and Euclidean Norm Bounds on the Sub-Gradients

We now instantiate the guarantee in Theorem 6 with a (M, ν)-self-concordant barrier ϕ for the set $\mathcal{K}$, with respect to which the local norms of the sub-gradients are bounded; that is, when $\|g_t\|_{\nabla^{-2}\phi(w_t)} \leq b$. We note that the regret bound in equation D.17 has an additive $\phi(w)$ which may be unbounded near the boundary of $\mathcal{K}$. However, it is still possible to compete against comparators in $\operatorname{int} \mathcal{K}$ by making additional assumptions on the range of the losses [36, 63]. We discuss some of these assumptions in the sequel. For the next theorem, we will state the regret bound of BARONS relative to comparators in the restricted set:

$$\mathcal{K}_c \triangleq (1-c)\mathcal{K} + cw^\dagger, \tag{3.4}$$

where $+$ denotes the Minkowski sum, $w^\dagger \in \arg\min_{w \in \mathcal{K}} \phi(w)$, and $c \in (0, 1)$ is a parameter.

With this, we now state a regret bound for BARONS when the sub-gradients of the losses have bounded local norms. The proof of the next theorem is in Appendix D.3.2.

Theorem 7 (Local norm bound). Let ϕ be an (M, ν)-self-concordant barrier for $\mathcal{K}$ and let $c \in (0, 1)$, $b, G \geq 0$. Further, suppose that for all $t > T_0$, $\|g_t\|_{\nabla^{-2}\phi(w_t)} \leq b$, where w_t are the iterates of BARONS with input parameters $(\phi, \varepsilon, \gamma, m_{\text{Newton}})$ such that

$$\gamma \leq \frac{\log c}{b^2 T}, \quad \varepsilon \leq \frac{\gamma}{T}, \quad 0.001 \geq \gamma, \quad \text{and} \quad m_{\text{Newton}} \geq \log \frac{1}{\varepsilon M}. \tag{3.5}$$

For $T \geq 1$ large enough such that $\beta \leq \frac{1}{1000dM}$, $\varepsilon \leq \frac{1}{20000M}$, the regret of BARONS is bounded as

$$\mathrm{Regret}_T^{\mathrm{BARONS}}(w^\star) \leq b\sqrt{T\log c}, \quad \forall w \in K_c, \tag{3.6}$$

where $K_c \subset K$ is as in equation 3.4. Further, the computational complexity of BARONS in this case is bounded by

$$O\left(\mathcal{G}_{\mathrm{oracle}}^{\mathrm{grad}}\, d^2 T \log\frac{T}{M} + \mathcal{C}^{\mathrm{hess}} M^2 T \log c\right).$$

Remark 1. The regret bound in Theorem 7 is stated with respect to comparators in the restricted set K_c defined in equation 3.4. It is possible to extend this guarantee to all comparators in $\mathrm{int}\, K$ under an additional assumption on the range of the losses. For example, if for $w^\dagger \in \arg\min_{w \in K} f(w)$, we have

$$\sup_{w \in \mathrm{int}\, K, t \geq T} \ell_t(w) - \ell_t\left(\frac{1}{T}w + \left(1 - \frac{1}{T}\right)w^\dagger\right) \leq O\left(\frac{1}{\sqrt{T}}\right), \tag{3.7}$$

then the regret guarantee in equation 3.6 can be extended to all comparators in $\mathrm{int}\, K$ up to an additive $O(\sqrt{T})$ term (see Lemma 72 in the appendix). In this case, the $\log c$ term in the computational complexity need be replaced by $\log T$. We note that the condition in equation 3.7 does not require a uniform bound on the losses. Instead, it only restrict the rate of growth of the losses $\ell_t(w)$ as w approaches the boundary of K. As we show in the sequel (§3.4.2), equation 3.7 is satisfied for some popular losses which are not Lipschitz.

We now instantiate the guarantee in Theorem 6 when the sub-gradients are bounded in Euclidean norm (instead of local norm); that is, we assume that for all $t \geq T$, $\|g_t\| \leq G$ for some $G > 0$. We note that this assumption implies equation 3.7, and we will be able to bound the regret against all comparators in $\mathrm{int}\, K$ as alluded to in Remark 1. The proof of the next theorem is in Appendix D.3.3).

Theorem 8 (Euclidean norm bound) Let f be an (M, ν) self-concordant barrier for K and let $\alpha = \alpha(M, \nu) = \frac{1}{2R^2}\|\cdot\|^2$. Further, let $G, R > 0$ and suppose that $K \subseteq B(R)$ and for all $t \geq T$, $\|g_t\| \leq G$, where $g_t \in \partial\ell_t(w_t)$ and (w_t) are the iterates of BARONS with input parameters $(\alpha, \varepsilon, \beta, m_{\mathrm{Newton}})$ such that

$$\beta \leq \frac{1}{RG}\sqrt{\frac{\log T}{T}}, \quad \varepsilon \leq \frac{1}{T^2}, \quad \beta \leq 0.001, \quad \text{and} \quad m_{\mathrm{Newton}} \geq \log\frac{1}{\varepsilon M}. \tag{3.8}$$

For $T \geq 1$ large enough such that $\beta \leq \frac{1}{1000GM}$, $\varepsilon \leq \frac{1}{20000M}$, the regret of BARONS is bounded as

$$\mathrm{Regret}_T^{\mathrm{BARONS}}(w^\star) \leq RG\sqrt{T\log T}, \quad \forall w \in \mathrm{int}\, K. \tag{3.9}$$

Further, the computational complexity of BARONS in this case is bounded by

$$O\left(\mathcal{G}_{\mathrm{oracle}}^{\mathrm{grad}}(\cdot)\, d^2 T \log\frac{T}{M} + \mathcal{C}^{\mathrm{hess}}(\cdot)\, M^2 T \log T\right).$$

3.4 Application to Polytopes Using the Lee-Sidford Barrier

In this section, we assume that the set K is a polytope in $\mathbb{R}^d$ specified by m linear constraints:

$$K := \{w \in \mathbb{R}^d : \langle a_i, w \rangle \geq b_i, \ i = 1, \ldots, m\}, \tag{3.10}$$

and we construct efficient gradient and Hessian Oracles for a self-concordant barrier for K. This will then allow us to instantiate the guarantees of BARONS in Section 3.3 and provide explicit and state-of-the-art bounds on the regret of BARONS.

We will assume without loss of generality that $\|a_i\|_2 = 1$, for all $i = 1, \ldots, m$, and let $A := [a_1, \ldots, a_m]^\top \in \mathbb{R}^{m \times d}$ denote the constraint matrix of the set K. For the rest of this section, it will be convenient to define the slack variables $s_{w,i} := \langle a_i, w \rangle - b_i$, for $i = 1, \ldots, m$. Here, $s_{w,i}$ essentially represents the distance of w to the ith facet of the polytope K. Further, we let $S_w := \mathrm{diag}(s_w)$ be the diagonal matrix whose ith diagonal entry is $s_{w,i}$.

The LS barrier. To perform Online Convex Optimization over K, we pick the regularizer of BARONS to be the Lee-Sidford (LS) barrier Φ_{LS} [2] with parameter $p = 1$, which is defined as

$$\Phi_{LS}(v) := \min_{v \in \mathbb{R}^m_{>0}} \log\det(A^\top S_w^{-1} V S_w^{-1} A) - \frac{1}{1 - p^{-1}} \mathrm{Tr}(V^{1 - 1/p}),$$

where $V := \mathrm{diag}(v)$. One way to think of the LS barrier is as a weighted log-barrier. As we will discuss in the sequel, this choice will confer computational and performance (in terms of regret) advantages over the standard log-barrier.

Self-concordance of the LS barrier. According to [64, Theorem 30], the LS barrier with the choice $p := \log(m)$ is a self-concordant function with parameter M_{LS} satisfying

$$M_{LS} := O(\log(m)^{2.5}) = \tilde{O}(1);$$

The other favorable property of this barrier is that its Newton decrement at any point $w \in K$ is of order $\tilde{O}(d)$; that is,

$$\|\nabla \Phi_{LS}(w)\|_{\nabla^2 \Phi_{LS}(w)^{-1}} \leq \tilde{O}(d). \tag{3.11}$$

Therefore, Φ_{LS} is a $(\tilde{O}(1), \tilde{O}(d))$-self-concordant barrier. For the log-barrier, the right-hand side of equation 3.11 would be m.

Cost of gradient and Hessian Oracles. We consider the computational complexities of gradient and Hessian Oracles for Φ_{LS}. By [2], we have that for $\varepsilon > 0$,

$$C_{\mathrm{grad}}(\Phi_{LS}) = \tilde{O}(C_{\mathrm{sys}}) \log(1/\varepsilon); \quad \text{and} \quad C_{\mathrm{hess}}(\Phi_{LS}) = \tilde{O}(C_{\mathrm{sys}}) \cdot d \log(1/\varepsilon);$$

where C^{sys} is the computational cost of solving a linear system of the form $A^T \mathrm{diag}(v) A x = y$, for vectors $v \in \mathbb{R}^d_{++}$ and $y \in \mathbb{R}^d$; we recall that $A = [a_1, \ldots, a_m]^T$ is the constraint matrix for K. In the worst-case, such a linear system can be solved with cost bounded as

$$C^{sys} = O(md^{\omega - 1}), \tag{3.12}$$

where ω is the exponent of matrix multiplication, and m is the number of constraints of K. However, as we show in the sequel, C^{sys} can be much smaller in many practical applications.

With this, we immediately obtain the following corollary for the regret and run-time of BARONS under local norm and Euclidean norm bounds on the sub-gradients.

Corollary 1 (OCO over a polytope with LS barrier). Let $c \in (0, 1]$, $G, R, b, A > 0$, and suppose K is given by equation 3.10 and that ϕ^{LS} is the corresponding LS barrier. Further, let (w_t) be the iterates of BARONS, and let K_c be the restricted version of K defined in equation 3.4. Then, the following holds:

- Local norm bound: If $\|g_t\|_{w_t}^* \le b$, for all $t \ge 1$, and the parameters $(\eta, \varepsilon, m_{Newton})$ of BARONS are set as in Theorem 7 with $\phi = \phi^{LS}$ and $(M, \alpha) = (\Theta(1), \Theta(d))$, then for T large enough (as specified in Theorem 7), the regret of BARONS is bounded by

$$R_T^{BARONS}(w) \le b\sqrt{dT \log c}, \quad \forall w \in K_c. \tag{3.13}$$

- Euclidean norm bound: If $K \subseteq B(R)$ and $\|g_t\| \le G$, for all $t \ge 1$, and the parameters $(\eta, \varepsilon, m_{Newton})$ of BARONS are set as in Theorem 8 with $\nu = \phi^{LS}(\nu) = \frac{1}{2R^2}\|x - y\|^2$ and $(M, \alpha) = (\Theta(1), \Theta(d))$, then for T large enough (as in Theorem 8), BARONS has regret bounded as

$$R_T^{BARONS}(w) \le RG\sqrt{T \log T}, \quad \forall w \in \mathrm{int}\, K. \tag{3.14}$$

In either case, the computational complexity is bounded by

$$O(C^{sys} + d^2) + T \cdot C^{sys} = d \cdot \tilde{T}, \tag{3.15}$$

where C^{sys} is the computational cost of solving a linear system of the form $A^T \mathrm{diag}(v) A x = y$, for vectors $v \in \mathbb{R}^d_{++}$ and $y \in \mathbb{R}^d$ (recall that A is the constraint matrix for the polytope K).

Using the log-barrier. We note that since K is a polytope, we could have used the standard log-barrier

$$\phi^{\log}(w) = \sum_{i=1}^{m} \log(b_i - a_i^T w). \tag{3.16}$$

This barrier is $(1, m)$-self-concordant, and so instantiating Theorem 6 with it would imply a $O(b\sqrt{mdT})$ regret bound in the case of local sub-gradient norms bounded by $A > 0$. Using the LS barrier replaces the m term in this bound by d regardless of the number of constraints see equation 3.13. However, this comes at a C^{sys} computational cost, which can

be as high as $md^{1-\alpha}$ in the worst-case (see equation 3.12). In the case of the log-barrier, this cost would be replaced by md (essentially because $C^{\text{grad}}(\nabla^2 R_{\log}) = O(md)$). Thus, when m is of the order of d, using the log-barrier may be more computational-efficient compared to using the LS barrier. In the next corollary, we bound the regret of BARONS when $R = R_{\log}$; this result is an immediate consequence of Theorem 8.

Corollary 2 (OCO over a polytope with the log barrier). Let $G, b > 0$, and suppose K is given by equation 3.10 and that $R_{\log}$ is the corresponding log-barrier. Further, let w_t be the iterates of BARONS. If $K \subseteq B(R)$ and $\|g_t\| \le G$, for all $t \ge 1$, and the parameters $\eta, \varepsilon, m_{\text{Newton}}$ of BARONS are set as in Theorem 8 with $\nu = \nu_{R_{\log}} = \frac{1}{2R^2}$, $\gamma = \frac{1}{2}\min\{\frac{1}{4GD}, \nu\}$ and $M = 4(GD + \frac{1}{\gamma})$, then for T large enough (as in Theorem 8), BARONS has regret bounded as

$$\mathrm{Reg}_T^{\text{BARONS}}(w) \le 8GD\sqrt{T \log T}, \quad \forall w \in \text{int } K. \tag{3.17}$$

The computational complexity is bounded by

$$O\left(md\left(d^2 + T\left(md^{1-\alpha} + \sqrt{mT}\right)\right)\right). \tag{3.18}$$

3.4.1 Implications for Lipschitz Losses

We now discuss implications of Corollary 1, and compare the bound of BARONS to those of existing algorithms for Lipschitz losses.

Dimension-free regret bound. We note when the Euclidean norms of the sub-gradients are bounded, BARONS achieves a dimension-free $O(\sqrt{T})$ regret bound. In contrast, the best dimension-free regret bound[4] achieved by existing projection-free algorithms is of order $O(T^{3/4})$ (see e.g. [29, 37]). We also note that existing separation/membership-based algorithms that achieve a $\sqrt{T}$ regret; for examples, those presented in [34, 37, 38], are not dimension-free. Their regret bounds are of order $O(r\sqrt{T})$, where $R = \alpha r$ with $r, R, \alpha > 0$ such that $B(r) \subseteq K \subseteq B(R)$. The asphercity parameter can depend on the dimension d [34], and even after a pre-conditioning step (which would involve putting the set K into near-isotropic position and can cost up to $O(d^4)$ [65]), α can be as large as d in the worst-case. Of course, to make a fair comparison with existing projection-free algorithms, we also need to take computational complexity into account. This is what we do next.

Computational cost. The computational cost in equation 3.15 should be compared with that of existing projection-free algorithms. For linear optimization-based projection-free algorithms, the computational cost after T rounds is typically of order $C^{\text{lin}} \cdot T$, where C^{lin} is the cost of performing linear optimization over K which, for a polytope K, reduces to solving a linear program. Using state-of-the-art interior point methods for solving such a linear program would cost $C^{\text{lin}} = O(\sqrt{d}\, C^{\text{sys}})$; see e.g. [2]. Thus, linear optimization-based projection-free algorithms[5] can have a cost that is a factor $\sqrt{d}$ worse than that of BARONS

[4] The dependence in T can be improved under additional structure such as smoothness or strong-convexity of the losses.

[5] This only concerns algorithms that use an interior point method to implement linear optimization over K.

in the setting of Corollary 1. On the other hand, separation/membership-based algorithms, the computational cost scales with $O(C^{sep} T)$ after T rounds, where C^{sep} is the cost of performing separation for the set K. For a general polytope in R^d with m constraints, we have $C^{sep} = O(md)$, which may be smaller than C^{sys} (the latter can be as large as $md + m^3$ in the worse case; see equation 3.12). Here, it may be more appropriate to compare against the computational guarantee of BARONS given in Corollary 2; by equation 3.18, we have that for $T \geq d^2 + m$, the computational cost of BARONS in the setting of the corollary is dominated by $(md + d^2) T$, which is comparable to that of existing separation-based algorithms.

3.4.2 Implications for Non-Lipschitz Losses

Another advantage BARONS has over projection-free, and even projection-based, algorithms is that it has a regret bound that scales with a bound on the local norms of the gradients see equation 3.13. We now showcase two online learning settings where this leads to non-trivial performance and computational improvements over existing OCO algorithms.

Online Portfolio Selection [45]. The portfolio selection problem is a classical online learning problem where the gradients of the losses can be unbounded. In this paragraph, we demonstrate how the guarantee of BARONS in Corollary 1 leads to a non-trivial guarantee for this setting both in terms of regret and computational complexity. In the online portfolio setting, at each round t, a learner (algorithm) chooses a distribution $w_t \in \Delta_d$ over a fixed set of d portfolios. Then, the environment reveals a return vector $r_t \in R^d_{\geq 0}$, and the learner suffers a loss

$$\ell_t(w_t) = -\log \langle w_t, r_t \rangle .$$

The goal of the learner is to minimize the regret $\mathrm{Reg}_T(w) = \sum_{t=1}^{T} (\ell_t(w_t) - \ell_t(w))$ after $T+1$ rounds. For this problem, it is known that a logarithmic regret is achievable, but the specialized algorithms that achieve this have a computational complexity that scales with $\min(d^3 T, d^2 T^2)$ [36, 45, 63, 66, 67]. On the other hand, applying the generic Online Gradient Descent or the Online Newton Step to this problem leads to regret bounds that scale with the maximum norm of the gradient (which can be unbounded). Instantiating the guarantees of BARONS in Corollary 1 with ψ set to the standard log-barrier for the simplex[6], in particular the bound in equation 3.13, to the online portfolio selection problem leads to an $\widetilde{O}(d\sqrt{T})$ regret bound, which does not depend on the norm of the observed gradients. Furthermore, we have $C^{sys} = O(d)$, and so by equation 3.15 the computational complexity is essentially $O(d^2 T)$ after T rounds. Technically, the bound in equation 3.13 is only against comparators in the restricted set K_c. However, by setting $c = 1/T$, it possible to extend this guarantee to all comparators in $\operatorname{int} K$ as explained in Remark 1, since the losses in this case satisfy equation 3.6 [36, Lemma 10].

Linear prediction with the log-loss. Another classical online learning problem with unbounded gradients is that of linear prediction with the log-loss [46]. For this problem, at each round t, the learner receives a feature vector $x_t \in X \subseteq R^d$, outputs $w_t \in W \subseteq R^d$, then

[6] Technically, we need to use a barrier for the set $\{w \in R^d_{\geq 0} : \sum_{i=d+1} w_i \leq 1\}$; see e.g. [63].

observes label $y_t \in Y = \{-1, 1\}$ and suffers loss

$$\ell_t(w_t) = \mathbb{I}\{y_t = 1\} \log(1 + \exp(-w_t \cdot x_t)) + \mathbb{I}\{y_t = 0\} \log(1 + \exp(w_t \cdot x_t)).$$

In the settings, where $(X, W) = (\mathbb{R}^d, B_2(1))$ and $(X, W) = (B_2(1), B_2(1))$, we have that $\|\nabla \ell_t(w)\|_2 \leq \|x_t\|_2 \leq 1$ for all $w \in \text{int } W$, where R is set to the corresponding log-barrier for W. Thus, instantiating Corollary 1 (in particular equation 3.13) in this setting implies that BARONS achieves a regret bound of the form:

$$\mathcal{O}(\sqrt{dT}), \tag{3.19}$$

and has computational complexity bounded by $\mathcal{O}(d^2 T)$, as long as $T \geq d$. Again, we emphasize that the bound in equation 3.19 does not depend on the norm of the gradients, which may be unbounded.

Finally, we note that there exist a few specialized algorithms that provide sublinear regret bounds for non-lipschitz losses. This includes, for example, the Soft-Bayes algorithm [68]. However, this algorithm is specialized to the log-loss with a particular dependence on the predictions, and it is not clear, for example, what regret bound it would have in the linear prediction setting and other similar settings with non-Lipschitz losses.

Chapter 4

Adversarial Online Learning with Temporal Feedback Graphs

abstract

We study a variant of prediction with expert advice where the learner's action at round t is only allowed to depend on losses on a specific subset of the rounds (where the structure of which rounds' losses are visible at time t is provided by a directed feedback graph known to the learner). We present a novel learning algorithm for this setting based on a strategy of partitioning the losses across sub-cliques of this graph. We complement this with a lower bound that is tight in many practical settings, and which we conjecture to be within a constant factor of optimal. For the important class of transitive feedback graphs, we prove that this algorithm is efficiently implementable and obtains the optimal regret bound (up to a universal constant).

4.1 Introduction

Prediction with expert advice is one of the most fundamental problems in online learning. In its simplest form, a learner must choose from one of K actions (possibly choosing a randomized mixture of actions) every round for T rounds. An adversary then reveals a loss vector containing the loss for each action to the learner, the learner incurs their appropriate loss, and play proceeds to the next round. The goal of the learner in such settings is usually to minimize their regret: the gap between their total utility at the end of the game and the maximum utility they could have received if they played the best fixed action in hindsight. Notably, it is possible to construct algorithms for the learner which achieve regret sublinear in T against any adversarially chosen sequence of losses.

Traditionally, a learner may use the entire history of losses up until round t to decide their action at round t. In this paper, we investigate the question of what happens if we restrict the learner's action at time t to depend on the losses in some subset of these rounds. Formally, we require the learner's (randomized) action at time t to be a function of the losses in some

subset of rounds S_t, for some fixed collection of subsets $S \subseteq \{S_t\}_{t=1}^{T}$ known to the algorithm a priori. We call this problem the problem of *online learning with temporal feedback graphs*, in analogy with the use of feedback graphs to understand the value of partial information in online learning settings (e.g. [69]).

In addition to being a mathematically natural extension of the classic problem of prediction with expert advice, this framework is general enough to model many problems of practical interest, including:

- **Batched learning**: In batched learning, the time horizon T is divided into B batches of size $T_1, T_2, \ldots, T_B$. Losses within a batch are only reported at the beginning of the next batch, so e.g. the action at a round t in the bth batch must only depend on losses from the first $b-1$ batches.
- **Learning with delayed feedback**: In learning with delayed feedback, the loss at round t is only reported to the learner at the end of round $t+d$, after d rounds of delay (it is also possible to consider a round dependent delay d_t, in which case this subsumes batched learning).
- **Learning with bounded recall**: In learning with bounded recall, the learner is only allowed to use losses from the past M rounds to decide their action. This captures the notion of bounded recall strategies for playing in repeated games introduced by [70].
- **Learning from the future**: Finally, nothing requires us to impose the constraint that S_t only contains rounds before t: if the adversary fixes their sequence of loss vectors at the beginning of the game, then we can let the learner play a function of the losses in S_t for any subset S_t of $\{1, 2, \ldots, T\}$. More practically, this can be used to model adversarial variants of prediction tasks used in the training of large language models: for example, in the task of masked language modelling (used in the training of BERT), the goal is to predict a masked token from a surrounding window of tokens [71].

4.1.1 Our results

We investigate the problem of designing low-regret algorithms and proving regret lower bounds for the problem of online learning with temporal feedback graphs.

Algorithms (Section 4.3). On the topic of algorithms, we first remark that the seemingly natural algorithm of simply approximately best-responding to the set of losses you can see can actually be very far from optimal. This follows immediately from a result of [72], who show that such algorithms can incur linear regret in T in bounded recall settings (even when the recall window M is large enough to obtain $O(\sqrt{T})$ regret by restarting every M rounds).

Instead, we propose a more sophisticated algorithm (Algorithm 2) based on constructing a fractional decomposition of the directed feedback graph S into ordered cliques, which we call *orders* for short. The key observation here is that the temporal feedback graph corresponding to the classic online learning setting (where you can observe all rounds in the

past) consists of a single order of size T. Therefore, by partitioning the loss vector among the orders that are subgraphs of S, we can reuse the regret guarantees we have from the standard prediction with experts problem and get strong regret bounds for our algorithm.

We analyze the regret of this algorithm and show it is at most $O(UB(S)\sqrt{\log K})$, where $UB(S)$ is the optimal value of a specific convex program that we call the *upper bound program* (Theorem 9). Unfortunately, the size of this convex program (and in particular, the number of variables) is proportional to the number of maximal orders in S, which can be small for some graphs but in general is exponentially large in n. Moreover, actually running Algorithm 2 requires a feasible solution to the upper bound program as input, and takes time proportional to the sparsity of .

Luckily, by examining the dual convex program, we show that there exists an optimal solution ‡ to the upper bound program supported on at most T distinct orders (Lemma 9). This implies that, for a fixed S, there always exists an efficiently implementable learning algorithm achieving the optimum regret bound stated above (Corollary 3), even if it may be hard to actually find this solution and construct this algorithm.

Lower bounds (Section 4.4). We present two different methods for obtaining worst-case regret lower bounds for a fixed feedback graph S (in the binary action case $K = 2$). Similarly as with $UB(S)$, in both methods the lower bound is obtained by solving a convex program defined by the structure of S.

We obtain the first lower bound by extending the classic stochastic lower bound for prediction with expert advice, with the key difference that instead of sampling losses in an iid fashion, we let the loss in round t have bias of magnitude $_t$ for some set of $_t \geq 0$. This results in a lower bound $LB(S)$ on the regret of any algorithm, where $LB(S)$ is given by the value of a polynomial-sized (and hence, efficiently solvable) convex program in the variables $_t$ (Theorem 11). Interestingly, the lower bound program is very similar to the dual of the upper bound program, which lets us immediately conclude that the upper bound $UB(S)$ is near optimal for a wide variety of feedback graphs. In particular, we show that if the in-neighborhood S_t of each round can be covered by at most R orders of S, then $UB(S)/LB(S)$ is at most $O(\overline{R})$ (Theorem 12).

Unfortunately, there are many cases where the gap between $LB(S)$ and $UB(S)$ can be quite large (even polynomial in T). Inspired by this, we introduce a second lower bound which further generalizes the existing stochastic lower bound by allowing correlations between the losses in different rounds. In particular, we construct an adversary who, instead of sampling losses independently for each round, samples a random variable for each independent set of S, and builds the loss for round t out of the random variables of all the independent sets containing t. By doing so, we introduce another convex program we call the *independent set program*. Although this program is large and hard to solve in general (with number of variables equal to the number of independent sets in S), we prove that its value $ILB(S)$ is a lower bound on the regret of any algorithm (Theorem 13) and conjecture that $ILB(S)$ is within a constant factor of $UB(S)$ (thus implying both this bound and our algorithm are within a constant factor of optimal).

Transitive feedback graphs (Appendix E.1). Finally, we consider the interesting subclass of transitive feedback graphs. These are feedback graphs S where if $s > S_t$ and $r > S_s$, then $r > S_t$; that is, if the learner can see loss s at round t, they can also see all the losses r that were visible at round s. Transitive feedback graphs that are a natural class of graphs that include many of our aforementioned applications, such as batched learning and learning with delayed feedback (both the round-dependent delay and round-independent delay case).

For transitive feedback graphs S, we show that we can indeed efficiently construct a sparse solution of the upper bound program, and hence efficiently construct and implement Algorithm 2 given S (Theorem 44 in Appendix E.1). Doing so requires two technical insights: i. first, we show that we can construct an efficient separation oracle for the dual of the upper bound program via dynamic programming, ii. second, we show how we can use a solution $\dagger$ to the dual problem to reduce the problem of constructing a sparse solution of the upper bound problem to a flow-decomposition problem, which we can solve with standard techniques.

Finally, we show that for any transitive graph S our upper bound $UB(S)$ is within a constant factor of optimal (Theorem 45 in Appendix E.1). To do so, we show how the adversary can take any feasible solution to the upper bound dual program and construct a correlated set of losses by observing a one-dimensional Brownian motion at various points in time. Specifically, the adversary associates each round t with an interval $[p_t, q_t]$ of time, and sets the loss at round t based on the sign of $L(q_t) - L(p_t)$ for a biased Brownian motion $L(\cdot)$. By doing so, we effectively construct a feasible solution to our independent set program (which we can show has value within a constant factor of that of the value of our initial feasible solution).

4.1.2 Related work

Since the seminal work of [69] introducing the problem of learning under partial information with feedback graphs, there has been an extensive body of literature extending and refining these results, including [73, 79]. Although we take the name "feedback graph" from this line of work, the similarities between this line of work and the problem we study seem relatively superficial, limited mostly to the fact that both problems are parameterized by a directed graph and have regret bounds that depend on graph-theoretic properties of said graph. It would be interesting to show a stronger connection between these two questions.

Batched feedback has been studied fairly extensively in the online learning community, largely in the contexts of bandits and stochastic rewards [80, 82]. The problem of learning with delayed feedback was first studied in the full-information setting by [83], and has since been studied in a variety of other learning settings [84, 87]. In particular, [87] study the delayed feedback problem with adversarial delays — our framework allows us to immediately recover optimal bounds for this setting given knowledge of the delays. Studying the behavior of agents with bounded recall is an active area in economics (see [88] for a survey) which has recently been studied in the online learning setting by [72]. Finally, many sequence models in machine learning are constrained (often for efficiency / training reasons) so that the tth element of their output can only be based on only of some pre-determined subset of the inputs. Often this subset is given by a context window (as in the bounded-recall setting), but it also can be structured in other interesting ways, as explored by [89].

4.2 Model and Preliminaries

4.2.1 Online learning preliminaries

We begin with an overview of the classic problem of prediction with expert advice. This can be viewed as a repeated game that takes place over T rounds, where in each round the learner selects an action $x_t \in X$ and the adversary selects a loss $\ell_t \in L$. Here X and L refer to the action and loss set respectively; we will generally consider the setting $X = \{1, \ldots, K\}$ and $L = [0,1]^K$ unless otherwise specified.

The learner selects their actions in accordance with some learning algorithm. Formally, a learning algorithm A is a collection of t functions $A_t : L^{t-1} \to X$, with A_t describing the action the learner takes at time t as a function of the losses $\ell_1, \ell_2, \ldots, \ell_{t-1}$. The regret of an algorithm A on a sequence of losses $(\ell_1, \ell_2, \ldots, \ell_T)$ is given by

$$\mathrm{Reg}(A, \ell) = \sum_{t=1}^{T} \ell(x_t, \ell_t) - \min_{x^\ddagger \in X} \sum_{t=1}^{T} \ell(x^\ddagger, \ell_t), \tag{4.1}$$

where in equation 4.1, $x_t = A_t(\ell_1, \ell_2, \ldots, \ell_{t-1})$. We are often concerned with the worst-case regret of a learning algorithm, which we denote via $\mathrm{Reg}(A) = \max_{\ell \in L^T} \mathrm{Reg}(A, \ell)$.

One algorithm with asymptotically optimal regret for the above problem is the Hedge algorithm of [90]. The Hedge algorithm with learning rate $\eta > 0$ can be defined via:

$$A_t(\ell_1, \ell_2, \ldots, \ell_{t-1})_i = \frac{\exp\left(-\eta \sum_{s=1}^{t-1} \ell_{s,i}\right)}{\sum_{j=1}^{K} \exp\left(-\eta \sum_{s=1}^{t-1} \ell_{s,j}\right)} \tag{4.2}$$

It is possible to show that the worst-case regret of Hedge is at most $O\left(\sqrt{T \log K}\right)$. We will need the following slightly finer-grained bound on the regret of Hedge.

Lemma 6. Let $\ell = (\ell_1, \ell_2, \ldots, \ell_T)$ be a sequence of losses such that each $\ell_t \in [0, \alpha_t]^K$. If we let A be the Hedge algorithm initialized with learning rate $\eta = \sqrt{\log K / \sum_{t=1}^{T} \alpha_t^2}$, then

$$\mathrm{Reg}(A, \ell) \le 2 \sqrt{\sum_{t=1}^{T} \alpha_t^2 \log K}.$$

Proof. This follows directly from Theorem 1.5 in [18]. □

4.2.2 Temporal feedback graphs

We now describe the variant of learning from experts with temporal feedback graphs. A temporal feedback graph S is a collection of subsets $S_t \subseteq [T - t]$ for each $t > T$, where the set S_t represents the set of losses visible to the learning algorithm at time t. Similar to our previous definition, an S-learning algorithm A is a collection of t functions $A_t : L^{S_t} \to X$, with each A_t describing the algorithm for mapping the visible losses to the action taken

in round t. The regret $\mathrm{Reg}(A, \ell)$ of A is defined identically as in equation 4.1, with the only difference being that $x_t = A_t(\ell_{S_t(1)}, \ell_{S_t(2)}, \ldots, \ell_{S_t(S_t(S))})$ (where $(S_t(1), S_t(2), \ldots, S_t(S_t(S)))$ is some explicit enumeration of S_t). Our goal throughout this paper is to design efficient S-learning algorithms A that minimize $\mathrm{Reg}(A)$.

In Appendix E.1 we consider the special subclass of transitive feedback graphs. We say a graph S is transitive if it is acyclic and has the property that if $s \to S_t$ and $r \to S_s$, then $r \to S_t$. That is, if the learner can see loss ℓ_s at round t, then they can also see all the losses they could see at round s. This class of graphs captures many natural applications (including the batched and delayed settings mentioned in the introduction).

4.3 Algorithms

4.3.1 A sub-optimal algorithm

We begin our discussion of S-learning algorithms with a remark about a natural algorithm which turns out to be surprisingly sub-optimal. This algorithm simply does the following: at round t, play the strategy suggested by Hedge (with some learning rate η) when run on all losses visible at time t. That is, in round t, play the strategy defined via

$$A_t(\ell_{S_t}) = \frac{\exp\left(-\eta \sum_{s \to S_t} \ell_{s,i}\right)}{\sum_{j=1}^{K} \exp\left(-\eta \sum_{s \to S_t} \ell_{s,j}\right)}$$

for some choice of learning rate $\eta > 0$. One can also think of this strategy as approximately best responding to the average loss visible at time t.

[72] show that in some bounded recall settings, this algorithm can incur regret that is linear in the time horizon T.

Lemma 7. Let $K = 2$, $M = T/10$ and let S be the associated bounded recall feedback graph (where $S_t = \{t - M, t - M + 1, \ldots, t - 1\}$). Then for any choice of learning rate η, the above algorithm A has worst-case regret $\mathrm{Reg}(A) = \Omega(T)$. In contrast, there exists an algorithm A^{alt} with $\mathrm{Reg}(A^{\mathrm{alt}}) = O(\sqrt{T})$.

Proof. See Theorem 2 of [72]. The sublinear regret algorithm A^{alt} simply restarts Hedge every M rounds. □

4.3.2 A better algorithm

In this section, we present an S-learning algorithm that does not have this detrimental trait, and indeed that we conjecture obtains within a constant factor of the optimal regret bound. As in the previous section, we will also use Hedge as a building block to construct this algorithm however, we will have to pay much more attention to the structural properties of the graph S.

For a temporal feedback graph S, we say a sequence of rounds $t_1, t_2, \ldots, t_w > T$ forms an order if for all $u < v$, $t_u \to S_{t_v}$. That is, every node later in the order can see the losses of all

nodes earlier in the order. An order C is maximal if no super-sequence of C forms an order. We will let $\mathcal{C} = \{C_1, C_2, \ldots, C_N\}$ denote the collection of all maximal orders in S.

Note that in the classic online learning problem, the entire temporal feedback graph S is just a maximal order of size T. We can therefore think of Hedge (or any standard learning algorithm) as being a learning algorithm tuned specifically for feedback graphs that are orders. Inspired by this, we introduce an algorithm for the general case based on the idea of running multiplicative weights in parallel for every order in the graph and taking an optimal convex combination of these sub-algorithms.

To define this optimal convex combination (and characterize the resulting regret bound we get), we need to solve the following convex program. We will call this program the upper bound (primal) program

$$\begin{aligned} \text{Minimize} \quad & \sum_{c=1}^{N} \sqrt{\sum_{t > C_c} \alpha_{c,t}^2} && (4.3)\\ \text{Subject to} \quad & \sum_{c=1}^{N} \alpha_{c,t} \geq 1 \quad \text{for all } t > T \\ & \alpha_{c,t} = 0 \quad \text{if } t \geq C_c \\ & \alpha_{c,t} \geq 0 \quad \text{for all } t > T, c > N \end{aligned}$$

Given a set of variables $\alpha_{c,t}$ satisfying the upper bound program equation 4.3, we can define our S-learning algorithm as in Algorithm 2. Intuitively, Algorithm 2 instantiates an instance of Hedge for every order in $\mathcal{C}$. In round t, the algorithm partitions the loss of that round among the orders that pass through t, where the algorithm corresponding to the order C_c receives a $\alpha_{c,t}$ fraction of the loss.

Note that Algorithm 2 is a valid S-learning algorithm, since the action taken at time t only depends on loss vectors ℓ_u for $t_u > S_t$. Let $UB(S)$ denote the optimal value of the upper bound program equation 4.3. The following theorem bounds the regret of Algorithm 2.

Theorem 9. If Algorithm 2 is run with an optimal solution $\alpha^\ddagger$ to the upper bound program equation 4.3, it incurs at most $O(UB(S)\sqrt{\log K})$ regret.

Proof. Fix an action $x^\ddagger \in K$, and consider the regret of this algorithm (which we will call A) against action $x^\ddagger$ for some fixed loss sequence ℓ. We can write this regret in the form:

$$\begin{aligned} \mathrm{Reg}_{x^\ddagger}(A, \ell) &= \sum_{t=1}^{T} \langle \ell_t, x_t - x^\ddagger \rangle \\ &= \sum_{t=1}^{T} \sum_{c=1}^{N} \alpha_{c,t}^\ddagger \langle \ell_t, x^{(c)} - x^\ddagger \rangle && (4.5)\\ &= \sum_{c=1}^{N} \left(\sum_{t=1}^{T} \alpha_{c,t}^\ddagger \right) \left\langle \ell_t, x^{(c)} - x^\ddagger \right\rangle : && (4.6) \end{aligned}$$

Algorithm 2 Algorithm for solving online learning with temporal feedback graphs

Input: A temporal feedback graph S with time horizon T and K actions, and a feasible solution $\{\alpha_{c,t}\} \subset [0,1]$ to the convex program equation 4.3.
For each $c \in [N]$, set $\eta_c = \sqrt{\log K / \sum_{t \in C_c} \alpha_{c,t}^2}$.
for each round $t = 1, \ldots, T$ **do**
  for each order C_c containing t **do**
    Let $t_1, t_2, \ldots, t_w = t$ be the prefix of C_c up to and including t.
    Let $x_t^{(c)} \in \Delta_K$ be the strategy defined via (for any $i \in [K]$):

$$x_{t,i}^{(c)} = \frac{\exp\left(-\eta_c \sum_{u=1}^{w-1} \alpha_{c,t_u} \ell_{t_u,i}\right)}{\sum_{j=1}^{K} \exp\left(-\eta_c \sum_{u=1}^{w-1} \alpha_{c,t_u} \ell_{t_u,j}\right)}. \tag{4.4}$$

  end for
  Play $x_t = \sum_{c:t \in C_c} \alpha_{c,t} x_t^{(c)}$.
  Receive loss vector ℓ_t (and loss $\langle x_t, \ell_t \rangle$).
end for

Here in equation 4.5 we have used the fact that $\sum_{c=1}^{N} \alpha_{c,t} \geq 1$ for any fixed $t \in [T]$ (since the $\alpha_{c,t}$ satisfy the convex program equation 4.3). Note that, by definition, $x_t^{(c)}$ is the output of an instance A_p of the Hedge algorithm initialized with learning rate η_c on losses $\ell^{(c)}$ given by $\ell_t^{(c)} = \alpha_{c,t} \ell_t$; moreover, the summand of the RHS of equation 4.6 corresponding to a given c is at most $\mathrm{Reg}(A_c, \ell^{(c)})$. Therefore, by Lemma 6, we have that

$$\mathrm{Reg}_{x^\ddagger}(A, \ell) \leq \sum_{c=1}^{N} \left(\sqrt{\log K \sum_{t=1}^{T} \alpha_{c,t}^2} \right) \leq \sqrt{\log K} \cdot \mathrm{UB}(S).$$

Since $\mathrm{Reg}(A, \ell) \leq \max_{x^\ddagger \in \Delta_K} \mathrm{Reg}_{x^\ddagger}(A, \ell)$, the conclusion follows. □

Theorem 9 leads to two natural questions:

1. First, although we can solve the convex program equation 4.3 in time polynomial in N, T, and K, one might notice that the parameter N is equal to the number of (maximal) orders in the temporal feedback graph S and can be very large (possibly exponential in T). In fact, even specifying a solution to the convex program and running Algorithm 2 requires time polynomial in N as written. Are there efficient learning algorithms (running in polynomial time in T and K) for this problem?
2. Secondly, does Algorithm 2 obtain the optimal regret bound for the S-learning problem? That is, must any S-learning algorithm A incur at least $\mathrm{UB}(S) \cdot \sqrt{\log K}$ regret on some sequence of losses?

The remainder of this paper will largely be concerned with providing answers to both of these questions (especially for the specific case of transitive feedback graphs S). For both of these questions, we will find it useful to examine the dual of the upper bound convex program, which we explore in the next section.

4.3.3 The dual convex program and an e cient learning algorithm

For multiple reasons, we will nd it easier to work with the dual of the upper bound convex program. In contrast with the original upper bound program, which is a minimization problem involving (up to) T N variables with T constraints, the dual program is a maximization problem on T variables with N constraints. We state this program (which we call the upper bound dual program) below.

$$\begin{aligned} \text{Maximize} \quad & \sum_{t=1}^{T} {}_t \\ \text{Subject to} \quad & \sum_{t>C_c} {}_t^2 \le 1 \quad \text{for all } c > N \\ & {}_t \ge 0 \quad \text{for all } t > T \end{aligned} \tag{4.7}$$

The following theorem proves that the program equation 4.7 is in fact the dual of the upper bound program.

Theorem 10. The optimal value of the upper bound dual program for temporal feedback graph S is equal to UB(S).[1]

Having established duality, complementary slackness allows us to infer strong structural statements about the optimal primal solution given an optimal dual solution.

Lemma 8. Let ‡ be an optimal solution to the upper bound dual program equation 4.7. Then there exists a primal solution ‡ to equation 4.3 such that, for any c > N where

$$\sum_{t>C_c} {}_t^{\ddagger} = 1, \tag{4.8}$$

we have that ‡c;t = 0 for all t > T , and for any c > N where

$$\sum_{t>C_c} {}_t^{\ddagger} \le 1, \tag{4.9}$$

we have that ‡c;t = c ‡t, for some constant c.

Given a solution ‡ to the upper bound dual, let C(‡) N denote the set of orders c where the equality equation 4.9 is tight. Lemma 8 implies that we only need to solve the upper bound program (and run Algorithm 2) for orders c > C(‡). If S(C(‡)) is small, this can be far more e cient than naively running Algorithm 2. In particular, since there are only T variables in the dual program, we would naively expect only T of the constraints to be binding, and therefore S(C(‡)) should generically equal T (in which case we have an e cient solution).

Of course, for many temporal feedback graphs S, the dual program is not generic, and it can be the case that many (or all of) the constraints bind at optimality. The following lemma

[1] For the sake of brevity, we defer many of the longer or more standard proofs to Appendix E.4.

shows that, even in such cases, it is possible to find an optimal solution of the primal program that is supported on at most T different orders.

Lemma 9. Given any optimal solution α^* to the upper bound dual program, there exists a subset $B \subseteq C(\alpha^*)$ with $|B| \leq T$ such that there exists a optimal solution α^* to the upper bound program with the property that $\alpha^*_{c,t} = 0$ if $C_c \notin B$.

Proof. By Lemma 8, we can restrict our attention to optimal solutions α^* where $\alpha^*_{c,t} = \gamma_c \alpha^*_t$ for $c \in C(\alpha^*)$ (for some collection of values $\gamma_c \geq 0$) and $\alpha^*_{c,t} = 0$ for $c \notin C(\alpha^*)$. Since $\sum_{t \in C_c} (\alpha^*_t)^2 = 1$ for all $c \in C(\alpha^*)$, we can rewrite the original upper bound primal program as the following linear program in the γ_c:

$$\begin{aligned} \text{Minimize} \quad & \sum_{c \in C(\alpha^*)} \gamma_c & (4.10) \\ \text{Subject to} \quad & \sum_{\substack{c \in C(\alpha^*) \\ \text{s.t. } t \in C_c}} \gamma_c \geq 1 - \varepsilon_t \quad \text{for all } t > T \\ & \gamma_c \geq 0 \quad \text{for all } c \in C(\alpha^*). \end{aligned}$$

But any extreme point of the linear program equation 4.10 will be the intersection of at least $|C(\alpha^*)|$ constraints, of which at most T are not of the form $\gamma_c \geq 0$. It follows that there is an optimal extreme point to this LP where at most T of the γ_c are non-zero, and hence a α^* with the property we described. □

Given a feedback graph S, we call a subset B of C a basis for S if there exists an optimal solution to the upper bound program supported entirely on orders $c \in B$. Lemma 9 shows that there always exists a basis of size at most T. Having an optimal solution with a small basis is valuable since it allows us to run Algorithm 2 more efficiently.

Lemma 10. Let α be a feasible point for the upper bound program equation 4.3 which is supported on a basis B of size $|B| = B$. Then we can run Algorithm 2 on α in time $O(BK)$ per iteration ($O(BKT)$ overall).

Proof. We modify Algorithm 2 by maintaining one instance of Hedge for each of the B non-zero orders (each of which takes $O(K)$ time to update per iteration). □

Together with Lemma 9, this provides us with a partial answer to our first question.

Corollary 3. For any feedback graph S, there exists an S-learning algorithm which runs in time $O(KT)$ per iteration and incurs regret at most $O(U\sqrt{S}) \sqrt{\log K}$.

The catch, of course, is that actually coming up with the learning algorithm of Corollary 3 involves computing an optimal basis B and associated primal solution α^*, which may not be computationally efficient. In Section E.1, we will see that for the important class of transitive feedback graphs, it is possible to compute this sparse primal solution efficiently.

4.4 Lower bounds for online learning with temporal feedback graphs

We now turn our attention to the second of the two questions: is our regret bound of $O(\sqrt{\log K})$ asymptotically tight? Throughout this section we will focus on the two-action setting ($K = 2$) for clarity of exposition. We expect that all lower bounds we present should extend to the K-action setting (with an additional factor of $\sqrt{\log K}$ in the lower bound).

4.4.1 A (naive yet efficient) lower bound program

We begin by examining what happens when we try to extend the original lower bound proof for the standard problem of learning with experts to the temporal feedback graph setting.

At a high-level, the original lower bound proof proceeds as follows. First, the adversary uniformly samples a bit $B \sim \{-1, 1\}$ unknown to the learner. The adversary then fixes an $\varepsilon \in (0, 1/2)$ and generates a sequence of i.i.d. random variables $X_1, X_2, \ldots, X_T$ where each $X_t \sim \{0, 1\}$ is drawn independently from a Bernoulli$(\frac{1}{2} - B\varepsilon)$ distribution. The adversary then chooses losses defined via $\ell_t = (X_t, 1-X_t)$.

Now, the learner cannot learn much about the random bit B until they have seen at least $(1-\varepsilon^2)$ samples from Bern$(\frac{1}{2} - B\varepsilon)$, and hence until they have seen at least $(1-\varepsilon^2)$ loss vectors. But during that time, they incur ε regret per round. Altogether, this implies that the adversary can force the learner to incur a total regret of at least $\varepsilon \max(1-\varepsilon^2, T)$. Picking $\varepsilon = 1/\sqrt{T}$, we obtain the well-known $\sqrt{T}$ lower bound.

We will start our discussion of lower bounds for the problem of SI-learning with a similar approach. Again, the adversary will begin by uniformly sampling a bit $B \sim \{-1, 1\}$. But now, to account for the additional potential asymmetry in the feedback structure across rounds, the adversary will select a sequence of scales $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_T \in [0, \varepsilon]$, and sample the r.v. X_t from the distribution Bern$(\frac{1}{2} - B\varepsilon_t)$ (for some global scale $\varepsilon > 0$). Finally, the adversary again sets $\ell_t = (X_t, 1-X_t)$.

Intuitively, we can understand the performance of this strategy as follows. As long as the learner cannot figure out the value of B based on the losses observable at round t, the learner will incur an expected regret of $\varepsilon\varepsilon_t$ in that round. In order to prevent the learner from figuring out the value of B in round t, the amount of information leaked by the losses in S_t about B should be small. This quantity is roughly proportional to $\sum_{s \in S_t} \varepsilon_s^2$ (in particular, observing ℓ_s leaks $\varepsilon\varepsilon_s$ bits of information about B). This motivates writing down the following convex program equation 4.11, which we call the lower bound program

$$\begin{aligned} \text{Maximize} \quad & \sum_{t=1}^{T} \eta_t && (4.11) \\ \text{Subject to} \quad & \sum_{t' > S_{t^{\circ}}} \eta_{t'}^2 \le 1 \quad \text{for all } t^{\circ} > T \\ & \eta_t \ge 0 \quad \text{for all } t > T \end{aligned}$$

We will write $LB(S)$ to denote the optimal value of the lower bound program. Note that this program shares many structural similarities with the upper bound dual program — the main difference is that whereas in the upper bound dual program equation 4.7, the constraints bound the sum of the squares of the variables over all orders in S, here the sums are over all neighborhoods $S_{t^{\circ}}$ in S. Since every order C_c is a subset of the neighborhood $S_{t^{\circ}}$ of its last element, the constraints of the lower bound program are stronger than that of the upper bound dual (and so $LB(S) \le UB(S)$, as we would expect).

The following theorem formalizes the above intuition and shows that $\frac{1}{4}LB(S)$ is indeed a valid lower bound for the S-learning problem (up to a universal constant factor).

Theorem 11. *Every S-learning algorithm A must incur worst-case regret $\mathrm{Reg}(A) \ge LB(S)/100$.*

One nice property of the lower bound program equation 4.11 is that it is polynomial-sized, and can therefore be optimized efficiently. Another feature of this program is that, due to its similarity with the upper bound dual program, it immediately gives us the following multiplicative bound on its optimality.

Theorem 12. *Let S be a temporal feedback graph where every neighborhood is contained in the union of at most R orders C_c. Then $UB(S) \le LB(S) \cdot R$ (and hence the regret bound of Algorithm 2 is within R of optimal).*

Proof. Let $\eta^{\ddagger}$ be an optimal solution to the upper bound dual program. Note that in such a graph, η_t would satisfy $\sum_{t > S_{t^{\circ}}} \eta_t^2 \le R$, and therefore setting $\eta_t = \eta_t^{\ddagger} / R$ forms a feasible solution to equation 4.11 with value at least $UB(S)/R$ and at most $LB(S)$. The conclusion follows. □

There are some classes of feedback graphs where the parameter R in Theorem 12 is $O(1)$ and hence where we can immediately conclude that Algorithm 2 is nearly optimal (one interesting such class of graphs is those arising from bounded recall, where each S_t is itself an order). The downside, however, is that in general the gap between $LB(S)$ and $UB(S)$ can be quite large (at least $\Omega(T^{1/4})$ — see Appendix E.2). In the next section we present a stronger lower bound that comes from a much larger convex program.

4.4.2 The independent set lower bound

One of the main reasons the lower bound in the previous section is lax is every loss which is visible to the learner at round t reveals independent information about B_t. But since not all of these rounds can observe each other, this is inherently a little wasteful — it would be better if we could "reuse" the same signal about B_t across multiple different rounds.

To implement this idea, let $I(S)$ denote the collection of independent sets of the (undirected version) of S. The adversary then begins by generating a random variable with some bias $B w_I$ for every independent set $I \in I(S)$. Finally, to generate the loss at round t, the adversary combines the random variables for sets in I containing t (by e.g. taking their majority). This motivates us to write down the following convex program:

$$\begin{aligned} \text{Maximize} \quad & \sum_{t=1}^{T} \sqrt{\sum_{I \in I(S) \text{ s.t. } t \in I} w_I^2} \\ \text{Subject to} \quad & \sum_{I \in I(S) \text{ s.t. } I \cap S_t \neq \emptyset} w_I^2 \leq 1 \quad \text{for all } t > T \end{aligned} \tag{4.12}$$

We call the above program the *independent set (lower bound) program*, and denote its optimal value by $ILB(S)$. Note that if we restrict the set of feasible set to w that take nonzero values on single nodes, then the independent set program reduces exactly to our original lower bound program, and thus implies that $ILB(S) \leq LB(S)$. But more importantly, this more general program still gives a lower bound on the achievable regret of any algorithm.

Theorem 13. *Every S-learning algorithm A must incur worst-case regret $\mathrm{Reg}(A) \geq ILB(S)/50$.*

We leave it as an interesting open question to characterize the gap between $ILB(S)$ and $UB(S)$. We conjecture that this gap is at most a universal constant (thus showing that both Algorithm 2 and this lower bound are optimal up to constant factors).

Conjecture 1. *There exists a constant β such that for any temporal feedback graph S, $UB(S) - ILB(S) \leq \beta$.*

In particular, in Appendix E.1 we employ a variant of this technique to prove a tight lower bound for all transitive graphs, providing partial evidence for this conjecture.

Chapter 5

Sampling Polytopes with Riemannian HMC: Faster Mixing Via The Lewis Weights Barrier

Abstract

We analyze Riemannian Hamiltonian Monte Carlo (RHMC) on a manifold endowed with the metric de ned by the Hessian of a convex barrier function and apply it to sample a polytope de ned by m inequalities in R^n. The advantage of RHMC over Euclidean methods such as the ball walk, hit-and-run and the Dikin walk is in its ability to take longer steps. However, in all previous work, the mixing rate of RHMC has a linear dependence on the number of inequalities. We introduce a hybrid of the Lewis weight barrier and the standard logarithmic barrier and prove that the mixing rate for the corresponding RHMC is bounded by $O\hat{}m^{1-3}n^{4-3}\bullet$, improving on the previous best bound of $O\hat{}mn^{2-3}\bullet$ (based on the log barrier). This continues the general parallels between optimization and sampling, with the latter typically leading to new tools and requiring more re ned analysis. To prove our main results, we overcomes several challenges relating to the smoothness of Hamiltonian curves and self-concordance properties of the barrier. In the process, we give a general framework for the analysis of Markov chains on Riemannian manifolds, derive new smoothness bounds on Hamiltonian curves, a central topic of comparison geometry, and extend self-concordance theory to the in nity norm, which gives sharper bounds; these properties all appear to be of independent interest.

5.1 Introduction

Generating nearly uniform random samples from a high-dimensional polytope is a fundamental algorithmic problem with a rich history and powerful applications, notably including the only known fully polynomial-time approximation schemes for computing a polytope's volume. All e cient algorithms known for this problem work by designing a Markov chain whose

stationary distribution is uniform over the polytope and showing that it mixes in a small number of steps.

In this paper, our main result is that we can construct such a Markov chain with an improved bound on its mixing time. For a polytope given by m linear inequalities in R^n, we describe chain that mixes in $O^*(m^{1/3}n^{4/3})$ steps, improving on the best previous bound of $O^*(mn^{2/3})$. This allows us to approximate the volume within relative error using $O^*(m^{1/3}n^{4/3}/\varepsilon^2)$ steps, which is a similar improvement over the best existing bound of $O^*(mn^{2/3}/\varepsilon^2)$.

5.1.1 Background and Related Work

In their seminal work [91], Dyer, Frieze and Kannan gave the first polynomial-time algorithm for this problem, as well as for the more general problem of sampling from a convex body specified by a membership oracle. The Markov chain in their algorithm was a *grid walk*, which takes steps along the edges of the graph obtained by intersecting the convex body with a discrete grid supported on Z^n for some $1/\mathrm{poly}^*(n)$. This graph is heavily dependent on the coordinate system its diameter is proportional to the diameter of the convex body, and its conductance can be arbitrarily small if the convex body is scaled so that is very long in some directions but short in others. However, they showed that, if one changes to a basis in which the convex body is appropriately well-rounded, the grid walk mixes in polynomial time and that one can use a random sample from the grid to obtain a one from the convex body.

The polynomial for the mixing time in [91] was quite large, and a sequence of later papers improved this by modifying the Markov chains and refining the analysis. Because one often wants to draw many samples from the body, these papers typically provide two bounds on the number of steps required: a bound when starting from an arbitrary point and including the cost of any preprocessing; and a bound when given a *warm start*, where the preprocessing has already been performed and the starting point is drawn from a distribution that is not too far from uniform.

In [92], Kannan, Lovász, and Simonovits showed that a *ball walk* whose steps are chosen uniformly from a Euclidean ball around the current point mixes in $O^*(n^3)$ steps from a warm start and $O^*(n^5)$ steps from an arbitrary starting point and including preprocessing. Later, Lovász and Vempala [93] studied the *hit-and-run* walk, which chooses a line in a random direction from the current point and then picks the next point randomly from the intersection of this line with the body, and they showed it also mixed in $O^*(n^3)$ steps from a warm start but needed only $O^*(n^4)$ steps for first sample and preprocessing. These algorithms work on general convex bodies presented by oracles, but like the grid walk, they are strongly coordinate dependent, and they thus require strong additional assumptions about the coordinate system. In particular, analyses of these algorithms typically assume that the body is close to being isotropic, i.e., that the covariance matrix of a random sample from the body is approximately the identity, and applying these algorithms to more general bodies requires costly preprocessing.

The dependence on the coordinate system in the aforementioned Markov chains comes from the dependence of the transition probabilities on the extrinsic geometry of the ambient Euclidean

space. The impact of this extends beyond the overhead from the isotropy requirements. The geometry of the ambient space does not incorporate any information about how close a point is to the boundary, which typically leads to difficulties making progress with steps near the boundary. For example, if one is running a ball walk with step radius δ on an n-dimensional cube, and the current point is some distance $d \gg \delta$ from one of the corners, a random point from the radius δ ball will lie outside the cube with probability exponentially close to 1, so naively trying random points until obtaining one in the cube would take a large number of tries. Moreover, even if one could sample a random point in the intersection of the ball with the cube, restricting the step to points inside the cube would distort the stationary distribution, and it would no longer be uniform. Remedying such difficulties typically involves (depending on the paper) some combination of taking smaller steps, enlarging the convex body (and failing if the walk ends up at a point outside the original body), and employing rejection sampling or a Metropolis filter to correct the stationary probabilities, all of which increase the required number of steps.

For polytopes specified by an explicit collection of linear constraints, one can use the barrier functions employed by interior point methods to design random walks whose steps depend only on the intrinsic geometry of the polytope and are independent of the basis chosen for the ambient space. The idea behind these random walks is to use the Hessian of the barrier function to define a local norm/Riemannian metric on the interior of the polytope and specify the steps in terms of the resulting geometry. This mitigates some of the problems described above and has led to Markov chains whose mixing times grow with the number of constraints but depend more mildly on the dimension.

In the first such work, Kannan and Narayanan [94] introduced the Dikin walk and gave a mixing time bound of $O(mn)$ from a warm start for a polytope with m facets in $\mathbb{R}^n$. This walk is similar to the ball walk, but it chooses its steps from Dikin ellipsoids, which are balls with respect to the Hessian of the standard logarithmic barrier function on the polytope. In [95], Laddha, Lee, and Vempala studied the analogous walk with respect to any self-concordant barrier and showed that it mixes in $O(\nu n)$ steps, where ν is a parameter they called the barrier parameter. By bounding this parameter for a different barrier function (a variant of a barrier due to Lee and Sidford [57]), they obtained an improved mixing rate bound of $O(n^2)$.

In 2017, Lee and Vempala [96] reduced the mixing rate to $O(mn^{3-4})$ using a process they called the geodesic walk. Similar to the Dikin Walk, the steps are constructed using the Hessian of a barrier function. However, instead of using the Hessian to define a Euclidean ellipsoid, they use it to define a Riemannian metric, and they then solve a differential equation in each step to follow geodesics on the resulting manifold. These geodesics tend to avoid the polytope's boundary, which allows longer steps in each iteration.

In 2018, Lee and Vempala [97] improved this to $O(mn^{2-3})$ using Riemannian Hamiltonian Monte Carlo (RHMC) [98], which is the class of processes we will use in this paper. While there is a large literature on using RHMC and related methods to sample smooth densities [99-103], there are relatively few provable results about applying it in constrained non-smooth settings like polytope sampling. Roughly speaking, this improvement over the geodesic walk came from RHMC's ability to avoid the use of a Metropolis filter, which the geodesic walk requires in order to obtain the correct stationary distribution (even when the target

distribution is uniform). RHMC chooses its trajectories according to a di erential equation that, remarkably, yields a reversible random walk with the desired stationary distribution, thus eliminating the need for a Metropolis lter and allowing greater progress in each step.

While barriers with better parameters have led to improved mixing times for the Dikin walk, obtaining similar improvements for the geodesic walk or RHMC have remained elusive, and improving upon the $\tilde{O}(mn^{2-3})$ bound attained by RHMC using the standard log barrier has been a major open problem for the past 5+ years.

The core issue that prevents the authors of [96] and [97] from using other barriers in place of the log barrier is that the geodesic walk and RHMC use their barrier functions in a fundamentally di erent way from how they are used in the Dikin walk. The Dikin walk, like the interior point methods for which self-concordant barriers were originally de ned, uses the Hessian of the barrier function at each point to specify an ellipsoid centered at the point and contained in the polytope and chooses its next iterate from this ellipsoid.

The geodesic walk and RHMC, however, use the barrier function to de ne the local geometry of a manifold, and they take a step by simulating the trajectory of a particle according to a corresponding second-order di erential equation. The solution depends on the geometry at every point of the trajectory, rather than just at the point where the particle was at the beginning of the iteration. As such, the steps of the random walk depend on the geometric structure at all of the points of the trajectory, and analyzing them requires one to understand how the locally-de ned structure at each point relates to those at other nearby points. This leads to a dependence on higher-order derivatives of the barrier function than the ones that self-concordance was designed to control. As a result, self-concordance by itself does not seem to be su cient in this setting, and analyzing these walks requires the authors of [96] and [97] to rely on new but ad hoc arguments tied to speci c properties of the logarithmic barrier.

Year	Algorithm	Steps
1997 [92]	Ball walk$^{\#}$	n^3 (+ n^5)
2003 [93]	Hit-and-run$^{\#}$	n^3 (+ n^4)
2009 [94]	Dikin walk	mn
2017 [96]	Geodesic walk	mn^{3-4}
2018 [97]	RHMC with log barrier	mn^{2-3}
2020 [95]	Weighted Dikin walk	n^2
2021 [104]	Ball walk$^{\#}$	n^2 (+ n^3)
This paper	RHMC with Hybrid barrier	$m^{1-3}n^{4-3}$

Table 5.1: The complexity of uniformly sampling a polytope from a warm start. All algorithms have a logarithmic dependence on the warm start parameter and each uses $\tilde{O}(n)$ bit of randomness. The entries marked $^{\#}$ are for general convex bodies presented by oracles, while the rest are for polytopes. The additive terms are pre-processing costs for rounding the polytope.

5.1.2 Background on Riemannian Hamiltonian Monte Carlo

The motivation for RHMC comes from the Hamiltonian formulation of classical Newtonian mechanics. Hamiltonian mechanics parameterizes a physical system in terms of a position vector x and a corresponding momentum vector v (which is also referred to as velocity in some prior work on sampling polytopes with RHMC). The physics of the system are encoded in its Hamiltonian $H(x, v)$, which is simply the energy of the system written as a function of x and v, and its time evolution is determined by Hamilton's equations

$$\begin{aligned} \frac{dx}{dt} &= \frac{\partial H}{\partial v}(x, v) \\ \frac{dv}{dt} &= -\frac{\partial H}{\partial x}(x, v). \end{aligned}$$

With the appropriate choice of H, these reproduce Newton's laws of motion, but they also generalize quite broadly, including to Riemannian manifolds.

In RHMC, one defines a Markov chain by choosing a Hamiltonian that appropriately encodes the target distribution. At each step, the Markov chain chooses a random momentum vector and then finds the next point by numerically solving a differential equation to follow the trajectory given by Hamilton's equations.

One can show that the value of the Hamiltonian and the volume element in the space of pairs (x, v) are conserved along the trajectory, which can be used to show that the trajectories are preserved by time reversal. One can then use this to show that, if one uses the Hamiltonian defined below, the marginal distribution of x will converge to the desired target distribution without requiring a Metropolis filter. (See [98] for the derivation for general RHMC and [97] for the specific class of Hamiltonians given below.)

More precisely, let the Hamiltonian at a point $x \in \mathbb{R}^n$ for a vector $v \in \mathbb{R}^n$ be defined as

$$H(x, v) = f(x) + \frac{1}{2} v^T g^{-1}(x) v + \frac{1}{2} \log \det g(x)$$

where $g(x)$ is a positive definite matrix defining a Riemannian metric at each point x as $\|u\|_{g(x)} = \sqrt{u^T g(x) u}$, and the target density to be sampled is proportional to e^f restricted to the support of g. One step of RHMC consists of the following: first pick v from the Gaussian $N(x, g(x)^{-1})$. Then for time τ follow the Hamiltonian curve jointly on (x, v):

$$\begin{aligned} \frac{dx}{dt} &= \frac{\partial H}{\partial v}(x, v) = g^{-1}(x) v \\ \frac{dv}{dt} &= -\frac{\partial H}{\partial x}(x, v) = \nabla f(x) - \frac{1}{2} \mathrm{tr}\left(g(x)^{-1} Dg(x)\right) + \frac{1}{2} Dg(x)\left(\frac{dx}{dt}, \frac{dx}{dt}\right). \end{aligned} \tag{5.1}$$

The final x at time τ is the sampled point from the Markov Kernel. A natural choice for the metric g turns out to be the Hessian of a self-concordant barrier function inside the polytope P. The standard logarithmic barrier, $\phi(x) = \sum_{i=1}^m \log(a_i^T x - b_i)$, was used in [97] to prove that the resulting RHMC mixes in mn^{2+3} steps, where the polytope is defined by the inequalities $\{a_i^T x \le b_i\}_{i=1}^m$. Improving on this bound is our motivating open problem.

5.1.3 Results

Algorithmic results. Our main algorithmic result in this paper is the construction of a random walk based on RHMC for sampling and approximating the volume of polytopes using only $\widetilde{O}(m^{1+3}n^{4+3})$ steps. Using a Hamiltonian based on the standard logarithmic barrier yields a mixing rate that depends linearly on m, the number of inequalities. We improve on this by developing a theoretical framework for designing and analyzing barriers for RHMC.

Our framework is motivated by the ways in which the requirements of RHMC differ from those of the Dikin walk. In the case of Dikin walk, we care about how much volume two nearby ellipsoids defined by our metric have in common, which can be controlled given that the first derivative of the metric is bounded, a property that self-concordant barriers possess [105]. Namely, the derivative of the Hessian of a self-concordant barrier is controlled by the Hessian itself, which can be seen as a property of the metric $g = \nabla^2$,

$$\|v\|_{g(y)} \le \|v\|_{g(x)} \cdot \|y - x\|_{g(x)},$$

where $Dg(x)[v]$ is the directional derivative of g along direction v.

On the other hand, to define the Markov kernel for RHMC, one reparameterizes the open set inside the polytope by the Hamiltonian trajectories, which map the tangent space of the current point to the manifold. This means the density of the RHMC Markov kernel depends on the distortion properties of this map, such as how fast the Hamiltonian curves with different initial conditions converge or diverge. Therefore, showing that this density is Lipschitz is linked to the geometry imposed by the Hessian of the barrier, and analyzing the second-order ODE regarding RHMC demands estimates on more than just the first order-derivative of the metric. As a result, the existing notions of self-concordance are insufficient for bounding the spectral gap of RHMC.

In this paper, we define a stronger notion, third-order ℓ_a -self-concordance, that is stringent enough to guarantee the stronger properties required by RHMC, but we show that it still admits a construction that improves upon the logarithmic barrier. Third-order ℓ_a -self-concordance strengthens standard self-concordance in two ways: it controls the higher order derivatives of g up to third order; and, instead of the local ellipsoidal norm $\|\cdot\|_{g(x)}$ that is conventionally used in self-concordance, we measure the spectral change of the metric in the local infinity norm $\|\cdot\|_{x,a}$, which we define next.

Definition 10. For arbitrary vector v we define the local norm $\|v\|_{x,a}$ at point $x \in P$ as the maximum relative change of the distance of x to an arbitrary facet of the polytope after taking step v. Formally:

$$\|v\|_{x,a} := \|S_x^{-1} A v\|_\infty ;$$

where the polytope is defined by the inequalities $Ax \le b$ and S_x is the diagonal matrix whose entries are the slacks of the linear constraints at point x, i.e. $(S_x)_{ii} = a_i^\top x - b_i$.

An intuitive description of $\|\cdot\|_{x,a}$ is via its unit ball; namely, $\|\cdot\|_{x,a}$ is the unique norm whose unit ball is the symmetrized polytope $P \cap (2x - P)$ around x, as illustrated in Figure F.1 in Appendix F.1. ($2x - P$ is the reflection of P around x.) Using this generalized notion, we are

not only able to control the change of density of the RHMC kernel (Section F.4), but also prove the stability of the Hamiltonian curves (Lemma 13), which is required for bounding the conductance.

De nition 11 (Third-order ℓ_a-self-concordance) We say that is c-third-order ℓ_a-self-concordant if its Hessian $g = \nabla^2$ obeys

$$\begin{aligned} \nabla_v \nabla_{x;a} g(x) &\preceq D g(x)[v], \\ \nabla_v \nabla_{x;a} \nabla_z \nabla_{x;a} g(x) &\preceq D^2 g(x)[v,z] + \nabla_v \nabla_{x;a} \nabla_z \nabla_{x;a} g(x), \\ \nabla_v \nabla_{x;a} \nabla_z \nabla_{x;a} \nabla_u \nabla_{x;a} g(x) &\preceq D^3 g(x)[v,z,u] + \nabla_v \nabla_{x;a} \nabla_z \nabla_{x;a} \nabla_u \nabla_{x;a} g(x), \end{aligned} \tag{5.2}$$

Furthermore, we say that is a (c, ν)-third-order ℓ_a-self-concordant barrierif in addition to the estimates equation 5.2 the norm of its gradient in the local norm is bounded as

$$(D \nu(x))^\top g^{-1} D \nu(x) \le \nu,$$

where D refers to the Euclidean gradient of .

Here $\preceq$ is the Löwner ordering between matrices ignoring logarithmic factors. Our second major contribution is to construct a barrier for polytopes that satis es third-order ℓ_a-self-concordance. Namely, we construct a hybrid barrier inside the polytope based on the Lewis weight barrier

$$_p(x) = \log\det(A_x^\top W_x^{1-2/p} A_x), \tag{5.3}$$

where W_x is a diagonal matrix whose diagonal entries are the Lewis weights of the rescaled matrix $A_x = S_x^{-1}A$, which we de ne in Section 5.1.4.

The hybrid barrier for a polytope is then de ned as a combination of the Lewis weight barrier and the log barrier. This combination is necessary so that the resulting manifold with metric $g = \nabla^2$ satis es a suitable isoperimetric inequality.

De nition 12 (Hybrid barrier) . We de ne the hybrid barrier inside a polytope $Ax \le b$ as

$$(x) = \left(\frac{m}{n}\right)^{\frac{2}{p-2}} \log\det A_x^\top W_x^{1-2/p} A_x + \frac{n}{m} \sum_i \log(s_i), \tag{5.4}$$

where $s_i = a_i^\top x - b_i$ are the slacks at point x. We denote the normalizing factor of by $_0 = \left(\frac{m}{n}\right)^{\frac{2}{p-2}}$.

Our main theorem is a bound on the mixing rate of RHMC with this hybrid barrier.

Theorem 14 (Mixing) . Given a polytope P, let be the distribution with density proportional to $e^{-(x)}$ over the open set inside P. Then, RHMC with stationary distribution on the manifold of the open set inside P equipped with metric $g = \nabla^2$ de ned by the Hessian of the hybrid barrier with $p = 4(1-\log(m))$ has mixing rate bounded by

$$\Omega\left(\min\left(n^{-2/3}, n^{-1/3} n^{5/9} m^{-1/9}, n^{1/3} m^{-1/6}, n^{-1/3} m^{-1/3}\right)\right).$$

In particular, for the uniform distribution over P (i.e. $= 0$), the mixing rate is $\Omega(m^{-1/3} n^{-1/3})$.

Specically, the Markov chain starting at $_0$ reaches $_t$ with TV-distance at most $$ to the target after

$$O\%mn^{1-3}n^{4-3}\log\hat{}M \sim \bullet^2 \log\hat{}M \bullet \log\log\hat{}M \sim \bullet^2\check{Z} \tag{5.5}$$

steps, where $M < \sup_{x>P} \frac{d\ _0\hat{}x\bullet}{d\ \hat{}x\bullet}$ and O in Equation equation 5.5 hides $\mathrm{polylog}\hat{}m\bullet$ factors.

Note that without a warm start, the $\log\hat{}M\bullet$ dependence in Theorem 14 could be another factor of n to the mixing time. However, applying the Gaussian Cooling framework [106] extended to manifolds [97] lets us sample from m for any $$ without a warm start penalty, and also allows us to compute the volume of the polytope without a signicant overhead. (Recent work [107] shows how to leverage the Gaussian Cooling method in more general metrics and for avoiding the warm start penalty for sampling also,)

Corollary 4 (Any start; Volume). For the manifold Gaussian Cooling scheme in [97] with the hybrid barrier equation 5.4 applied to sample from the density $\hat{}x\bullet$ inside a given polytope starting from $\arg\min\ \hat{}x\bullet$, the total number of RHMC steps for any $C0$ is bounded by

$$O\%mn^{1-3}n^{4-3}\log\hat{}1\sim\bullet^2 \log\log\hat{}1\sim\bullet^2\check{Z};$$

Moreover, to compute the integral of e in the polytope and in particular the volume of the polytope up to multiplicative error 1 $\circ e$, the total number of RHMC steps is bounded by $O\hat{}m^{1-3}n^{4-3}\sim\circ^2\bullet$.

This improves on the previous best bound of mn^{2-3} due to [97] based on the standard logarithmic barrier.

Geometric results. The proof of Theorem 14 requires the development of several technical ingredients. We summarize a few that are likely to be of independent interest.

The rst is a new isoperimetric inequality for this hybrid barrier, which we prove in Section G.7.4. (See Section F.1.3 for the denition of the isoperimetric constant.)

Theorem 15. [Isoperimetry of the hybrid barrier] Let g be the metric corresponding to the Hessian of the hybrid barrier, with support given by a polytope dened by $$ inequalities in R^n. Then for $C0$, the distribution with density proportional to e has isoperimetric constant at least

$$\max\tilde{}\circ\frac{1}{\bar{n}}\hat{}\frac{n}{m}\bullet^{\frac{1}{p\ 2}};\ \mathrm{poly}\hat{}\frac{1}{4\sim p\quad 1}\bullet^{\circ}\ \ \bullet:$$

Moreover, in order to use the abstract framework that we introduce in this work to control the change of the RHMC Markov kernel, we establish the third-order $\grave{}_a$ -self-concordance of the hybrid barrier dened in Equation equation 5.4, which we prove in Section F.2.

Theorem 16 (Third-order $\grave{}_a$ -self-concordance of the hybrid barrier). The hybrid barrier, dened in equation 5.4, is a $\hat{}c_2;\ _0n\bullet$-third-order $\grave{}_a$ -self-concordant barrier where c_2 $\mathrm{poly}\hat{}\frac{1}{4\sim p\ 1}\bullet$. In particular, with our choice $p\quad 4\ \hat{}1\sim\log\hat{}m\bullet\bullet$ in Theorem 14 we have $c_2\quad\mathrm{polylog}\hat{}m\bullet$.

These estimates allow us to prove important smoothness properties of certain quantities on the manifold that we are interested in. As far as we know, this is the rst proof of such

regularity for higher order derivatives of the Lewis weight barrier. The main challenge we face to prove the third-order ℓ_a-self-concordance is estimating higher order derivatives of the Lewis weights in the PSD cone, which we do in Sections F.2 and F.6.

We remark that our notion of third-order ℓ_a-self-concordance is a strengthening of a well-studied notion in differential geometry. We show in Lemma?? that $\|v\|_{x;a} \le \|v\|_g$ when g is the metric derived from the hybrid barrier. This implies the following corollary, which says that we can obtain the same third-order derivative estimates of g when the norm $\|\cdot\|_{x;a}$ is replaced by $\|\cdot\|_g$ in Theorem 16.

Corollary 5 (Calabi estimates for the hybrid barrier). The metric g of the hybrid barrier equation 5.4 satisfies Calabi estimates up to third order, namely

$$c_2\|v\|_g \le Dg(v)[v] \le c_2\|v\|_g$$
$$c_2\|v\|_g\|z\|_g \le D^2g(v,z)[v] \le c_2\|v\|_g\|z\|_g$$
$$c_2\|v\|_g\|z\|_g\|u\|_g \le D^3g(v,z,u)[v] \le c_2\|v\|_g\|z\|_g\|u\|_g,$$

where $c_2 = \mathrm{poly}(\frac{1}{4-p}, 1)$. khasha: turn the paranthesis to brackets for derivative

These type of estimates on the derivatives of the metric are known as the Calabi estimates in the differential geometry literature [108, 109]. It turns out that the Calabi-type estimates in Corollary 5 are insufficient to improve the mixing rate, which is why we develop the third-order ℓ_a-self-concordance for our hybrid barrier to further exploit the randomness of the Hamiltonian curves.

5.1.4 Technical Overview

Mixing and Conductance. Our general approach to bounding the mixing rate is based on bounding the conductance [110]. The standard approach to bounding the conductance of geometric walks of this type is to show an isoperimetric inequality for the underlying metric space and then prove that steps of the random walk behave well with respect to the underlying metric. Formally, we show two properties for the manifold M obtained by equipping the interior of the polytope P with the metric $g(x)$:

- Isoperimetry. The target density $e^{-f(x)}$ has a good isoperimetry constant on M.
- One-Step Coupling. The one-step distributions of the Markov chain given two close-by points x_0, x_1 on the manifold are close in TV-distance. Namely, for some parameter $A > 0$, after excluding a tiny set $S^c \subset M$, given any two points $x_0, x_1 \in S$ with $d(x_0, x_1) \le B$ we show

$$TV(T_{x_0}, T_{x_1}) \le 0.01, \tag{5.6}$$

  where T_x denotes the Markov kernel starting from x.

Isoperimetry. The log barrier metric gives an isoperimetric coefficient of $\frac{1}{\sqrt{m}}$, which leads to a factor of m in the conductance. In principle, this can be improved to $O(n)$ by using a barrier with barrier parameter $O(n)$, as the general bound on the isoperimetry is

Figure 5.1: Family of Hamiltonian curves $_s\hat{}t\bullet$ all ending in y with starting point varying from x_0 $_0\hat{}0\bullet$ to x_1 $_{s\text{œ}}\hat{}0\bullet$, where $_s\hat{}0\bullet$ is a geodesic in s.

$1\tilde{}^{o}\ \bar{}$ for any strongly self-concordant barrier with barrier parameter [111]. The barrier parameter is an indicator of how well the ellipsoids of the metric de ned by the barrier approximate the symmetrized polytope $P\ 9\ 2x\quad P$ around x. khasha: reference to the barrier parameter section While the universal and entropic barriers have $O\hat{}n\bullet$, they are expensive to compute. The LS barrier [57] has $O\hat{}n\bullet$ while being e cient to compute. However, as we will see in more detail, as far as we know, the derivatives of the metric of the LS barrier are not smooth" enough khasha: elaborate more on this nonsmoothness in most directions, which means we would have to take rather small steps while running RHMC.

We will prove that the hybrid barrier has signi cantly better isoperimetry (Thm. 15) than the log barrier while maintaining su cient smoothness.

Smoothness of Hamiltonian Curves and Comparison Geometry. The starting point of our analysis is the fact that one can look at the ordinary di erential equation of RHMC in Equation equation 5.1 as a second-order ODE on the manifold M of the open set inside the polytope with metric g. We will introduce this alternative form shortly. Looking at the Markov Kernel T_{x_0} of RHMC for a xed point x_0, the randomness to de ne this kernel comes from the initial velocity v_0, which can be viewed as a vector on the tangent space of x_0 on the manifold M distributed as a standard Gaussian with respect to the local metric, namely $N\hat{}0; g\hat{}x\bullet\ ^{1}\bullet$ in the Euclidean chart. In order to show the one-step coupling (Lemma 5.6) for the Markov kernel of RHMC, we bound the di erence between the densities $T_{x_0}\hat{}y\bullet$ and $T_{x_1}\hat{}y\bullet$ at a given point y on the manifold. These densities are the pushforwards of the Gaussian density in the tangent space of x_0 and x_1 respectively, onto the manifold through the Hamiltonian map $Ham\ \hat{}x_0; v_{x_0}\bullet$ for some xed time , which maps the initial velocity v_{x_0} to the solution of the ODE $y\quad x\hat{}\ \bullet$ at time . The key to bound the change of density is to understand how the Hamiltonian curves vary as we change the initial point from x_0 to x_1 for a xed destination y, given the particular geometry imposed by our hybrid barrier inside a polytope. In fact, understanding the extremal scenarios of the behavior of geometric

quantities on a certain class of manifolds is the topic of Comparison Geometry [112] [113] [114]. In particular, to argue that the Hamiltonian curve changes su ciently slowly, we need the metric g of the manifold and its derivatives to be stable". The simplest form of stability of the metric is the so-called self-concordance property, namely, g is self-concordant if the derivative of g^x• in a unit direction in the tangent space is controlled by g itself. This type of self-concordance for the rst derivative of the metric is already known for the Lewis weight barrier [2]. An important part of our contribution is to build an abstract framework which shows that self-concordance of the metric up to third-order derivatives is su cient for characterizing the stability of Hamiltonian curves (see section F.4).

It turns out that we need to bound the rate of change of the density only for Hamiltonian curves with typical values of the initial velocity and can ignore sets with small probability when bounding the conductance. However, the typical value of $\|v\|_g$ for a Gaussian vector $v \sim N(0, g^{-1}(x))$ in the tangent space is still quite large to improve the mixing time, hence it is ideal to show self-concordance with respect to a better norm. Taking a closer look into proving self-concordance for the Lewis weight khasha: talk about Lewis weights barrier before this barrier, we need to control the change of the Lewis weights multiplicatively, which in nitesimally is equivalent to bounding the norm of the Jacobian of the function that maps the location $x \in P$ to the logarithm of its Lewis weights. Our key observation here is that when $p < 4$, we can bound the $\|\cdot\|_{\infty,a}$ norm of this Jacobian by constant, which results in a stability argument for the Lewis weights with respect to $\|\cdot\|_{x,a}$. Building upon this estimate, we show that our hybrid barrier is rst-order ℓ_a-self-concordant. This is favorable for us as the typical value of $\|v\|_{x,a}$ for $v \sim N(0, g(x)^{-1})$ is much smaller than $\|v\|_g$. In fact, we show that the $\|\cdot\|_{x,a}$ norm of the tangent vector to the RHMC curve remains small for all small enough positive times with high probability. Following this idea, to lift the ℓ_a-self-concordance of the barrier to second and third order derivatives, we need to control the $\|\cdot\|_{\infty,a}$ norm of operators that arise from higher order derivatives of Lewis weights, and use them to estimate the derivatives of the Lewis weights barrier by analysis on the PSD cone. Our framework for obtaining these estimates is summarized in Section F.2.2 and elaborated upon in Section F.6. khasha: can't import lemmas here since we haven't de ned the variables We use these estimates to derive self-concordance estimates for g.

Lewis weights stability. Lewis weights of A_x can be de ned as the solution of the following optimization problem (for more detail, see Section F.1.2):

$$w_x < \operatorname{argmax}_{w \in \mathbb{R}^n_{>0}} \log\det(A_x^{\top} W^{1-2/p} A_x) - \|1 - 2/p\|_1 \|w\|_1.$$

In particular, in Section F.6 we obtain the following infinity norm estimates on higher-order derivatives of the Lewis weights:

$$\frac{1}{(4-p)\,1^{\bullet}}\|u\|_{x,a}\,\|z\|_{x,a}\,w_x \preceq Dw_x(z,u)\preceq \frac{1}{(4-p)\,1^{\bullet}}\|u\|_{x,a}\,\|z\|_{x,a}\,w_x,$$

$$\frac{1}{(4-p)\,1^{5}}\|u\|_{x,a}\,\|z\|_{x,a}\,w_x \preceq D^2w_x(z,u)\preceq \frac{1}{(4-p)\,1^{5}}\|u\|_{x,a}\,\|z\|_{x,a}\,w_x,$$

$$\frac{1}{(4-p)\,1^{7}}\|u\|_{x,a}\,\|v\|_{x,a}\,\|z\|_{x,a}\,w_x \preceq D^3w_x(v,z,u)\preceq \frac{1}{(4-p)\,1^{7}}\|u\|_{x,a}\,\|v\|_{x,a}\,\|z\|_{x,a}\,w_x,$$

where we used $\preceq$ to show Löwner ordering up to universal constants. These estimates are indeed used to drive infinity norm estimates on the derivatives of the Lewis weights barrier (see Section F.2 and F.6.)

Isoperimetry vs Smoothness. We show stronger stability results for the derivatives of the metric of the p-Lewis weights barrier with $p \approx 4$ based on the $\|\cdot\|_{x,a}$ norm. However, for small p the ellipsoid of the p-Lewis weights barrier does not approximate the symmetrized polytope as well as larger p; in particular, a large subset of the ellipsoid lies outside the symmetrized polytope. This necessitates a larger barrier parameter and implies a smaller isoperimetric constant. To construct a barrier that is smooth enough along typical directions whose ellipsoids also approximate the symmetrized polytope more accurately, we go back to an idea of Vaidya from optimization and use a hybrid barrier by "regularizing" the Lewis weight barrier for $p \approx 4$ with the standard log barrier; we observe that penalizing the p-Lewis weights barrier with the log barrier improves its barrier parameter, while not affecting the smoothness of the barrier since the log barrier is already third-order $\|\cdot\|_x$-self-concordant. Therefore, the particular choice of our barrier is essential to simultaneously guaranteeing third-order $\|\cdot\|_a$-self-concordance and good isoperimetry.

Hamiltonian Curves and Variations. To see the high-level idea of how we show the one-step coupling of the Markov kernel, let the curve γ_s parameterized by $s > 0, s^{\bullet}$ be a length-minimizing geodesic connecting $x_0 \gamma_0 = x_0$ to $x_1 \gamma_{s^{\bullet}} = x_1$ with distance $d(x_0, x_1)$; geodesics are generalization of straight lines in the Euclidean space to arbitrary manifolds (see Section F.7 for more background.) Suppose now that running the Hamiltonian ODE with initial location $x_0 = p$ and initial velocity v_{x_0} up to time τ takes us to a point y on the manifold. As we start moving toward x_1 on the geodesic γ_s parameterized by $s > 0, s^{\bullet}$, we consider the variation of the initial Hamiltonian curve; namely a family of Hamiltonian curves parameterized by s, where the s-curve starts from point γ_s, perhaps with a different initial velocity v_{γ_s}, but ends up to the same destination y at time τ. The geodesic γ_s from x_0 to x_1 and the corresponding Hamiltonian curves are illustrated in Figure 5.1. Looking at the the value of the density $T_s(y)$ at point y after taking one step of the Markov chain starting from γ_s, we see that it depends on two major components: (1) the Gaussian density of the initial velocity v_{γ_s} which is proportional to $\exp\left(-\frac{\|v_{\gamma_s}\|_g^2}{2}\right)$, and (2) the determinant of the Jacobian or the differential of the map from the initial velocity v_{γ_s} to the destination point y, denoted by $J_y^{v_{\gamma_s}}$. Therefore, to study how quickly the density $T_s(y)$ changes from x_0 to x_1, we need to study the rate of change of the initial velocities v_{γ_s} and the Jacobians $J_y^{v_{\gamma_s}}$; the latter will depend on the rate

of change of the Ricci tensor on the manifold. This necessitates studying the variations of Hamiltonian curves, which we de ne next.

As we mentioned earlier, one can identify the location variable in the Hamiltonian ODE equation 5.1 as a point on the manifold M with metric g, and the velocity variable v as a vector in the tangent space of x, $T_x\hat{}M \bullet$. Then, the Hamiltonian ODE in Equation equation 5.1 can be written as a second-order ODE on the manifold M using the covariant derivative, illustrated in Lemma 11. For background on Riemannian geometry and covariant di erentiation, we refer the reader to Appendix F.7.

Lemma 11. The Hamiltonian ODE in Equation 5.1 can be written using the covariant derivative of the manifold in a simpli ed form:

$$\copyright_{\oe t\bullet}\ \oe t\bullet \quad \hat{}\ \hat{}t\bullet\bullet: \tag{5.7}$$

where $\copyright$ is the covariant derivative and $\hat{}x\bullet$ is the bias (drift) vector eld of the Hamiltonian curve:

$$\hat{}x\bullet < g\ {}^{1}Df\ \hat{}x\bullet \quad \frac{1}{2}g\hat{}x\bullet\ {}^{1}\,\mathrm{tr}\ g\hat{}x\bullet\ {}^{1}Dg\hat{}x\bullet\ ;: \tag{5.8}$$

In the above notation, $\mathrm{tr}\ g\hat{}x\bullet\ {}^{1}Dg\hat{}x\bullet$ is a vector whose ith entry is $\mathrm{tr}\ g\hat{}x\bullet\ {}^{1}D_i g\hat{}x\bullet$. See Appendix F.7 for a proof of Lemma 11. The above ODE equation 5.7 for Hamiltonian curves is similar to the second order ODE for geodesics; for the latter the bias vector is zero, i.e., the geodesic Equation is given by [115]

$$\copyright_{\oe t\bullet}\ \oe t\bullet \quad 0: \tag{5.9}$$

In physics, the Hamiltonian ODE in Equation 5.7 models the motion of a particle on a manifold acting under a force eld devised by . Next, we de ne the notion of a family of Hamiltonian curves and an operator $\hat{}t\bullet$ which plays an important role in the study of variations of Hamiltonian curves.

De nition 13 (Family of Hamiltonian curves). We say $\%\!\!\%_s\hat{}t\bullet\check{Z}$ is a family of Hamiltonian curves ending at some xed y whose starting point varies from $x_0 \quad {}_0\hat{}0\bullet$ to $x_1 \quad {}_{s_1}\hat{}0\bullet$ if for every xed time $0 \mathrm{B} s \mathrm{B} s_1$, ${}_s\hat{}t\bullet$ is a Hamiltonian curve in t, and ${}_s\hat{}0\bullet$ as a function of s is a geodesic on M from x_0 to x_1. Unless speci ed otherwise, ${}_s\hat{}t\bullet$ refers to a curve in t for a xed s, and $\oe_s t\bullet \quad @_s\hat{}t\bullet$ refers to its derivative.

De nition 14 (Operators and M_x). At any point $x > M$, we de ne the operator M_x as

$$\text{¦ } u > T_x\hat{}M \bullet; M_x\hat{}u\bullet < \copyright_u\ \hat{}x\bullet;$$

where $\copyright$ is the covariant derivative on the manifold and is the Hamiltonian bias. Given the Hamiltonian curve $\hat{}t\bullet$, we de ne the operator $\hat{}t\bullet \quad \hat{}\ \hat{}t\bullet;\ \oe t\bullet\bullet$ on the tangent space $T_{\hat{}t\bullet}\hat{}M \bullet$ as

$$\hat{}t\bullet < \quad \hat{}\ \hat{}t\bullet\bullet < R\hat{}:;\ \oe t\bullet\bullet\ \oe t\bullet \quad M_{\oe t\bullet}:$$

where R is the Riemann tensor.

Similar to Jacobi fields for geodesics (see section F.7.3), for a given family of Hamiltonian curves $\hat{\gamma}_s(t)$, one can write a second order ODE for the variational vector field $\hat{J}(t) = \frac{d}{ds}\hat{\gamma}_s(t)$ along the Hamiltonian curve, which depends on operator (see Appendix?? for the proof):

Lemma 12 (ODE for Hamiltonian fields). Given a family of Hamiltonian curves $\{\hat{\gamma}_s(t)\}$, the vector field $\hat{J}(t) = \left(\frac{\partial}{\partial s}\hat{\gamma}_s(t)\right)_{s=0}$ is characterized by the following second order ODE:

$$\hat{J}''(t) = \hat{\Omega}(t)\hat{J}(t), \tag{5.10}$$

where $\hat{\Omega}(t)$ is defined in Definition 14. We refer to $\hat{J}$ as a Hamiltonian field.

For variation of Hamiltonian curves, the log determinant of the Jacobian of the Hamiltonian map $J_y^{v_s}$ can be characterized by a weighted integral of the trace of $\hat{\Omega}(t)$. Therefore, to study the rate of change of $\det(J_y^{v_s})$ as we move from x_0 to x_1, we need to study the change of $\mathrm{tr}(\hat{\Omega}(t))$ along the variation of Hamiltonian curves $\hat{\gamma}_s(t)$, which in turn depends on the rate of change of the Ricci tensor and the trace of operator M_x. These ideas are formalized as the (R_1, R_2, R_3)-normality of the Hamiltonian curve in the definition below.

Definition 15. [Normal Hamiltonian curves] We say a Hamiltonian curve $\hat{\gamma}(t)$ is (R_1, R_2, R_3)-normal up to time T if for all $0 \le t \le T$ we have the following:

- Bound on the Frobenius norm of $\hat{\Omega}$ (with respect to the metric g): $\|\hat{\Omega}(t)\|_F \le R_1$.
- For any parameterized family of curves $\hat{\gamma}_s(t)$ such that $\hat{\gamma}_0(t) = \hat{\gamma}(t)$ for all times $0 \le t \le T$, then for all such t, the derivative of $\mathrm{tr}(\hat{\Omega}(t))$ with respect to $z = \frac{d}{ds}\hat{\gamma}_s(t)$ satisfies

$$\left|\frac{d}{ds}\mathrm{tr}(\hat{\Omega}(t))\right| \le \left(R_2\|\hat{\gamma}'(t)\|_g + \|\nabla_z \hat{\gamma}'(t)\|_g\right).$$

- For $\hat{\gamma}(t)$ defined as the parallel transport of $\hat{\gamma}'(0)$ along $\hat{\gamma}(t)$: $\|\hat{\gamma}(t)\|\|\hat{\gamma}'(t)\|_g \le R_3$.

Parallel transport of a vector on the manifold is a generalization of shifting vectors in Euclidean space, using the covariant derivative of the manifold (see Appendix F.7 for the rigorous definition.)

In order to show the (R_1, R_2, R_3)-normal property for the family of Hamiltonian curves, we need to define a more fundamental regularity condition for the Hamiltonian curves which states that both $\|\cdot\|_g$ and $\|\cdot\|_{x,a}$ norms remain small for the tangent vector along the Hamiltonian curve.

Definition 16 (Nice Hamiltonian curve). We say a Hamiltonian curve $\hat{\gamma}(t)$ is $(c, \bar{n})$-nice if for $0 \le t \le T$:

$$\|\gamma'(t)\|_g \le c\sqrt{\bar{n}},$$
$$\|\gamma'(t)\|_{\hat{\gamma}(t),a} \le c.$$

Our $(c, \bar{n})$-niceness framework is a simpler and allows us to work with any third-order self-concordant barrier and avoids the technical machinery of auxiliary functions on curves used in [97], which needs additional parameters and only works for the specialized case of log barrier.

The second major part of our contribution in this paper is that we relate this abstract notion of (R_1, R_2, R_3)-normality to the notion of third-order ℓ_a-self-concordance. Our framework can potentially be reused on other manifolds and distributions.

Theorem 17 (From third-order ℓ_a-self-concordance to Hamiltonian normality.) Given a Hessian manifold defined by the metric $g = \nabla^2$ inside the polytope for $(c_2, \bar{\eta}_0)$-third-order ℓ_a-self-concordant barrier , define a Hamiltonian curve (t) by the ODE in Equation equation 5.7 with target log density f . If is $(c,)$-nice, then it is also (R_1, R_2, R_3)-normal with parameters

$$R_1 \quad c_3 c^2 \quad \frac{}{\bar{\eta}_0} \quad \bar{n}, \quad R_2 \quad c_3 c^2 \quad \frac{}{\bar{\eta}_0} \, n, \quad R_3 \quad c_3 c^2 \quad \bar{n} \quad nc \quad n \, c \quad \frac{}{\bar{\eta}_0},$$

where $c_3 \quad \text{poly}(c_2)$.

Proof. The proof follows from combining Lemmas 103, 109, and 117. □

In order to show the closeness of one step distributions between x_0 and x_1, we need the (R_1, R_2, R_3)-normality for the family of Hamiltonian curves $(_s(t))$ for all $0 \le s \le$ as we defined in 15. Therefore, we need to show that the $(c,)$-niceness property is stable for a third-order ℓ_a-self-concordant barrier. We show this in Lemma 13, which we prove in a more technical form as Lemma 121.

Lemma 13 (Stability of norms). In the same setting as Theorem 17, given a family of Hamiltonian curves $_s(t)$ for which $_0(t)$ is $(c,)$-nice for

$$\le \frac{1}{\sqrt{c^2 \quad \bar{\eta}_0 n^{1/4}}},$$

then $(_s(t))$ is a $(O(c),)$-nice family of Hamiltonian curves in the intervals $(0,)$.

Chapter 6

Convergence of the Riemannian Langevin Algorithm

Abstract

We study the Riemannian Langevin Algorithm for the problem of sampling from a distribution with density e^{-f} with respect to the natural measure on a manifold with metric g. We assume that the target density satisfies a log-Sobolev inequality (with respect to the metric) and prove that the manifold generalization of the Unadjusted Langevin Algorithm converges rapidly to the target e^{-f} for Hessian manifolds. This allows us to reduce the problem of sampling non-smooth constrained densities in $\mathbb{R}^n$ to sampling smooth unconstrained densities over appropriate manifolds, while needing access only to the gradient of the log-density, and this, in turn, to sampling from the natural Brownian motion on the manifold. Our main analytic tools are (1) an extension of self-concordance to manifolds, and (2) a stochastic approach to bounding smoothness on manifolds. A special case of our approach is sampling isoperimetric densities restricted to polytopes by using the metric defined by the logarithmic barrier.

6.1 Introduction

Sampling is a fundamental problem with connections to many areas of mathematics and algorithms. A natural approach to sampling from a desired distribution is *diffusion*, i.e., a stochastic process that converges to the target density in continuous time. This is a subject of classical study with many powerful and surprising theorems [116]. Sampling algorithms, which are inherently discrete, are also a major subject due to their generality and plethora of applications [117–119]. The connections between the two are still being developed and the following is not well-understood:

> When does the rapid convergence of a diffusion process result in an efficient sampling algorithm?

In recent years, progress has been made on this via the Langevin Diffusion and associated

Langevin algorithm, both in Euclidean space. For a target density proportional to $e^{-f(x)}$, Langevin Diffusion (LD) is the following stochastic process[1]:

$$dX_t = -\nabla f(X_t)dt + \sqrt{2}dB_t, \tag{6.1}$$

where B_t is the standard Wiener process. LD converges to the target density in continuous time. The rate of convergence in relative entropy (KL divergence) can be bounded for densities satisfying a Log-Sobolev Inequality (LSI, defined in section G.7.4) with parameter $A > 0$:

$$H(p_t) \le e^{-2At} H(p_0).$$

When the target e^{-f} is strongly logconcave and gradient Lipshitz, the work of [120, 121] showed that the simple Euler discretization converges rapidly. This is given by

$$x_{k+1} = x_k - h\nabla f(x_k) + \sqrt{2h}Z$$

where $Z \sim N(0, I)$ is a standard Gaussian. The convergence rate was refined and improved in many subsequent papers [100, 122]. It was later shown that this discretization gives an efficient algorithm for any density proportional to e^{-f} where ∇f is L-Lipshitz, with the rate of convergence polynomial in L, n and $1/\varepsilon$ [123]. Li and Erdogdu [124] extended this framework to the manifold given by a product of spheres, and [125] to more general manifolds under some assumptions. These results suggest a more comprehensive picture is waiting to be discovered.

Diffusion can be generalized naturally to Riemannian manifolds. Given a manifold M defined by a local metric $g(x)$ for each $x \in M$, Riemannian Langevin Diffusion (RLD) is the following stochastic process (in Euclidean coordinates):

$$dX_t = -D(g^{-1}(X_t)) - g^{-1}(X_t)\nabla f(X_t))dt + \sqrt{2g^{-1}(X_t)}dB_t, \tag{6.2}$$

where dB_t is again the standard Wiener process. The above equation has stationary distribution $e^{-f}dx$. If $f = 0$, then its soltion is the canonical Brownian motion on the manifold with metric g. The benefit of this process is that one can choose the local metric to facilitate convergence and hence potentially extend the method to more general densities. This leads to our main motivating questions:

1. Under what conditions does the discretization of RLD, the Riemannian Langevin Algorithm, converge?
2. Can non-smooth and/or constrained distributions be sampled by mapping to appropriate manifolds and using RLA? This is analogous to optimization, where non-smooth convex optimization can be mapped to smooth convex optimization via self-concordant barriers (the Interior-Point Method [105]).

For the first question, it is useful to consider RLD as a process on a manifold (the equivalence is shown formally in Lemma 186).

[1] We use D for Euclidean derivative and reserve ∇ for manifold derivative; when the manifold is R^n, they coincide.

De nition 17 (Riemannian Langevin Di usion (RLD)). Given a metric g in the Euclidean chart x >R^n, let M be the manifold with measure dv_g ^x•, whose density with respect to the Lebesgue measure in the Euclidean chart is » det ^g•S Given a distribution with density e ^F, Riemannian Langevin Di usion is the solution to the following SDE whose stationary distribution is dv_g.

$$dX_t = \hat{\odot}\, \hat{g}^{-1}(X_t)\bullet\bullet \odot F(X_t)\bullet\bullet\, dt + \sqrt{2}\, g^{-1}(X_t)\bullet dB_t: \tag{6.3}$$

Here © denotes the divergence with respect to the manifold and it is applied separately to each row of a matrix (see Sec. G.7 for background on manifolds). Convergence in continuous time holds for any target density satisfying LSI and for any Riemannian metric. The SDE in Equation equation 6.3 can also be written as

$$dX_t = \odot F(X_t)\bullet dt + dX_t;$$

where X_t is the Brownian motion on M .

Theorem 18 (RLD Convergence). [116] Suppose that the density f ^x• dx e ^F dv_g ^x• has log-Sobolev constant A0 with respect to the metric g. Then, the distribution _t obtained by RLD at time t C0 satis es

$$H(\_t\bullet \mid e^{-2t} H(\_0\bullet:$$

When the metric g is de ned by the Hessian of a strictly convex function as g^x• D^2 ^x•, then the corresponding process is also called Mirror Langevin Di usion , as it can be viewed as passing to the dual via the mirror map y D ^x•. This is discussed in several recent papers [126 128], which prove convergence under rather strong assumptions.

The simple discretization of RLD is the Riemannian Langevin Algorithm, de ned next.

De nition 18 (Riemannian Langevin Algorithm (RLA)). To sample from the distribution dv_g ^x• over a stochastically complete manifold M , we de ne the following chart-independent iterative process on the manifold for a xed step size

$$y = \exp_{x_k}\hat{}\ \big(-\odot F(x_k)\bullet\bullet\big)$$
$$x_{k+1} = B(y; \bullet$$

where B^x_0; t• samples from Brownian motion on the manifold, starting from x_0 after time t. The stochastic completeness can be obtained from a lower bound on the Ricci tensor, which holds naturally for the manifolds that we consider 194

In words, at any point x, the algorithm travels for some time along the geodesic with initial gradient © F^x• to reach a point y, and then performs a Brownian motion on the manifold starting at y for time (we describe Brownian motion on a manifold in Section G.7.2). We note that in this paper we assume that we have access to an oracle that can sample Brownian motion on a manifold starting from a given point up to a given time increment (similar to [127].) One approach to implement such an oracle is to further consider a discretization of the SDE corresponding to the Brownian motion over the manifold. In this work, however, we look at the oracle as black-box and leave its implementation to future work. It is worthy to

mention that our Riemannian Langevin algorithm is only one possibility of discretizing the continuous SDE among other choices.

Such geodesic walks" have been considered in the literature, e.g., in [96, 125, 129], under various additional assumptions or with more algorithmic modi cations. [96] applies a geodesic walk to sample uniformly from a polytope by mapping to the manifold given by a strictly convex barrier function, but their process requires a Metropolis step which explicitly computes the probability of transition and uses a lter to ensure the desired stationary distribution. [129] show that geodesic ball" walk converges rapidly to the manifold measure for geodesically strictly convex subsets of a positive curvature manifold subject to additional conditions. [124] obtains strong estimates of the rate of convergence for some classes of functions on the manifold de ned by a product of spheres.

The discretization of Mirror Langevin Di usion has been an active research topic in recent years. It was rst proposed as a sampling algorithm by [126] and later studied by [127, 128, 130, 131]. The results of [128, 130] assume that the convex function whose Hessian de nes the metric satis es a property they callmodi ed self-concordance Roughly speaking it requires that the square-root of the metric changes slowly (in Frobenius norm), namely the change is bounded by a constant times the change in the gradient in Euclidean norm. They also require the function of interest to be smooth and strongly convex. Even with these assumptions, convergence to the target distribution with vanishing bias was only established recently [128], using an analysis technique developed in [132]. Unfortunately, the modi ed self-concordance parameter is not a ne-invariant (even though the process itself is!) and is unbounded for the log barrier, even in two dimensions [128]. The work of [127] shows how to use self-concordance of the convex barrier function together with the assumptions that the target f is convex and Lipshitz to reduce the sampling problem to that of simulating Brownian motion in the barrier metric. Relaxing their assumptions, while maintaining an a ne-invariant analysis is a motivating open problem.

6.1.1 Results

A natural candidate for the condition on the metric is self-concordance, as usually de ned in optimization, i.e., the third directional derivative of the convex function can be bounded by a constant times the second directional derivative to the power of 3/2, an a ne-invariant condition [105]. Unfortunately, this does not imply that the square-root of the inverse of the metric satis es self-concordance (or changes slowly), leading to the above mentioned modi ed self-concordance assumption. We get around this by a natural extension of standard self-concordance to manifolds.

Here we prove a general theorem for Hessian manifolds based on its self-concordance parameters of the metric g and smoothness parameters of f. We de ne second-order self-concordance for a metric, an extension of the standard de nition to one higher derivative. Our main result can be interpreted as a reduction from the problem of sampling a general density on a manifold to sampling Brownian motion on the manifold. As a special case of this theorem, it follows that for smooth functions f satisfying LSI with respect to the log barrier metric, RLA can sample from the density proportional to e^{-f} restricted to polytopes with

the Langevin algorithm. More generally, our framework can be applied to sample from a distribution restricted to a convex body which satisfies a log-Sobolev inequality, when the body can be equipped with a suitable hessian metric. We concretely instantiate our framework for polytopes. Another example is the space of PSD matrices equipped with the metric from the log determinant barrier which also satisfies our second-order self-concordance condition. Previous work was able to sample smooth, strongly logconcave densities or smooth, unconstrained densities satisfying isoperimetry, or convex and Lipshitz densities. To describe the results more precisely, we first define manifold self-concordance. Recall that the local norm of a vector v in a manifold with metric g is given by $\|v\|_g^2 = v^T g v$.

Definition 19. (Second-order self-concordance) Let $M \subseteq \mathbb{R}$ be a four-times differentiable function and $g(x) = D^2 f(x)$. Then g is α_1-self-concordant if for any $v \in \mathbb{R}^n$

$$\alpha_1 \|v\|_g \nabla^3 f(v) \preceq \alpha_1 \|v\|_g^3.$$

We say that g is $(\alpha_1, \alpha_2, \alpha_3)$-second-order self-concordant if in addition to the above (standard) self-concordance, for any $v, w \in \mathbb{R}^n$, it satisfies

$$\langle \nabla g(v), \nabla g^{-1} w \rangle \preceq n \alpha_2^2 \|v\|_g \|w\|_g$$

and

$$\alpha_3^2 \|v\|_g \|w\|_g \nabla^2 g(v, w) \preceq \alpha_3^2 \|v\|_g \|w\|_g g.$$

The last condition on the second derivative of the Hessian is similar in spirit to-regular self-concordance (Def 2.1 in [NN98]), although our norm is more natural for the Hessian metric setting.

Remark 2. As we will show, for the manifold defined by the log barrier metric in a polytope, the parameters $\alpha_1, \alpha_2, \alpha_3$ are all bounded by absolute constants (14).

We can now state our main theorem. For convenience, we assume that all self-concordance and Lipshitzness parameters are at least 1. The assumption $\eta \le 1$ below is for simplicity of exposition, and one can simply replace with $\min(1, \eta)$.

Theorem 19 (General Hessian Manifolds) Let M be a Hessian manifold with $(\alpha_1, \alpha_2, \alpha_3)$-second-order self-concordant metric. Consider the distribution $d\pi(x)$ on M with density e^{-F} w.r.t to the manifold volume measure. Assume that π satisfies the log-Sobolev inequality with parameter $\rho \ge 1$. Let the function F be L_2-gradient Lipschitz, L_3-Hessian Lipschitz on M. Then, there exist universal constants c_1, c_2, such that after k steps of the Riemannian Langevin Algorithm with step-size $\eta \le 1$, starting from a distribution with density ρ_0 which is square integrable with respect to the manifold measure and has finite second moment (with respect to an arbitrary, fixed point on M), the distribution ρ_k satisfies

$$H(\rho_k) \le e^{-\frac{3}{16}\eta k} H(\rho_0) + c_1 \eta n^{5/2} \left(\alpha_1^2 + \alpha_2^2 + \alpha_3^2\right) \left(L_2 + \sqrt{n} L_2^2 + n L_3\right). \tag{6.4}$$

Hence, for any target accuracy $\varepsilon > 0$, after $k = O\left(\frac{1}{\eta} \log \frac{2H(\rho_0)}{\varepsilon}\right)$ steps of RLA with

$$\eta \le c_2 \frac{\varepsilon}{n^{5/2} \left(\alpha_1^2 + \alpha_2^2 + \alpha_3^2\right) \left(L_2 + \sqrt{n} L_2^2 + n L_3\right)}, \tag{6.5}$$

the distribution ρ_k satisfies $H(\rho_k) \le \varepsilon$.

The theorem recovers as special cases some results in the literature[2] It also extends the reach of Langevin style algorithms to constrained distributions as illustrated in the corollaries below. We note that our gradient and Hessian Lipschitness assumptions are the natural ones on the manifold.

Remark 3. In general, we work with Gibbs form of the distribution $e^{-F}\,dv_g(x)$, which can also be presented by its density with respect to the Lebesgue measure in the Euclidean chart as $e^{-f}\,dx$, where

$$f = F - \frac{1}{2}\log\det(g(x)).$$

The LSI parameter is determined by the distribution and the metric, hence remains the same property for either of the representations with f or F with respect to the same metric g.

Polytope with log barrier. We apply our result to the special case when the manifold is the open set inside a polytope $Ax > b$, and the metric at point x is given by the Hessian $D^2\,\phi(x)$, where ϕ is the logarithmic barrier function:

$$\phi(x) = -\sum_i \log(a_i^T x - b_i).$$

The Hessian can be written as $D^2\,\phi(x) = A_x^T A_x$, where A_x is equal to the matrix A whose ith row is reweighted by $1/(a_i^T x - b_i)$. We denote the leverage score vector of A_x by σ_x, i.e.,

$$\sigma_x = \mathrm{diag}\left(A_x(A_x^T A_x)^{-1}A_x^T\right).$$

Then, Brownian motion with this geometry is given by

$$dX_t = \sqrt{2}(A_t^T A_t)^{-1}dB_t + \frac{1}{2}(A_t^T A_t)^{-1}A_t^T \sigma_t\,dt,$$

where we denoted A_{X_t} and σ_{X_t} by A_t and σ_t for brevity.

We show that second-order self-concordance holds for the geometry induced by the log barrier: (proof in Appendix G.9.1)

Lemma 14. For a polytope $Ax < b$ with the geometry of log barrier, the metric is $(2, 2, \sqrt{6})$-second-order self-concordant.

Corollary 6. Let the manifold M be the open set inside a polytope with the log barrier metric. Consider the distribution with density e^{-f} with respect to the Lebesgue measure inside the polytope s.t. the corresponding function F defining the density e^{-F} on M is L_2-gradient Lipschitz and L_3-Hessian Lipschitz (i.e. taking the metric into account), and also satisfies a log-Sobolev inequality with constant with respect to the log barrier metric. Assuming $L_2, L_3 \geq 1$ and $\varepsilon, B \leq 1$ for simplicity of exposition, there is a constant c s.t., RLA with step-size

$$\eta = \frac{c}{n^{5.2}L_2 + \sqrt{n}L_2^2 + nL_3};$$

starting from a distribution ν_0, samples from a distribution whose KL-divergence with respect to the target ν is at most ε, and the number of iterations bounded by $\tilde{O}\left(\frac{1}{\eta\varepsilon}\log\left(\frac{2H(\nu_0\|\nu)}{\varepsilon}\right)\right)$.

[2] The main theorem of [125] claims a result in a similar spirit with additional parameters K_1, K_2, K_3, K_3 as well as curvature bounds; we avoid these parameters here and do not see how they could be bounded.

To make the above statement explicit in terms of the original function f, we bound all the required smoothness parameters for the log barrier in a polytope and obtain the following result. Specifically, in Appendix G.9.5, we work with a slightly weaker assumption than the manifold gradient and Hessian Lipschitz parameters (L_2, L_3) to obtain a bound depending on the conventional gradient and Hessian Lipschitz parameters of f. We note that while we focus on a polytope, the techniques extend to more general convex bodies with analogous barriers (e.g., as investigated in [133]).

Corollary 7. Suppose the target distribution is $d\pi(x) \propto e^{-f} dx$ for f inside a polytope $Ax \le b$ of diameter R, assuming $R \ge 1$ for simplicity, such that f is ℓ_2-gradient Lipschitz, and ℓ_3-Hessian Lipschitz. Moreover, suppose π satisfies a log-Sobolev inequality with constant $B \ge 1$ with respect to the log barrier metric. For any $\varepsilon > 0$, the query complexity of RLA to reach at most ε error in KL divergence starting from density π_0 is $O\left(\frac{1}{\eta} \log \frac{2H(\pi \| \pi_0)}{\varepsilon}\right)$, for step size

$$\eta \le \frac{1}{n^{7.2} \left(n^{5.2} \ell_2 R + \sqrt{n \ell_2^2 R^2 + n \ell_3 R^3}\right)^2}.$$

To prove these corollaries, we bound the self-concordance parameters of the log barrier metric and bound the manifold smoothness parameters of F in terms of the smoothness parameters of the given function f (Section G.8). Also, we separately bound the manifold Lipschitz coefficients of F for the case of sampling from the uniform distribution with log barrier in the polytope G.9.2, which improves the query complexity with the dependence on the diameter (parameter R) being dropped.

Remark 4. It is known that logconcave measures with bounded support satisfy LSI with respect to the Euclidean metric [116]. While showing an LSI for log concave distributions inside the polytope with respect to the log barrier can be tricky in the general, we define a perturbed version of the target distribution which (1) is close to the initial distribution in total variation distance and (2) satisfies LSI with respect to the metric. To this end, we only need to speed up the decay of the distribution close to the boundary of the polytope (see Appendix G.6).

6.1.2 Basic notation

We use D for the Euclidean derivative, $\partial_i, \partial_{ij}$ for Euclidean partial derivatives, ∇ (or grad) for manifold derivative, and d for the differential over the manifold. For a background on manifolds, see appendix G.7. So, e.g., $D_y \log \det g(x) = \langle g^{-1}(x), Dg(x)[y] \rangle$ where $g(x)$ is a matrix defined by x and Dg is a third-order tensor, so that the RHS becomes a vector. The directional derivative in the direction y would be $D_y \log \det g(x)[y] = \mathrm{Tr}[g^{-1}(x) Dg(x)[y]]$. We interchangeably refer to the gradient and Hessian of F over the manifold by $\mathrm{grad} F, \mathrm{Hess} F$ and $\nabla F, \nabla^2 F$ respectively.

We use π for the steady state or target distribution. The KL-divergence (relative entropy) of a distribution ρ with respect to another distribution π is denoted by $H(\rho \| \pi)$ and defined as

$$H(\rho \| \pi) = E_{x \sim \rho}[\log(\rho(x)/\pi(x))] = \int \rho(x) \log \frac{\rho(x)}{\pi(x)} \, dv_g(x).$$

The relative Fisher information is denoted by$I(\cdot)$ and defined as

$$I(\rho) = \int \|\nabla \log(\rho/\pi)\|^2 \, d\rho(x). \tag{6.6}$$

We use $\Phi_t(x)$ to denote the point on the manifold obtained after going on a geodesic starting from x after time t. We also use $\Phi_t \# (\cdot)$ to denote the push forward measure of under the map Φ_t.

6.2 Approach: Bounding the discretization error using a stochastic process

Langevin diffusion, even on Riemannian manifolds, converges exponentially fast to the target distribution in KL divergence for any target satisfying LSI. To turn this into an algorithm, we bound the "discretization error", i.e., how much the time-discretized algorithm deviates from the continuous process in terms of relative entropy. The discretization proceeds by taking a step defined by the gradient of$F$ at the current point for some positive time, and then performing a Brownian motion for the same length of time.

In the Euclidean setting, the error of discretization can be bounded using smoothness of the target (gradient of f is Lipschitz) and LSI itself [123]. In the Riemannian setting, there is a new source of error, namely the gradient step is along the geodesic with initial velocity determined by the gradient of$F$. The error analysis of [123] was cleanly generalized by [124] with an additional Riemannian discretization error term. They then focused on the manifold given by a product of spheres, where quantities of interest can be written quite explicitly. The general case presents formidable further challenges to carry out without restrictive additional assumptions.

We denote the current step of the algorithm byx_0 and its distribution by ρ_0. ρ_t is the distribution after one discrete step with time parameter$t$.

For a point $x \in M$, let $\Phi_t(x)$ be the exponential map at point$x$ applied to the vector $-\nabla F(x)$, i.e.

$$\Phi_t(x) = \exp_x(-\nabla F(\cdot)).$$

Let $b(t; x(\cdot)) \in T_{\Phi_t(x)}(M)$ be the parallel transport of the vector$-\nabla F(x)$ from x to $\Phi_t(x)$, i.e.

$$b(t; x(\cdot)) = P_{x \to \Phi_t(x)}(-\nabla F(\cdot)).$$

Given x_0, let $\rho_{t|0}$ be the density of the Brownian motion on manifold at time$t$ with respect to $\mathrm{vol}_g dx$, starting from $\Phi_t(x_0)$, and let ρ_{t0} be the joint density ofx_0 and x_t, where$x_t \in B(\Phi_t(x_0), t)$.

To derive the discretization error, we first compute the derivative of the density ρ_t with respect to time in the following lemma, similar to [124] (details of the derivation is in Appendix??).

Lemma 15. For the derivative of the relative entropy,

$$\frac{d}{dt} H(\rho_t) = I(\rho_t) - 2 \mathbb{E}_{\rho_0} \mathbb{E}_{x \sim \rho_{t|0}(\cdot|x_0)} \langle \nabla \log(\rho_t/\pi), -\nabla F(x) \rangle = 2 \mathbb{E}_{\rho_0} \langle -\nabla F(\Phi_t(x_0)), \mathbb{E}_{x \sim \rho_{t|0}} \nabla \log(\rho_t/\pi), b(t; x_0) \rangle.$$

Therefore, we have

$$\partial_t H(t) = I(t) \, DE. \tag{6.7}$$

where the discretization error DE is given by

$$DE_t = E_0 E_{x_{t+\delta}}[\text{grad}_x \log \phi(t, \cdot); \text{grad} F(x)] - E_0[\text{grad}_{t(x_0,\cdot)} E_{t+\delta} \log \phi(t, \cdot); b(t, x_0)] \tag{6.8}$$

In R^n, the parallel transport vector $b(t, x_0)$ above would simply be $\text{grad} F(x_0)$ and could be factored out; here we will have to bound its change along the geodesic.

Next, note that in the above expression, it would be useful to be able to exchange the gradient and the inner expectation in the second term. In R^n, we can write $\nabla E(h(x)) = E(\nabla h(x))$ for a function h, but this is false in general on a manifold, since (i) the gradient is locally defined and (ii) expectation doesn't make sense as $\nabla h(x)$ is a point in the tangent space at x.

6.2.1 Stochastic analysis viewpoint

To address this, we consider the solution flow of the process X_t over the manifold, $H_t : M \to M$, namely $H_t(x_0) = X_t^{x_0}$ is a (random) map from x_0 to the value of the Brownian process started at x_0 at time t (X_t). We set $x_0 = {}_t(x_0, \cdot)$ as the algorithm uses the landing point of the geodesic step from x_0 as the starting point for the Brownian motion. Now recall that

$$dX_t = A(X_t) \circ dB_t + Z(X_t) dt \tag{6.9}$$

where $A(x) = \sqrt{2 g^{-1}(x)}$ and $Z(x) = \frac{1}{2} \nabla \cdot g^{-1}(x)$ are defined by the metric, and $\nabla \cdot$ is the row-wise divergence over the manifold, defined as $(\nabla \cdot g^{-1})_i = \frac{1}{\sqrt{g}} \sum_j \partial_j \sqrt{g} g^{ij} + \sum_{kj} \Gamma^i_{kj} g^{kj}$.

Then we write

$$dE_{x_{t+\delta}}[h(x)] = E_{t+\delta}[dh(x) \, dH_t(x_0, \cdot)],$$

where $dH_t(x)$ is the Jacobian of the random map H, and we are considering $E_{x_{t+\delta}}[h(x)]$ as a function of x_0. In other words the equation above holds in the cotangent space at x_0. To analyze this, we use an extension of a theorem due to [134], which will imply that

$$\text{grad}_{t(x_0,\cdot)} E_{t+\delta}[h(x); b(t, x_0)] = E[\text{grad} h(X_t); V_t], \tag{6.10}$$

for a stochastic process V_t with SDE

$$dV_t = DA(X_t) \circ V_t \, dB_t + DZ_t(V_t) dt. \tag{6.11}$$

with initial condition $V_0 = b(t, x_0)$.

Here V_t will be the action of the Jacobian of the random map H_t on $b(t, x_0)$. We think of V_t as an element in the tangent space $T_{X_t} M$ written in the Euclidean chart. Note that $b(t, x_0)$ is the same as the parallel transport of $\nabla F(x_0)$ from x_0 to $_t(x, \cdot)$.

To show 6.10, recalling $x_0^{\circ e}{}_t(x_0)$, one can rewrite the left hand side as

$$\langle \text{grad}_{x_0^{\circ e}{}_t(x_0)} Eh(X_t), b(t, x_0)\rangle = dEh(X_t)(b(t, x_0)) = dEh(H_t(x_0^{\circ e}{}_t(x_0)))(b(t, x_0)). \quad (6.12)$$

where $dEh(H_t(x_0^{\circ e}{}_t(x_0))) \in T^*_{x_0^{\circ e}{}_t(x_0)}M$ is an element in the cotangent space at x_0, acting upon $b(t, x_0) \in T_{x_0^{\circ e}{}_t(x_0)}M$. We remind the reader that grad is the gradient on the manifold. Now if (1) the random map H was differentiable, (2) the chain rule holds, (3) and one could exchange differentiation and expectation in Equation equation D.46, then

$$dEh(H_t(x_0^{\circ e}{}_t(x_0)))(b(t, x_0)) = Ed(h(H_t(x_0^{\circ e}{}_t(x_0)))(b(t, x_0)) = Edh(dH_t(x_0^{\circ e}{}_t)(b(t, x_0)))(;). \quad (6.13)$$

where $dH_t(x_0^{\circ e}{}_t)$ is the differential (equivalent to Jacobian) of $H_t(x_0^{\circ e}{}_t)$ (this is only a formal derivative) and $dh \in T^*_{H_t(x_0^{\circ e}{}_t)}M$ is the differential (derivative) of h. Denoting $dh_t(x_0^{\circ e}{}_t)(b(t, x_0))$ by V_t, we then obtain Equation equation 6.10. Here, the notion of differentiability that we work with is that the limit defining the derivative of the flow $H_t(\cdot)$ (in the space variable) exists in probability. In order to make the above derivation equation 6.13 feasible, we impose a technical condition on the diffusion 6.9 called strong 1-completeness, which guarantees for every curve $r(\cdot) : [0, 1] \to M$ starting from $r(0) = x_0^{\circ e}{}_0$, that $H_t(r(\cdot))$ is differentiable in r almost surely, and the derivative in some fixed direction V_0 satisfies the SDE in equation 6.11 with initial condition V_0 (recall that the derivative $dh_t(x_0^{\circ e}{}_t)(V_0)$ is a random variable itself). Furthermore, in order to be able to change the order of differentiation and integration as we did in equation 6.13, we need a slightly stronger result in [135], as stated in our Lemma 16 (Appendix G.5).

Lemma 16. [135][Theorem 9.1] Suppose that H is continuously differentiable and we have the strong 1-completeness assumption of the Brownian motion on M, as defined in equation 6.9. If we further assume that M is compact then f and $Y, dh(Y)$ are continuous and bounded, for any vector $V_0 \in T_{x_0}M$:

$$d(E_{t\to\infty} h(X_t))(V_0) = E_{t\to\infty} dh(dH_t(x_0^{\circ e}{}_t)(V_0)),$$

Furthermore, one can relax the compactness assumption under the existence of the second moment $E_{t\to\infty}\|dh(X_t)\|^2$. Here, $\|\cdot\|$ refers to the manifold norm of the cotangent element $dh(X_t) \in T^*_{X_t}M$.

In Section G.9.2, we will verify the strong-1-completeness condition for the log barrier.

Now applying Equation equation 6.10, we can bound the discretization error as

$$E_0 E_{t\to\infty}\langle \text{grad} h(X_t), V_t\rangle \le F(X_t) + B\left(E_t\frac{1}{4}\|\text{grad} h(X_t)\|^2 - E_t\|V_t\|^2 \le F(X_t)\right),$$

using Cauchy-Schwarz and the AM-GM inequality. The first term, with our choice of $h(x) = \log(\pi_t(x)/\pi(x))$ will be a quarter of the relative Fisher information, which is good the continuous process decreases the relative entropy by the relative Fisher information and so we still get a decrease overall. Now the second term needs to be bounded, this takes up most of our effort, and will crucially use the self-concordance of f as well as the smoothness of F. We will do this by showing that the squared norm of V in expectation grows slowly and so does that of F.

Using these results, the discretization error DE can be written as (stated formally in Lemma 164):

$$\begin{aligned} DE &= \mathbb{E}_0 \mathbb{E}_{X_t} \left\langle \operatorname{grad} \log \hat{\rho}_t(\cdot), V_t \right\rangle_{\hat{\rho}_t(X_t)} \\ &= \frac{1}{4} \mathbb{E}_0 \left\| \operatorname{grad} \log \hat{\rho}_t(\cdot) \right\|^2_{\hat{\rho}_t(X_t)} \left\| V_t \right\|^2_{\hat{\rho}_t(X_t)} \\ &\quad + \frac{1}{4} \mathbb{E}_t \left\| \operatorname{grad} \log \hat{\rho}_t(\cdot) \right\|^2_{\hat{\rho}_t(X_t)} - \mathbb{E}_0 \mathbb{E}_{X_t} \left\| V_t \right\|^2_{\hat{\rho}_t(X_t)} \\ &\quad + \frac{1}{4} I_v(\hat{\rho}_t) - \mathbb{E}_0 \mathbb{E}_{X_t} \left\| V_t \right\|^2_{\hat{\rho}_t(X_t)} : \end{aligned} \tag{6.14}$$

In the next section, we bound the second part, i.e. $\mathbb{E}_0 \mathbb{E}_{X_t} \left\| V_t \right\|^2_{\hat{\rho}_t(X_t)}$.

6.3 Bounding the discretization error

Note that $g^{-1/2}$ is the term appearing in the Langevin equation. Since the time derivative of the evolving density w.r.t to the stationary measure $\hat{\rho}_t$ is equal to L (the in nitesimal operator of the Markov chain), and L involves the term $\langle g^{-1/2}, D^2 \hat{\rho}_t \rangle$, one might hope to control this term to bound the discretization error. However, this is not possible for two reasons: rst, $g^{-1/2}$ is not self-concordant in general; in fact, its self-concordance itself is not an a ne-invariant property. Second, the matrix $D^2 \hat{\rho}_t$ is not necessarily positive de nite. Instead of directly trying to control the derivative of the evolving density, we observe that the two terms in the discretization error are similar but di er in the order of computing the gradient and taking the expectation with respect to the conditional distribution $\rho_{t|s}$. In Euclidean space, this change of derivation and integration could be easily handled, but here we need to use a more sophisticated machinery.

The overall approach from here onward is to bound the discretization error $\mathbb{E}_0 \mathbb{E}_{X_t} \left\| V_t \right\|^2_{\hat{\rho}_t(X_t)}$. Notably, we bypass the argument about self-concordance of $g^{-1/2}$ in Lemma 17 by a weaker condition which turns out to be su cient for our needs:

Lemma 17. Suppose the metric g is α_1-self-concordant (in the standard sense). Then, for the square root matrix $A(x) := \sqrt{2} g^{-1}$, we have

$$\begin{aligned} \langle g, D A(v)[v] \rangle &\le 2 \left\| v \right\|_g^2 \left(\frac{2}{3} \alpha_1 \left\| v \right\|_g \right), \\ \langle g^{-1/2}, D A(v) \rangle &\le \sqrt{n} \left(\frac{2}{3} \alpha_1 \left\| v \right\|_g \right). \end{aligned}$$

This bound shows how the self-concordance assumption can be leveraged. In general, the square-root of the metric is not self-concordant [128]. To bound $\mathbb{E}_0 \mathbb{E}_{X_t} \left\| V(X_t) \right\|^2_{\hat{\rho}_t(X_t)}$, we start by bounding the expected norm of $V(t)$ for small enough times t. To this end, we write the squared norm of the V_t process as a stochastic integral along the stochastic curve $X(t)$ which directly follows from Itô's rule (see Appendix G.2).

Lemma 18. Given stochastic vector eld $V(t)$ on the stochastic curve X_t, we have

$$d\langle V, V\rangle_g(t) = 2\langle dV(t), V\rangle_g + \sum_{i,j,p} V_i \partial_p g_{ij} V_j \, dX_p + \sum_{i,j,p,r} \frac{1}{2} V_i V_j \partial_{p,r} g_{ij} \, d[X_p, X_r] + \sum_{i,j,p} \partial_p g_{ij} \, d[V_i, X_p] V_j + \sum_{i,j} \frac{1}{2} g_{ij} \, d[V_i, V_j] \, .$$

Our nal bound on $E_{t\le\tau}\|V_t\|_g^2$ is stated in the following Lemma, proved in section G.3.1.

Lemma 19. For

$$t \le \frac{1}{2n^{3/2}(\alpha_1^{-2} + \alpha_2^{-2} + \alpha_3^{-2})}, \tag{6.15}$$

we have

$$E_{t\le\tau}\|V_t\|_g^2 \le \|V_0\|_g^2 + t n^{3/2} n(\alpha_1^{-2} + \alpha_2^{-2} + \alpha_3^{-2}) \cdot 2\|V_0\|_g^2. \tag{6.16}$$

Next, we control the squared norm of F along the stochastic curve, stated in Lemma 20, proved in section G.3.3. Here, we rst extract useful information from the gradient and Hessian Lipschitzness assumption of f on M using again the self-concordance of the metric (see Section G.3.2).

Lemma 20. For

$$t \le \frac{1}{n(\alpha_1^{-2} + \alpha_3^{-2}) + L_2 n^{3/2}},$$

we have

$$E_{t\le\tau}\|J_t\|_g^2 \le \|J_0\|_g^2 \quad t. \tag{6.17}$$

Finally, we study $\|V(X_t) + F(X_t)\|_g^2$ again using Itô expansion, which is more challenging as the processes V and J have nonzero quadratic variation with respect to each other. We remind the reader that V_t and $F(X_t)$ are elements in the tangent space of X_t and their norms are computed based on the metric of M, so in all the calculations we need to take into account the derivatives of the metric itself. There, we use the norm bounds on V_t and J_t from Lemmas 19 and 20.

Moreover, we also employ a generalization of the Talagrand Wasserstein-KL inequality to Riemannian manifolds by authors in [136]. The nal bound is stated below.

Lemma 21. For arbitrary $A > 0$ under the assumption $t \le B$ and

$$t \le B \wedge \frac{1}{n^{3/2}(\alpha_1^{-2} + \alpha_2^{-2} + \alpha_3^{-2}) + nL_3 + L_2 n^{3/2}\sqrt{n + \log A} + L_2^2},$$

we have

$$E_{t\le\tau}\|V_t - J_t\|_g^2 \le \varepsilon_0 + \varepsilon_H A;$$

with parameters

$$\varepsilon_0 = c_1 e^{c_1 n^{3/2}(\alpha_1^{-2} + \alpha_2^{-2} + \alpha_3^{-2})}; \quad \varepsilon_H = c_2 e^{c_2 n^{3/2}(\alpha_1^{-2} + \alpha_2^{-2} + \alpha_3^{-2})} n L_2^2 \sqrt{n + \log A}\, L_2^2 + n L_3 \, .$$

We combine Lemma 21 with Equations equation 6.14 and equation 6.7 to obtain a di erential inequality for relative entropy. The nal result of this section is presented in Lemma 22, which is used to prove Theorem 19 in Section G.4.

Lemma 22. For time at most

$$B \frac{1}{n^{3-2\hat{}} \; {}^{2}_{1} \quad {}^{2}_{2} \quad {}^{2}_{3} \; nL_3 \quad L_2 n \quad \bar{n}_{1}\bullet \quad L_2^2}^{o};$$

we have fort B :

$$@_t H\hat{}_t\bullet B \frac{3}{4} I\hat{}_t\bullet \quad ! \quad !_0 H\hat{}_0\bullet;$$

Full proofs. The main body of the proofs is in the appendix and is structured as follows: In Section?? we present the calculation of the discretization error. In Section G.3 we bound the expected di erence between V_t and $©F\hat{}X_t\bullet$ which is in turn used to bound the discretization error as we discussed above. Sections G.5 and G.1 are devoted to establishing the prior assumptions regarding the strong completeness condition for our stochastic analysis view point. In Section G.6 we investigate a log-Sobolev inequality for the log barrier geometry inside a polytope. In Section G.7 we review some basic de nitions of manifolds, then we state some key lemmas on the manifold which are used in the analysis. In Section G.8 we start from normal Euclidean smoothness conditions and translate them to ones required by Theorem 19. Finally in Section G.9 we show the required assumptions for the geometry of log barrier inside a polytope, e.g. the second-order self-concordance.

6.4 Discussion

An interesting future direction would be the following algorithmic problem: Given a metric g, an initial point x_0 and a time interval t, sample from the distribution of Brownian motion on the manifold starting at x_0 for time t. The question of sampling according to the natural manifold measure is a special case of this Brownian increment sampling problem for a su ciently large time increment.

We would like to also acknowledge the fact that due this work does not have any potential negative societal impacts.

Chapter 7

Learning Mixtures of Gaussians using Diffusion Models

Abstract

We give a new algorithm for learning mixtures of k Gaussians (with identity covariance in $\mathbb{R}^n$) to TV error ε, with quasi-polynomial ($O(n^{\text{poly}\log\frac{n}{\varepsilon}k})$) time and sample complexity, under a minimum weight assumption. Our results extend to continuous mixtures of Gaussians where the mixing distribution is supported on a union of k balls of constant radius. In particular, this applies to the case of Gaussian convolutions of distributions on low-dimensional manifolds, or more generally sets with small covering number, for which no sub-exponential algorithm was previously known. Unlike previous approaches, most of which are algebraic in nature, our approach is analytic and relies on the framework of diffusion models. Diffusion models are a modern paradigm for generative modeling, which typically rely on learning the score function (gradient log-pdf) along a process transforming a pure noise distribution, in our case a Gaussian, to the data distribution. Despite their dazzling performance in tasks such as image generation, there are few end-to-end theoretical guarantees that they can efficiently learn nontrivial families of distributions; we give some of the first such guarantees. We proceed by deriving higher-order Gaussian noise sensitivity bounds for the score functions for a Gaussian mixture to show that that they can be inductively learned using piecewise polynomial regression (up to poly-logarithmic degree), and combine this with known convergence results for diffusion models.

7.1 Introduction and main results

We address the problem of learning a generalized mixture of Gaussians (with identity covariance) from samples, using the framework of diffusion models. Formally, we wish to learn the following distribution on $\mathbb{R}^n$ from iid samples:

$$P_0 = Q_0 * N(0, I),$$

which we think of as a (possibly continuous) mixture of Gaussians, where Q_0 is the distribution of the means. We assume that the support of Q_0 is contained within k Euclidean balls, each holding a non-trivial amount of mass. A precise definition will be provided shortly in Assumption 1. In the special case that $Q_0 = \sum_{j=1}^{k} \alpha_j$ [illegible], this is exactly a mixture of k Gaussians; however, our results hold for any generalized mixture of Gaussians under Assumption 1. Note that if the covariance is known but non-identity, we can first transform the data to be in this setting. Our goal is distribution learning, to output samples from a distribution ε-close in TV distance to the actual one.

Our motivation for this class is twofold: First, as mixtures of Gaussians are one of the simplest but nevertheless challenging mixture models, this is a classic learning problems in statistics and computer science. As a special case, our result also obtains a completely new method distinct from the common algebraic approaches for learning a discrete mixtures of k Gaussians, e.g., [4] which obtains quasi-polynomial complexity for that setting (see Section 7.1.2 for a discussion of this and other related work). Second, diffusion models are an empirically successful paradigm for generative modeling, which work well for learning multimodal distributions in practice but for which theoretical guarantees are lacking. By applying diffusion models to the problem of learning generalized Gaussian mixtures, our work is the first to give theoretical grounding to the success of diffusion models by fully learning a highly non-trivial class of distributions without assuming oracle access to the score function estimates. This necessitates solving the problem of learning the score function in sub-exponential time. Interestingly, our approach to learning the score across multiple noise levels essential for executing the diffusion process actually leverages the diffusion process itself via maintaining a set of "warm-starts."

7.1.1 Main results

For $x_0 \in \mathbb{R}^n$, let $B(x_0, R) = \{x \in \mathbb{R}^n : \|x - x_0\| \le R\}$ denote the closed ball of radius R around x_0. We make the following assumptions on Q_0.

Assumption 1 (k-locality). Fix $R_0 \ge 1$, D, and k. The following hold:

1. For every point α in the support of Q_0, we have $Q_0(B(\alpha, R_0)) \ge w_{\min}$.
2. There exist $\mu_1, \ldots, \mu_k$ such that the support of Q_0 is contained in $\bigcup_{i=1}^{k} B(\mu_i, R_0)$.
3. $Q_0(B(0, D)) = 1$.

Note that generalized mixture models are a strict generalization of a mixture of k Gaussians: we permit mixtures of k arbitrary distributions supported on balls of radius R_0, convolved with Gaussians. Our main theorem is that these mixtures can be learned with quasi-polynomial time and samples with an algorithm based on diffusion models.

Theorem 20. Given $A > 0$ with $B \ge \min(w_{\min}/2, \frac{\varepsilon}{R_0}, cD, cn, \varepsilon_{\min})$, and given Assumption 1, ?? learns a distribution that is ε-close in TV distance to P_0 with time and sample complexity

$$n \ln(c/\varepsilon) \cdot O\left(\ln(1/\varepsilon)^3 \left(\frac{R_0}{\varepsilon}\right)^6 \cdot \ln(1/\varepsilon)^4\right)$$

with probability $1 - \varepsilon$.

Note that because of the restriction on , $\ln \hat{} c \bullet$ implicitly has logarithmic dependence on D, n, and $c_{\min} C k$. In the case where $\frac{R_0}{0}$ is a constant, we can remove the dependence on $\frac{R_0}{0}$. We further remark that in the case of a (discrete) mixture of k Gaussians, a straightforward SVD pre-processing step can replace the dependence on n to $\min\tilde{} n; k \bullet$, at an extra additive cost polynomial in n (see e.g., [137]).

We note that as a non-parametric family of distributions, it is already highly non-trivial that a k-local generalized mixture of Gaussians has sample complexity that is sub-exponential in dimension. This is an interesting example of a class which does not su er from the curse of dimensionality in either the sample or time complexity.

?? consists of two parts: The learning" part involves learning the score functions of distributions that bridge the data distribution with a pure noise (Gaussian) distribution. Once these scores are obtained, the generation" part uses these learned score functions and can generate as many samples as desired. The high-probability bound is over the learning part: with high probability, the learned score functions are such that the generation process satis es the TV distance bound.

For the learning part, a step size schedule is chosen that would allow generation with speci ed error; the algorithm sequentially learns the score functions from large to small (time or noise level). As the noise level t decreases, a set of warm starts (or cluster centers) C at the current resolution is maintained and updated. Initially (at the highest noise level $t_{N_{\text{step}}}$), all points belong to the same cluster, so $C_{N_{\text{step}}} \tilde{} \bullet$. At each time step, following the recipe for learning di usion models, we cast the score estimation problem for $P_t \quad P_0 \ddagger N \hat{}; \backprime I \bullet$ as a supervised learning problem involving denoising data points. The score function is learned within the family of piecewise low-degree polynomials, whose regions are given by the Voronoi diagram of the cluster centers. This is a polynomial regression problem that can be e ciently solved. Whenever the noise level t is reduced by a constant factor, the set of warm starts C is re ned using ??. ?? uses the current score estimate to denoise the data points and obtain estimates of the means, accurate at the current resolution. These means are clustered to obtain C . Finally, the generation part follows the generation procedure for a di usion model: start with a Gaussian sample, and follow the reverse SDE with the score estimate to obtain a sample from the learned data distribution.

A special case of Theorem 20 is the problem of learning a distribution that is equal to a distribution on a low-dimensional manifold convolved with a Gaussian. The manifold" assumption that we need is much weaker: simply that it has can be covered by N balls of radius R_0, for some constant C, where $l \; A \; 0$ is the parameter that governs the sample complexity. It is straightforward to obtain the following.

Corollary 8. Fix $_0 \quad 1$ and constants $R_0; C \; A \; 1$. Let $0 \; @ \; @ c 2$. Suppose that Q_0 is supported on a set M such that M has radius D, M can be covered with C^{l} balls of radius R_0, and such that for every point in the support of Q_0, $Q_0 \hat{} B_{R_0} \hat{} \bullet \bullet C_{\overline{C^{l}}}$. Then ?? learns a distribution that is -close in TV distance to P_0 with time and sample complexity $\hat{} n \ln \hat{} c \bullet \bullet O \text{‰} \ln \text{‰} \frac{nD}{\text{‰}} \check{z} \check{z}^{7}$ with probability $C 1 \quad$.

We note that this is a setting where di usion models can provably learn under a manifold assumption, but in contrast to most prior work, the distribution cannot be learned using

straightforward methods such as through binning or kernel density estimation. For example, if M can be covered with $\left(\frac{C}{\varepsilon}\right)^d$ balls of radius ε, then learning a distribution exactly supported on M can be done to Wasserstein distance ε with complexity $O\left(\left(\frac{C}{\varepsilon}\right)^d\right)$ simply with a binning procedure. However, we consider learning a distribution on M convolved with a Gaussian, which is a more challenging problem.

Additionally, the complexity of our algorithm (stated in Theorem 20) for the special case of discrete mixtures of k Gaussians is known using a completely different algorithm based on algebraic methods [4] (without the dependence on σ_{min} or D and with better exponents). More precisely, their result learns a distribution ε-close in TV distance to the mixture with time and sample complexity $poly(n^{k^{O(k^2)}}, \frac{1}{\varepsilon^{O(k^2)}})$, and outputs a density function. Though their algorithm does obtain better dependencies, their result relies essentially on a discrete mixture structure, while our result holds for the much larger, non-parametric family of generalized mixtures; to our knowledge the extension to a generalized mixture is novel. (Because their algorithm relies on finding an ε-cover of possible parameters, and an ε-cover of a constant-radius ball is exponential in dimension, it seems unlikely that their methods extend to this setting.) It is interesting to note that the generic framework of diffusion models allows us to match (up to polylogarithmic factors in the exponent) guarantees obtainable using more specialized algebraic procedures.

Note that we do not proceed by learning the density function; instead, "learning" the distribution means that we have a procedure to generate an additional sample, each step of which involves evaluation of a learned score function. It may be possible to upgrade this guarantee to a guarantee of learning the density [138].

We leave open the questions of removing the requirement on σ_{min} and improving the polynomial dependence on $\ln\left(\frac{1}{\varepsilon}\right)$ (in Theorem 20) and d (in Corollary 8). The question remains of whether the complexity of learning mixtures of Gaussians (of equal known covariance) is truly quasi-polynomial, or is actually polynomial. Since our methods work just as well in the more general setting of continuous mixtures, we believe that doing better than quasi-polynomial complexity would require an algorithm specific to a mixture of k Gaussians. Further structure, e.g., hierarchical structure, could also make the problem easier, though our current analysis does not benefit from such assumptions.

We note that mixtures of Gaussians are particularly suited to learning with diffusion models because diffusions preserve the Gaussian. We expect that similar results are possible in other settings with a "match" between the family and the diffusion, e.g., mixtures of product distributions on the hypercube where diffusion is a random walk on the hypercube (i.e. bit-flip noise). It would be interesting to find other families of distributions which be learned using diffusion models, including families of conditional distributions such as mixtures of linear regressions.

Finally, we note that in contrast to our algorithm based on piecewise polynomial regression, in practice the score function is typically learned with a neural network. The score function for a Gaussian mixture is exactly represented by a softmax neural network, which raises the question of whether it can be learned by a neural network with gradient descent. Understanding

neural network training dynamics for di usion models is an important open direction.

7.1.2 Related work

Learning mixtures of Gaussians. The problem of learning a mixture of Gaussians from samples has an illustrious history [139]. We rst distinguish between several types of results. First, one can ask for parameter learning or distribution learning either outputting parameters that are close to the ground-truth parameters, or simply a distribution that is close (e.g., in TV distance) to the ground-truth distribution. For distribution learning, the learning can be improper, that is, the output need not be a mixture of Gaussians. Second, one can either study the problem from an information theoretic perspective the minimum number of samples required to get within a speci ed error regardless of computational cost or computational complexity perspective where the emphasis is on the running time of the algorithm. Below, n denotes the ambient dimension, k the number of components, and the target accuracy.

Most earlier works go through parameter learning and require the means to be su ciently separated. For identity-covariance Gaussians, [137, 140, 141] show that spectral methods work with a separation of at least $\Omega(\min(n, k)^{1/4})$. Allowing arbitrary covariances, [142] show that whenever the mixture is -statistically learnable," it can be learned with running time and sample complexity $\exp(k)\,\mathrm{poly}(n, 1/\varepsilon)$. Several works use the sum-of-squares method [143, 144] to learn a mixture with separation (k) in time $n^{\mathrm{poly}(1/\varepsilon)}$. These methods extend to a broader class of mixture distributions, where components have moments which can be certi ably bounded. [145] obtain a polynomial-time algorithm whenever the separation is $(\ln^{c} k)$ for constant $c > 0$.

Separation conditions are unavoidable for sample-e cient parameter learning. [146] shows that the threshold for e cient parameter learning is $(\sqrt{\ln k})$: with separation $o(\sqrt{\ln k})$, polynomially many samples su ce (information-theoretically), while with separation $(\sqrt{\ln k})$, super-polynomially many samples are required. [147] conducts a more ne-grained study of this problem.

Hence, any sample-e cient algorithm for learning mixtures of Gaussians without separation cannot go through parameter learning. The optimal information theoretic complexity is known up to logarithmic factors: [148] show a sample complexity bound in TV distance of $\Theta(kn^2/\varepsilon^2)$ for a mixture of k Gaussians in $\mathbb{R}^n$ (with any variance) and $\Theta(kn/\varepsilon^2)$ for axis-aligned Gaussians. However, their algorithms are based on brute-force search and have exponential running time. [149] give an improper nearly linear-time algorithm based on polynomial interpolation which learns a mixture of Gaussians with arbitrary variances in 1 dimension, with $O(k/\varepsilon^2)$ samples. Most relevant for us, for a mixture of Gaussians with identity covariance, a breakthrough work by [4] uses algebraic geometry to obtains a time and sample complexity of $(k/\varepsilon)^{O(\ln^2 k)}$ plus polynomial factors.

In the statistics literature, the model is referred to as the Gaussian location mixture. [150, 151] consider arbitrary mixing measures Q_0 and give nite-sample bounds using non-parametric MLE (maximum likelihood estimation) for squared Hellinger risk, which scale as $\frac{\log N}{N}$ in

terms of the number of samples N. The risk depends on the volume of an approximate support of Q_0, but also include a constant with unspecified dependence on the dimension n.

We note that the picture is more complicated when variances are unknown: [152] obtain statistical query (SQ) lower bounds ($2^{n^{\Omega(1)}}$ queries of fixed polynomial precision) based on a connection with non-Gaussian component analysis, with a "parallel pancake" construction in a unknown direction. [153] show that $\log n$ components are enough to obtain a super-polynomial lower bound assuming exponential hardness of the classical LWE problem.

Finally, a recent line of work considers the problem of robust learning of Gaussian mixtures [154, 155], i.e., under adversarial corruption of some fraction of samples; these methods have complexity $n^{O(k)}$.

Concurrent work. During the preparation of this manuscript, we were made aware of independent and concurrent work by Chen, Kontonis, and Shah [156] which also gave guarantees for learning Gaussian mixtures using diffusion models and piecewise polynomial regression for score estimation. They consider the more general case where covariances are well-conditioned but arbitrary, but their runtime scales exponentially in $\mathrm{poly}(k)$ rather than $\mathrm{poly}\log(k)$, which is unavoidable by SQ lower bounds [152]. In contrast to our work, their work uses a different approach to polynomial approximation and does one-shot learning of parameters via spectral methods.

Diffusion models. Diffusion models [157–159] are a modern generative modeling paradigm which involves defining a forward noising process which turns a data distribution into a pure noise (e.g., Gaussian) distribution, and then learning to simulate the reverse process. For diffusion models based on SDEs (stochastic differential equations), the reverse process involves the score function (gradient of log-pdf) of the intermediate distributions; hence they are also called score-based generative models (SGM). See [160] for a technical tutorial. We note that diffusion models are essentially a reparameterization of stochastic localization [161] as pointed out by [162]. Stochastic localization has been independently studied in the probability literature and been used to obtain new results in sampling [163, 164].

Theoretical work has focused on two problems: (1) When is it possible to efficiently learn the score? (2) Given a learned (L^2-accurate) score function, what guarantees can we obtain for sampling from the data distribution? Answers to these two questions together would give an end-to-end result for learning via diffusion models.

Addressing (2), it is a remarkable fact that having an L^2-accurate score function for the sequence of distributions is sufficient for sampling under minimal distributional assumptions, allowing even multimodal distributions which cause slow mixing for local MCMC algorithms [165–167]. [168] show that it suffices to have a number of steps linear in the dimension.

Question (1) has proved to be thornier; it has been a challenge to obtain end-to-end results for non-trivial settings. Several works consider the problem of representability by neural networks, such as [169] for distributions whose log-density relative to a Gaussian can be represented by a low-complexity neural network, or [170] for graphical models. Following the work on neural network function approximation for smooth functions, [171] give nearly

minimax optimal estimation rates for densities in Besov spaces. [172] relates score learning to kernel density estimation. The manifold assumption is another popular setting for analysis: [173] considers generalization error, and [166] give learning guarantees when the distribution is supported on a subspace (however, this family of distributions can be trivially learned by first recovering the subspace). Finally, we note that score matching can be used to learn exponential families for which sampling is difficult [174, 175]; however, these methods only use the score for the data distribution, rather than a sequence of distributions as in a diffusion model.

[176] consider using diffusion models to learn Gaussian mixtures, and show that diffusion models can do as well as the EM algorithm. However, they either require $k = 2$, or the components to be well-separated and $O(1)$-warm starts to be given for all the means. Gaussian mixtures are a popular toy model for understanding various aspects or behavior of diffusion models, including learning behavior [177], sample complexity [178], guided diffusion [179], and critical windows [180].

Analytic conditions for learning functions. We rely on the noise sensitivity/stability framework [181], which shows that Gaussian noise stability (or small Gaussian surface area) implies approximability by a low-degree polynomial, giving an efficient "low-degree algorithm" for learning. We note a similar-in-spirit result that under the Gaussian distribution, intersections of k halfspaces can be learned in time $n^{O(\ln k)}$. The noise sensitivity framework was previously developed for boolean functions on the hypercube [182] and applied to learning function classes such as functions of halfspaces [183, 184].

Learnability by neural networks can also be related to complex analytic properties of the function: [185] relate a radius of analyticity to the Hermite expansion which gives results on representability by neural networks. Learnability by neural networks for multi-index models [186] functions depending on the projection of the input to a few dimensions (such as the score function of a Gaussian mixture with $k \ll n$) is also related to the Hermite expansion of the function.

7.1.3 Notation

We let γ_{μ,σ^2} denote the density of $\mathcal{N}(\mu, \sigma^2 I_n)$, and abbreviate $\gamma_{\sigma^2} = \gamma_{0,\sigma^2}$. We abbreviate this as γ when σ^2 is understood. In general, we denote probability measures by uppercase letters and their corresponding densities by lowercase letters, though sometimes we will conflate the two.

We use $\|v\| = \|v\|_2$ to denote the norm of a vector $v \in \mathbb{R}^n$, to avoid confusion with function norms. For a measure μ on $\mathbb{R}^n$, let $\|f\|_{L^p(\mu)} = \left(\int_{\mathbb{R}^n} |f|^p \, d\mu\right)^{1/p}$. When the measure is clear, we may simply write $\|f\|_p$. Let $\|f\| = \|f\|_{L^2(\mu)}$. For a $\mathbb{R}^n$-valued function, we write $\|f\|_{L^p(\mu)}$ to mean $\|\|f\|\|_{L^p(\mu)}$. For $x_0 \in \mathbb{R}^n$, let $B_R(x_0) = \{x \in \mathbb{R}^n : \|x - x_0\| \le R\}$, and let $B_R = B_R(0)$. Let $\binom{S}{k}$ denote the set of subsets of S of size k, and $\binom{S}{\le k}$ denote the set of subsets of S of size at most k. Let $h_{k;1}$ be the Hermite polynomial of degree k, which satisfies the following

recursive equation for $z \in \mathbb{R}$,

$$h_{k;1}(z) = z h_{k-1;1}(z) - h''_{k-1;1}(z); \quad h_{0;1}(z) = 1$$

and define the rescaled version (with variance σ^2) to be $h_{k;\sigma^2}(z) = h_{k;1}(z/\sigma)$. We use the notation k to denote a multi-index in $\mathbb{N}_0^n$. The multivariate Hermite polynomials, indexed by $k \in \mathbb{N}_0^n$ is defined as $h_{k;\sigma^2}(z) = \prod_{i=1}^n h_{k_i;\sigma^2}(z_i)$, where $z \in \mathbb{R}^n$. It is well-known that $\{h_{k;\sigma^2}\}_{k \in \mathbb{N}_0^n}$ forms an orthogonal basis for $\mathbb{R}^n$ with respect to γ_{σ^2}.

We give some further background and notation on Markov semigroups and generators in Section 7.4.

7.2 Proof overview

Diffusion models give a way to reduce the problem of learning a probability distribution P_0 from samples, to the problem of estimating *score functions* of $P_0 * \mathcal{N}(0, tI)$ for a sequence of noise levels t. A sequence of works [165, 168] shows that this reduction works for general distributions, with explicit error bounds.

Our proof then proceeds by sequentially learning the score functions for a decreasing sequence of t's. The proof hinges on the fact that the score function for a mixture of Gaussians is approximable by a piecewise low-degree polynomial function. We first give some intuition as to why we expect this to be true. Consider a mixture of two Gaussians $P = \frac{1}{2}\mathcal{N}(\mu, \sigma^2) + \frac{1}{2}\mathcal{N}(-\mu, \sigma^2)$. The score function for this distribution is

$$\nabla \ln\left(e^{-\frac{(x-\mu)^2}{2\sigma^2}} + e^{-\frac{(x+\mu)^2}{2\sigma^2}}\right) = x - \mu \tanh\left(\frac{x\mu}{\sigma^2}\right).$$

Consider two cases:

1. When μ is bounded, the two Gaussians have non-negligible overlap. Using smoothness properties of $\tanh$, we can approximate the score function uniformly with a low-degree polynomial on $[-C, C]$.
2. When $\mu \gg \sigma$, then we can no longer uniformly approximate the score function with a low-degree polynomial on $[-C, C]$, because $\tanh(x\mu/\sigma^2)$ is very steep around 0 and close to flat for an interval whose length approaches ± 1. However, importantly, we don't need to: we only care about error with respect to P, which is mostly supported on $[-C, C] \cup [\mu - C, \mu + C]$. On each of these intervals, the posterior of the mean is close to flat because the other mixture component has negligible effect and so well-approximated by a polynomial.

In general, we can hope that we can cluster the Gaussians, show that the score function for each cluster can be approximated by a polynomial, and each cluster has negligible effect on the other clusters, giving the piecewise polynomial structure.

Before giving a detailed proof sketch, we we highlight some key techniques used in our proof which may be more generally useful.

- Higher-order noise sensitivity: We first consider the case of a single cluster, that is, all the means in the mixture are within a small ball. Let f be the function we wish to estimate. We use the technique of Gaussian noise sensitivity by bounding the norm of Lf with respect to a Gaussian γ, where L is a differential operator, namely the generator of the Ornstein-Uhlenbeck (OU) process, we can show that f is approximable by a low-degree polynomial with respect to γ. However, this only allows approximation by a polynomial of degree $O(1/\epsilon)$, leading to a sample and time complexity of $n^{O(1/\epsilon)}$. To overcome this, we instead prove higher-order noise sensitivity bounds, bounding $\|L^m f\|$ for m logarithmically large. To our knowledge, this is the first time that higher-order noise sensitivity has been considered. The high level of smoothness required to bound $\|L^m f\|$ derives from the fact that the score function has a nice representation in terms of a posterior expectation; we bound its derivatives with careful bookkeeping.
- Higher-order smoothing: A particularly delicate part of our proof is a change-of-measure argument. Gaussian noise sensitivity gives error bounds under the Gaussian measure γ; however, we care about the error under the true data distribution ρ_0, which is a mixture. If f is the true function (related to the score function) and g is the estimate, we have by the Cauchy-Schwarz inequality that $\mathbb{E}_{\rho_0}(f-g)^2 \le \mathbb{E}_{\gamma}(f-g)^2 \cdot \mathbb{E}_{\gamma}(f-g)^2 \cdot \left\|\frac{d\rho_0}{d\gamma}\right\|_{L^\infty}$. However, this means that we need $\|f-g\|_{L^\infty}$ to be small, while we only have control over $\|f-g\|_{L^2(\gamma)}$. A standard way that we can bound a higher L^p norm by a lower one is to use hypercontractivity: smooth f by the OU semigroup to obtain $P_t f$. Again, this "first-order" smoothing turns out to be insufficient, and we introduce a higher-order smoothing obtained from a higher-order finite-differencing.
- Using the diffusion model for maintaining clusters: In the case of a single cluster, the above argument shows the existence of a logarithmic-degree polynomial approximation of the score function, which can be efficiently learned using polynomial regression. In the general case, by localizing the effect on the score function from different clusters, there exists a piecewise polynomial approximation. The pieces can be taken to be the Voronoi cells of a suitable set of cluster centers, and the score function can be efficiently learned if these cluster centers were known. The key observation is that having an estimate of the score function at the previous (higher) noise level allows this approximate clustering: by Tweedie's formula, the score function exactly points in the direction of denoising a data point, i.e., the posterior of the Gaussian mean that the point came from! Thus, the denoising principle of the diffusion model is an integral part of inductively allowing us to maintain the clusters as the noise level t decreases. As t decreases, we obtain more accurate estimates of the means, which allows us to refine the clusters to the current resolution.

The first two techniques are already present in the single-cluster setting, while the last one is a key part of extending the argument to the multiple-cluster setting.

7.2.1 Learning with diffusion models

The idea behind diffusion models is to first define a forward process based on a SDE that takes the data distribution to a pure noise distribution, in our case a Gaussian. Then using

a result on reversing a SDE [187], we can write down the reverse process which involves the score function. We apply this in the case when the forward process is simply Brownian motion,

$$dx_t = dW_t; \quad x_0 \sim P_0$$

to obtain that this process on $[0, T]$ is equivalently described by

$$dx_t = -\nabla \ln p_{T-t}(x_t)\,dt + d\bar{W}_t; \quad 0 \le t \le T; \quad x_T \sim P_T$$

where x_t has distribution p_t and $\bar{W}_t$ is reverse Brownian motion. We choose T large enough so that P_T is close to Gaussian. Hence, if we learn the score function $-\nabla \ln p_t(x_t)$, then we can approximately simulate the reverse process. In Section 7.3, we make this precise by adapting known convergence results on diffusion models, which show that we can approximately sample from the data distribution given a L^2-accurate score function.

By Tweedie's formula, the score function admits an interpretation in terms of the posterior mean given an observation (see equation 7.4), so the score matching objective $\mathbb{E}_{p_t} \|s + \nabla \ln p_t\|^2$ can be rewritten as the *supervised* denoising auto-encoder objective (see Section 7.3.2). The problem is now reduced to that of learning the score function $-\nabla \ln p_t$ for each time t. For this, we will show that it resides in a low-dimensional function class we can optimize over.

7.2.2 Learning the score for a single cluster

In Section 7.4, we show that in the special case where all centers are close together, i.e. Q_0 is supported on a small Euclidean ball, there is a polynomial of low ($\mathrm{poly}\log k$) degree that approximates the score function with respect to the mixture distribution. The key enabling result is Lemma 23, which shows that Gaussian noise stability implies low-degree polynomial approximability, by considering the expansion in Hermite polynomials (eigenfunctions of L). In contrast to existing literature, we employ a *higher-order* version of noise stability which involves bounding the L^2-norm of $L^m f$ iterates of the generator L of the Ornstein-Uhlenbeck process. Specifically, any function f can be approximated by a polynomial g of degree d such that

$$\|f - g\|_{L^2(N(0,I))} \le \frac{\|L^m f\|_{L^2(N(0,I))}}{d^m}.$$

As long as we can bound $\|L^m f\|_{L^2(N(0,I))}$ by $\|L^m f\|$ for reasonable L, then the dependence of the necessary degree d on the desired accuracy is $O(\mathrm{poly}\log(1/\varepsilon))$ rather than $O(1/\varepsilon)$.

The interpretation of the score function as a posterior mean gives us a handle on computing (Lemma 24 in Section 7.4.1) and bounding (Lemma 25 in Section 7.4.2) its derivatives. For convenience, we consider a function f which is linearly related to the score function, and bound $\|L^{2m} f\|$. Specifically, $f(y) = \langle x, e_y \rangle$, where $\langle \cdot, e_y \rangle = \mathbb{E}_{P_{x|y}}$ is an expectation of the posterior distribution of the mean of the Gaussian that y originated from. Differentiating this creates "replicas," giving expectations of higher moments under the posterior. Arguments using Jensen's and Hölder's inequalities then bounds this using Gaussian moments.

However, one key problem remains: Lemma 25 gives us low-degree approximability with respect to the Gaussian γ, not the actual mixture P_t. A standard change-of-measure

argument can bound the L^2 error under ε by a higher L^p error under ε: by the Cauchy-Schwarz inequality,

$$\|f - g\|_{\varepsilon}^2 \le \|f - g\|^2 \, \|f - g\|_{L^{4\alpha}} \cdot \varepsilon^{2\alpha} \varepsilon^{1-2\alpha}.$$

Here, f is the actual function and g is the polynomial approximation. However, we are not able to directly obtain a higher L^p polynomial approximation of f. The problem is that the degree of the polynomial required depends poly-logarithmically on the desired L^2 accuracy ε, while passing to a higher L^p norm results in a multiplicative factor exponential in the degree, resulting in an error $e^{\ln C \alpha \cdot 1-\alpha} \gg 1$.

By hypercontractivity, we know that we can bound higher L^p norms of $P_t f$ the smoothing of f by the Ornstein-Uhlenbeck operator for time t by $\|f\|_{L^2}$. We can approximate $f - P_t f$ and then approximate $P_t f$ by the polynomial g. In order for $\|f - P_t f\|_{\varepsilon} \le \varepsilon$, we need $t = O(\varepsilon)$. However, this turns out to be not enough smoothing. We would like to write f approximately as a reasonably-sized linear combination of $P_t f$ (for different t's), for t's that are as large as possible. The insight is that $P_t f - f = -t \, \mathrm{d}/\mathrm{d}t \, P_t f + \cdots$; higher-order finite differencing gives

$$\left(P_t - \mathrm{id}\right)^m f = t^m \frac{\mathrm{d}^m}{\mathrm{d}t^m} P_t f + \cdots.$$

Hence, by taking $m = \ln(c)$, we can take t much larger independent of ε and still obtain an ε-approximation. This is carried out in Lemma 26 in Section 7.4.3. In summary, we approximate f by a higher-order smoothing by the Ornstein-Uhlenbeck process before the polynomial approximation.

7.2.3 Learning the score for multiple clusters

In Section 7.5 we consider the general setting of multiple clusters. Recall that as we would like to run the backwards diffusion process from pure Gaussian noise back to our mixture, we want to learn the score function $\nabla \ln p_t(x_t)$ at various times $t \le t'$. It is no longer true that a single polynomial of low degree approximates the score function with respect to the mixture distribution. We must deal with two challenges:

1. The score function now has a more complex structure, as it is the posterior mean with respect to multiple clusters of Q_0.
2. We care about the error with respect to a mixture measure which is concentrated around k clusters.

How can we overcome these issues?

First assume we are given a set of warm starts $\{\mu_i\}_{i=1}^{k'}$ to the mixture measure $P_t Q_0 * N(0, \sigma^2 I)$, in the sense that balls of radius roughly $O(\sigma + \bar{t})$ around these warm starts cover the support of Q_0 (Definition 21). Then, we hope to approximate the score function by a suitable piecewise polynomial function with respect to the Voronoi diagram of the warm starts, such that, in every Voronoi cell V_i, the corresponding polynomial approximates well the score function restricted to that cell:

$$g(x) = \sum_{i=1}^{k'} 1_{V_i}(x) \sum_{|\alpha| \le d} a_{i,\alpha} h_\alpha(x - \mu_i).$$

Here, h_k is the Hermite polynomial with multi-index k. To reduce the problem of approximating the score in a Voronoi cell in the multiple cluster case to the single cluster case, we prove that there is a neighborhood S_i of the Voronoi cell V_i, such that, viewing the score function as the posterior mean, we can approximate it in V_i by pretending that the prior distribution has been restricted to the mixing measure Q_0 restricted to S_i (Lemma 27 in Section 7.5.1). Given that the radius of this neighborhood is small, we can use this argument to overcome challenge 1. For challenge 2, we bound the Rényi divergence of the mixture measure with respect to the Gaussian centered at the warm start point in the Voronoi cells (Lemma 29 in Section 7.5.2), and use that for a change-of-measure argument given by Lemma 26, to bound the approximation error under the mixture measure in terms of the approximation error under the Gaussian. Having shown the existence of the piecewise polynomial approximation by combining the Hermite approximations within all the Voronoi cells (Lemma 30 in Section 7.5.2), we can then e ciently approximate the score by applying a polynomial regression with suitable choice of degree restricted to each Voronoi cell. We analyze the sample complexity of piecewise polynomial regression in Section 7.5.3 by bounding the generalization gap of the squared loss of low-degree polynomial functions in predicting the score (Lemma 32).

The remaining problem is to obtain a set of warm starts. It turns out that we can exploit the score function learned in the previous iteration of the algorithm (for a larger t) to generate these warm-start points (Lemma 37,?? in Section 7.5.4); the idea is that the score function exactly points toward the denoising direction, i.e. approximately towards points sampled form the mixing measure Q_0. Therefore, adding the score function estimate to su ciently many noisy samples y_i is su cient for obtaining a suitable set of warm starts:

$$\hat{A}_i \quad y_i \quad {}^2\hat{s}(y_i, \bullet):$$

The $\hat{A}_i$ can be greedily clustered and subselected to keep the set of warm starts small. Based on this observation, we propose an inductive recipe to iteratively estimate the score functions for di erent noise levels, while maintaining a warm-start set.

Contents

7.3 Diffusion models

Using the framework of diffusion models, the algorithm and analysis consist of two main parts: showing that we can learn this score function efficiently, and showing that an accurate score function allows generating more samples from the distribution. In this section, we focus on the second part, which has been well-studied.

7.3.1 Convergence guarantees for diffusion models

Two popular parameterizations of diffusion models are the variance-exploding (VE) and variance-preserving (VP) processes. The diffusion models are also known as score matching with Langevin dynamics (SMLD) and denoising diffusion probabilistic models (DDPM), respectively. The forward SDE for the VE process is simply Brownian motion, while the forward SDE for the VP process is the Ornstein-Uhlenbeck process. Below, we lay out the forward SDE for each of these process, and the backward SDE obtained by [187]:

VE:	VP:
$dy_t = dW_t$	$dx_t = -\frac{1}{2} x_t\,dt + dW_t$
$dy_t = -\nabla \ln p_t(y_t)\,dt + d\bar{W}_t$	$dx_t = -\frac{1}{2} x_t - \nabla \ln p_t(x_t)\,dt + d\bar{W}_t$
$dy_t = -\nabla \ln p_{T-t}(y_t)\,dt + dW_t'$	$dx_t = -x_t - 2\nabla \ln p_{T-t}(x_t)\,dt + \sqrt{2}\,dW_t'$

where p_t is the density of the either x_t or y_t, $\bar{W}_t$ is reverse Brownian motion, and W_t, W_t' are usual Browian motions, and the reverse processes are initialized at $y_0 = p_T$, $x_0 = p_T$ and we can match up trajectories $y_t = y_{T-t}$, $x_t = x_{T-t}$ for $t \in [0, T]$. For convenience, we work with the VE process. As most convergence guarantees in the literature are for the VP process, we need to adapt those results. Note that as continuous processes we have $x_t = e^{-t} y_{e^{2t}-1}$, but some care is required when matching up the discretizations.

The following is an adaptation of [168, Theorem 1], after reparameterizing the VP into the VE process.

Theorem 21 (Reverse KL guarantee for variance-exploding diffusion models) Let $0 = t_1 < t_2 < \cdots < t_{N_{\text{step}}} < T$ and $\gamma_k = t_{k+1} - t_k$. Suppose $T \geq 1$. Assume the following.

1. We have a score function estimate $s_{t_k}(y)$ for every t_k such that
$$\mathbb{E}_{P_{t_k}} \|\nabla \ln p_{t_k}(y) - s_{t_k}(y)\|^2 \leq \varepsilon_k^2.$$
2. The data distribution has bounded second moment $m_2 = \mathbb{E}_{P_0} \|y\|^2$.
3. For some $\alpha > 1$, the step size schedule satisfies
$$\gamma_k = 1 \wedge (t_{k+1} - t_k) \leq \max\left\{\alpha \varepsilon_k^2, (t_{k+1} - t_k)\right\}.$$

Let $\hat{p}_t$ denote the distribution of the following discretization of the reverse process when

initialized at $\hat{\mu}_T = N(\hat{\mu}_T, \hat{\sigma}_T^2 \cdot I)$.[1]

$$\hat{\mu}_{t_k} = \hat{\mu}_{t_{k-1}} + 2(\hat{t}_{k-1} + 1)\cdot(\hat{t}_k - \hat{t}_{k-1}) \cdot s_{t_{k-1}}(\hat{\mu}_{t_{k-1}}) + \sqrt{(\hat{t}_k - \hat{t}_{k-1})} \cdot z_k.$$

Then

$$D_{KL}(p_{t_1} \| \hat{p}_{t_1}) \le \frac{n + M_2}{T + 1} \sum_{k=1}^{N_{step}-1} \ln\left(\frac{t_{k+1} + 1}{t_k + 1}\right) \cdot (t_k + 1) + \varepsilon_1^2 + n \ln(T + 1) \cdot \varepsilon_2^2 n N_{step} + M_2 \varepsilon_1^2. \tag{7.1}$$

Moreover, we can choose a schedule of length $N_{step} = O(\ln(T/t_1))$ to make assumption 3 hold. If we choose $\varepsilon_1^2 = \frac{M_2 n}{\varepsilon_2^2}$ and $\varepsilon_2^2 = \frac{\varepsilon^2}{M_2 n \ln(T+1)}$, and we have $\varepsilon_k^2 = \frac{2(t_k + 1)}{\ln(T+1)} \varepsilon^2$ for each k, then the error is $O(\varepsilon^2)$.

The proof is deferred to Appendix H.1.

7.3.2 Score function computation for Gaussian mixture

To learn a distribution using a standard diffusion model, we need to learn its score function (gradient of the log of the pdf) under Gaussian convolution, with varying levels of noise. For Gaussian mixtures, the score function has a particularly nice form, which we now derive.

First, we let $P_t = P_0 * N(0, \sigma_t^2 \cdot I)$ and $Q_t = Q_0 * N(0, \sigma_t^2 \cdot I)$. Then $P_t = Q_0 * N(0, (\sigma_t^2 - \sigma_0^2) \cdot I) = Q_{\sigma^2}$ where $\sigma^2 = \sigma_t^2 - \sigma_0^2$. We consider the following probabilistic model where μ_1, μ_2 are drawn independently:

$$Q_0, \mu_1, \mu_2 \to N(\mu_1, \sigma^2 \cdot I), \quad X = \mu_1 + \varepsilon, \quad Y = \mu_2 + \varepsilon. \tag{7.2}$$

Then $X \sim P_0$ and $Y \sim P_t$. Letting p_t, q_t be the corresponding densities and $V_t = -\ln p_t$, we have by Tweedie's formula [188] (see Appendix H.5 for a derivation) that

$$\nabla V_t(y) = -\sigma^2 E[\nabla \ln q_\sigma(Y - y) \mid Y = y] = -\sigma^2 \nabla \ln \int_{\mathbb{R}^n} \exp\left(-\frac{\|y - \mu_1\|^2}{2\sigma^2}\right) dQ_0(\mu_1). \tag{7.3}$$

$$f_{\sigma^2}(y) = e^{-V_t(y)} = E[q_\sigma(Y - y) \mid Y = y] = \frac{\int_{\mathbb{R}^n} \exp\left(-\frac{\|y - \mu_1\|^2}{2\sigma^2}\right) dQ_0(\mu_1)}{\int_{\mathbb{R}^n} \exp\left(-\frac{\|y - \mu_1\|^2}{2\sigma^2}\right) dQ_0(\mu_1)}. \tag{7.4}$$

Note that in the same way we have

$$\nabla V_t(y) = -t \nabla \ln E[X \mid Y = y] = -t \nabla \ln \int_{\mathbb{R}^n} x \exp\left(-\frac{\|y - x\|^2}{t}\right) dP_0(x). \tag{7.5}$$

$$y - t \nabla V_t(y) = E[X \mid Y = y] = \frac{\int_{\mathbb{R}^n} x \exp\left(-\frac{\|y - x\|^2}{t}\right) dP_0(x)}{\int_{\mathbb{R}^n} \exp\left(-\frac{\|y - x\|^2}{t}\right) dP_0(x)}. \tag{7.6}$$

[1] It would be more natural to have h_k as the coefficient in front of the score term. This coefficient comes from doing a change-of-variable from Appendix H.1 for the DDPM process, rather than reproving the theorem for this process.

Because of this identity, the score function can be learned as the minimal mean squared estimator in a (supervised) denoising problem of estimating X given Y, known as the denoising auto-encoder (DAE) objective [189]. equation 7.3 equation 7.4 will be useful for analysis while equation 7.5 equation 7.6 is used for the actual learning algorithm. Because of the nice form of equation 7.4, we will actually aim to learn f_{σ^2}. We remark that in the case of a finite mixture, equation 7.4 shows that f_{σ^2} is represented by a softmax neural network with 1 hidden layer of k units.

7.4 Learning the score for a single cluster

We will follow the approach in [181], though with functions that are $\mathbb{R}^n$-valued rather than $\{0,1\}$-valued.

First, we give some background on the Ornstein-Uhlenbeck process. Let γ_{σ^2} denote the density of $N(\mu, I)$. The generator L L_{σ^2} of the scaled Ornstein-Uhlenbeck process with variance σ^2, also known as Langevin dynamics for γ, is

$$Lf(x) = \sigma^2\langle x, \nabla f(x)\rangle + \sigma^2 \Delta f(x). \tag{7.7}$$

For a $\mathbb{R}^d$-valued function f, we interpret this componentwise, i.e.,

$$Lf(x) = \sigma^2 Df(x)x - \sigma^2 \sum_{i=1}^{d} \partial_{ii} f.$$

The eigenfunctions are the (suitably scaled) Hermite polynomials $\{h_k\}_{k\in\mathbb{N}_0^n}$, with $h_k = h_{k,\sigma^2}$, as defined in Section 7.1.3, having eigenvalues $-\frac{|k|\sigma^2}{2}$. Here we used $|k|$ to denote $|k_1| + \cdots + |k_n|$. The Hermite polynomials form a complete orthogonal basis for $L^2(\gamma)$. The scaled Ornstein-Uhlenbeck process can be described by the SDE

$$dx_t = -\sigma^2 x_t\,dt + \sqrt{2}\sigma\,dW_t. \tag{7.8}$$

Let $(P_t)_{t\geq 0}$ be the Markov semigroup of the scaled Ornstein-Uhlenbeck process, with generator given by equation 7.7. That is, we have

$$P_t f(x) = E_{x_0 = x} f(x_t) \text{ when } x_t \text{ solves equation 7.8}, \tag{7.9}$$
$$Lf = \lim_{t\to 0} \frac{P_t f - f}{t}.$$

The following encapsulates the technique of noise sensitivity/stability: a bound on the $L^2(\gamma)$ norm of $L^m f$ implies approximability of f by a low-degree polynomial. To our knowledge, only the case $m = 1$ has been considered in the literature; however, similarly to how bounds on higher moments imply better tail bounds, bounding $\|L^m f\|$ for larger m can give better approximation.

Lemma 23 (Noise stability implies low-degree approximability) Let γ denote the density of $N(\mu, I)$. Suppose that $f : \mathbb{R}^d \to \mathbb{R}^n$ satisfies $\|f\|_{L^2(\gamma)} \leq a$, and that $m > N, L$ are such that $\|L^m f\|_{L^2(\gamma)} \leq BL^m$. For $d > N$, there exists a polynomial g of degree $\leq d$ such that

$$\|f - g\|_{L^2(\gamma)} \leq B\left(\frac{L^2}{d}\right)^m.$$

Proof. Expand f in the eigenfunction basis of L^2 as $f = \sum_{k>N_0^d} a_k h_k$ where $a_k \in \mathbb{R}^d$. Then

$$L^m f = \sum_{k>N_0^n} \frac{\lambda_k^m}{2^m} a_k h_k .$$

Hence

$$\frac{d^{2m}}{4^m} \sum_{\lambda_k \le d} \|a_k\|^2 \le \sum_{k>N_0^n} \frac{\lambda_k^{2m}}{4^m} \|a_k\|^2 \le L^{2m} \cdot \sum_{\lambda_k \le d} \|a_k\|^2 \le \left(\frac{L^2}{d}\right)^{2m} .$$

Taking $g = \sum_{\lambda_k \le d} a_k h_k$, we have that $\|f - g\|_{L^2} \le \sum_{\lambda_k \le d} \|a_k\|^2$, which gives the desired bound. □

7.4.1 Calculation of $L^m f$

Let μ be a measure on $\mathbb{R}^n$ with all moments finite. Consider generating $x_1, \ldots, x_n$ where $x \sim N(\theta, I)$. Let $\ell = \ell_y = E_{P(\theta|y)}$ where $P(\theta|y)$ is the posterior distribution given by

$$\frac{dP(\theta|y)}{d\mu}(\theta) = \exp\left(\langle \theta, x \rangle - \frac{\|\theta\|^2}{2}\right) .$$

(This is not to be confused with the inner product.) Then letting $f = f_2$ and Q_0 as in equation 7.4, we have $f(y) = f_2(y) = \ell_y(x)$.

We first derive some formulas for differentiating posterior expectations ℓ. Let $\xi = x - \ell_y$ denote the centered random variable. For a function $g : \mathbb{R}^n \to \mathbb{R}$, we have the following general formula for differentiation with respect to y_i:

$$\begin{aligned} \frac{\partial}{\partial y_i} \ell_y(g(x)) &= \frac{\int_{\mathbb{R}^n} g(x) x_i \exp\left(\langle \theta, x \rangle - \frac{\|\theta\|^2}{2}\right) d\mu(\theta)}{\int_{\mathbb{R}^n} \exp\left(\langle \theta, x \rangle - \frac{\|\theta\|^2}{2}\right) d\mu(\theta)} \\ &\quad - \frac{\int_{\mathbb{R}^n} g(x) \exp\left(\langle \theta, x \rangle - \frac{\|\theta\|^2}{2}\right) d\mu(\theta) \int_{\mathbb{R}^n} x_i \exp\left(\langle \theta, x \rangle - \frac{\|\theta\|^2}{2}\right) d\mu(\theta)}{\left(\int_{\mathbb{R}^n} \exp\left(\langle \theta, x \rangle - \frac{\|\theta\|^2}{2}\right) d\mu(\theta)\right)^2} \\ &= \ell_y(g(x) \xi_i) \\ \nabla \ell_y(g(x)) &= \ell_y(g(x) \xi) . \end{aligned}$$

More generally, let $\ell_y(g(x^{(1)}, \ldots, x^{(r)}))$ denote $E g(x^{(1)}, \ldots, x^{(r)})$ where $x^{(1)}, \ldots, x^{(r)}$ are independent draws from the posterior $p(\theta|y)$. We use the trick of replacing the means with independent copies of the random variable (see e.g., [190]) to obtain

$$\begin{aligned} \nabla \ell_y(g(x^{(1)}, \ldots, x^{(r)})) &= \ell_y(g(x^{(1)}, \ldots, x^{(r)}) \cdot (\xi^{(1)} + \cdots + \xi^{(r)})) \\ &= \ell_y(g(x^{(1)}, \ldots, x^{(r)}) \cdot (x^{(1)} + \cdots + x^{(r-1)})) . \end{aligned}$$

and for a function $h : R^r \to R$,

$$\begin{aligned}\nabla(g(x^{(1)},\ldots,x^{(r)}\cdot h(\cdot\cdot a^{(1)},f,\ldots,ax^{(r)},f)\\ = 2(g(x^{(1)},\ldots,x^{(r)}\cdot h(\cdot\cdot a^{(1)},f,\ldots,ax^{(r)},f)\cdot(\cdot\cdot^{(1)} \quad x^{(r)} \quad rx^{(r\ 1)})\\ (g(x^{(1)},\ldots,x^{(r)}\cdot \sum_j (\partial_j h(\cdot\cdot a^{(1)},f,\ldots,ax^{(r)},f)\cdot x^{(j)}.\end{aligned}$$

Now suppose h is a homogeneous polynomial of degree s. Then

$$\begin{aligned}D(g(x^{(1)},\ldots,x^{(r)}\cdot h(\cdot\cdot a^{(1)},f,\ldots,ax^{(r)},f)\cdot y\\ = 2(g(x^{(1)},\ldots,x^{(r)}\cdot h(\cdot\cdot a^{(1)},f,\ldots,ax^{(r)},f)\cdot(\cdot\cdot a^{(1)},f \quad ax^{(r)},f \quad r(ax^{(r\ 1)},f)\\ s(g(x^{(1)},\ldots,x^{(r)}\cdot h(\cdot\cdot a^{(1)},f,\ldots,ax^{(r)},f) \quad (7.10)\end{aligned}$$

by Euler's formula $\sum_{j\ 1}^{r} \partial_j h(x_1,\ldots,x_j\cdot x_j = s\ h$.

Now consider $h(z_1,\ldots,z_r\cdot = \sum_{\ 1}^{t} z_{\cdot}$. Let $u(x^{(1)},\ldots,x^{(r)}\cdot = g(x^{(1)},\ldots,x^{(r)}\cdot h(\ x^{(1)},e,\ldots,\ x^{(r)},e\cdot$. We have

$$\begin{aligned}\partial u(x^{(1)},\ldots,x^{(r)}\cdot f = 4(u(ax^{(1)}\cdot \quad x^{(r)}\cdot \quad rx^{(r\ 1)}\cdot x^{(1)}\cdot \quad x^{(r\ 1)}\cdot \quad (r\ 1)\cdot x^{(r\ 2)}\cdot f)\\ \frac{2}{2}(dg(\sum_{\cdot e\ 1}^{s} M(ax^{(i\cdot\cdot}\cdot f \quad ax^{(i\cdot e}\cdot x^{(1)}\cdot \quad x^{(r)}\cdot \quad rx^{(r\ 1)}\cdot f)i\\ (dg(\sum_{1 B \cdot e;\cdot e e B s} M(ax^{(i\cdot\cdot}\cdot f \quad ax^{(i\cdot e}\cdot x^{(i\cdot e e}\cdot f)i \quad (7.11)\end{aligned}$$

Applying the above to monomials $g(x^{(1)},\ldots,x^{(r)}\cdot = \sum_{\ 1}^{s} ax^{(i\cdot\cdot}\cdot x^{(i e}\cdot f$ and using induction, we have the following.

Lemma 24 (Probabilistic interpretation of $L^m f$). Let $f = f_2$ be as in equation 7.4. We have

$$L^m f(y\cdot = dx^{(1)}\cdot \sum_{s\ tBm} \sum_{i;i e\ 2m\ 1} \sum_{s;j > 2m\ 1\ t} a_{i;i e;j} \quad 2(s\ t\ m\cdot M_{\ 1}^{s}(ax^{(i\cdot\cdot}\cdot x^{(i e}\cdot f M_{\ 1}^{t}(ax^{(j\cdot\cdot}\cdot y,f)i \quad (7.12)$$

where $\sum_{i;i e;j} |a_{i;i e;j}| B 30^m m!^2$.

Proof. Recall that $Lf(x\cdot = c\ ^2Df(x\cdot x \quad f(x\cdot$. We induct on m. Suppose that the lemma holds for $m\ 1$; each term has at most a number of replicas $sB2m\ 1$ and $s;t Bm\ 1$. We consider the e ect of the map $f \to c\ ^2Df(y\cdot$ y equation 7.10 and equation 7.11 on a single term in equation 7.12. First, note that in each case, we obtain a factor ^{2}, and a factor of $c\ ^2$ for each additional $ax^{(i\cdot\cdot}\cdot x^{(i e}\cdot f$ term as well as $\cdot x^{(i\cdot\cdot}\cdot y e$ term (with a ^{2} factor if we remove a term); this justi es the $2(s\ t\ m\cdot$ factor.

1. In equation 7.10, the number of replicas s in the resulting terms increases by at most 1, t increases by at most 1, and the sum of absolute value of coe cients is at most

$$2r \quad sB2(2m\ 1\cdot(m\ 1\cdot B5m\ 3:$$

2. In equation 7.11, the number of replicas s in the resulting terms increases by at most 2, s increases by at most 1, and the sum of absolute value of coefficients is at most

$$\begin{aligned}
&2r\ 2^r\ 1 \cdot\ 2\ s\ 2r\ s^s\ 1 \cdot \\
&B\,4^{2m}\ 1 \cdot^{2m}\ 1 \cdot\ 2^m\ 1 \cdot^{2m}\ 1 \cdot\ ^m\ 1 \cdot^m\ 2 \cdot \\
&B\,16m^2\ \ 8m^2\ \ m^2\ B\,25m^2
\end{aligned}$$

The sum of absolute value of coefficients multiplies by at most $30m^2$. This finishes the induction step. □

7.4.2 Bounding $\|L^m f\|$

From Lemma 23, we know that that a bound on $\|L^{2m} f\|_{L^{p^2}}$ implies a low-degree approximation for f. Here we would like to bound $\|L^m f\|_{L^{p^2}}$ for all m. The kind of growth we get in m is captured by the following definition.

Definition 20. We say $f : \mathbb{R}^n \to \mathbb{R}^n$ is a $(r, \cdot)$-Gaussian-noise-sensitive function if for all $m > N$ and $p \geq 1$,

$$\|L^m f\|_{L^{p^2}} \cdot B r^{\frac{m}{2}} \cdot^m \max\left(r, \sqrt{mp}\cdot^m\right).$$

In the following lemma, we bound the moments of higher iterates of the Ornstein-Uhlenbeck operator L applied to the score function f under the Gaussian measure. This shows the Gaussian-noise-sensitivity property for f in the case where Q_0 is supported on a ball of radius R, forming a single cluster. We begin by switching from the Gaussian measure to the mixture measure using Lemma 213, and then leverage the probabilistic interpretation of $L^m f$ derived in Lemma 24, which is based on moments of replicas of the posterior distribution.

Lemma 25 (Control on iterates of the OU operator) Let $f = f_2$ be as in equation 7.4. Suppose Q_0 is supported on $B_R(0, \cdot)$, $R \geq$. Let P be the density of $Q_0 * N(0, I \cdot)$. Then for any $m > N$, we have the following:

$$\|L^m f\|_{L^p(P)} \cdot R\ O(E)^{\frac{1}{\cdot}} \cdot^2 \left(\frac{mR}{\cdot^2}\right)^2 (1 + \frac{(mp\cdot^{1/2})}{R})^m$$

$$\|L^m f\|_{L^{p^2}} \cdot BR\ O(E)^{\frac{1}{\cdot}} \cdot^2 \left(\frac{m^2 R}{\cdot^2}\right)^2 \max\left(\frac{R}{2}, (mp\cdot^{1/2})\right)^m$$

Therefore, f is $(O(R), \cdot)$-Gaussian-noise-sensitive.

Proof. In equation 7.12 in Lemma 24,

$$\max_{1 \leq i \leq s} \|x^{(i)} \cdot x^{(i)^T} f\| \cdot BR^{2s} \cdot BR^{2m}.$$

Note that the joint distribution of $(x^{(j)}, y)$ is the same for any j, namely, it is the distribution when $x^{(j)} \sim p_0$ and $y = x^{(j)}\beta^{(j)}$ when $\beta^{(j)} \sim N(0, I)$ is independent of $x^{(j)}$. Then for $p > N$,

$$
\begin{aligned}
&E\left(\max_{1 \le j \le t} |x^{(j)}\beta^{(j)}|\right)^p \\
&\le E_{x^{(1)} \sim p_0, y = x^{(1)}\beta^{(1)}, x^{(r)} \sim p, \beta^{(r)} \sim N(0, I), r \ge 1} \max_{1 \le j \le t} |x^{(j)}\beta^{(j)}|^p && \text{by Jensen's inequality} \\
&\le \sum_{j=1}^{t} E|x^{(j)}\beta^{(j)}|^{tp} && \text{by Hölder's inequality} \\
&= E|x^{(1)}\beta^{(1)}|^{tp} && (x^{(j)}\beta^{(j)}, y) \overset{d}{=} (x^{(1)}\beta^{(1)}, x^{(1)}\beta^{(1)}) \\
&= \sum_{k=0}^{tp} \binom{tp}{k} (x^{(1)}\beta^{(1)})^{tp-k} (-x^{(1)}\beta^{(1)})^k && \text{Binomial theorem} \\
&= \sum_{k=0}^{tp} \binom{tp}{k} R^{tp-k} R^k E_{X \sim N(0,S)} X_1^k \\
&= \sum_{k=0}^{tp} \binom{tp}{k} R^{tp-k} R^k (k-1)!! && \text{Gaussian moment bound} \\
&= \sum_{k=0}^{tp} \binom{tp}{k} R^{tp-k} R^k (tp)^{k/2} \le 2R^{tp} \left(\frac{(tp)^{1/2}}{R}\right)^{tp}.
\end{aligned}
$$

Consider one term $a_{i_1} \cdots a_{i_s} (x^{(1)})^{m_1} \cdots (x^{(s)})^{m_s} (x^{(i_1)})^{m_{i_1}} \cdots (x^{(i_s)})^{m_{i_s}} (x^{(j_1)})^{m_{j_1}} \cdots (x^{(j_t)})^{m_{j_t}} y^m$ in equation 7.12. We have by Jensen's inequality that

$$
\begin{aligned}
&\left|E\left[(x^{(1)})^{m_1} \cdots (x^{(s)})^{m_s} \max_{1 \le i \le s} |x^{(i)}\beta^{(i)}|^{m_i} \max_{1 \le j \le t} |x^{(j)}\beta^{(j)}|^{m_j}\right]\right| \\
&\le E\left|(x^{(1)})^{m_1} \cdots (x^{(s)})^{m_s} \max_{1 \le i \le s} |x^{(i)}\beta^{(i)}|^{m_i} \max_{1 \le j \le t} |x^{(j)}\beta^{(j)}|^{m_j}\right| \\
&\le 2R^{2s} R^{2t} \left(\frac{(tp)^{1/2}}{R}\right)^{t} \\
&\le \left(\frac{R}{2m}\right)^{2s} \left(\frac{(tp)^{1/2}}{R}\right)^{t} + \left(\frac{R}{2m}\right)^{2m} \left(\frac{(mp)^{1/2}}{R}\right)^{m},
\end{aligned}
$$

using $s \le t \le m$. Then

$$
\begin{aligned}
\|L^m\|_{L^p(P)} &\le 3^m m! \left(\frac{R}{2m}\right)^{2m} \left(\frac{(mp)^{1/2}}{R}\right)^{m} \\
&\le \left(\frac{1}{2}\right)^{2m} \left(\frac{mR}{2}\right)^{2m} \left(\frac{(mp)^{1/2}}{R}\right)^{m}
\end{aligned} \tag{7.13}
$$

By Hölder's inequality and Lemma 214 (appropriately scaled),

$$\|\nabla^m f\|_{L^{p'}(\mu)}^{p'} \le \int_{\mathbb{R}^n} \|\nabla^m f\|_2^{p'} \left(\frac{d\mu}{dP}\right)^{p'} dP \le \left\|\|\nabla^m f\|_2^{p'}\right\|_{L^{1+q'}(P)} \left\|\left(\frac{d\mu}{dP}\right)^{p'}\right\|_{L^{1+q}(P)}.$$

$$\le \left(\frac{mR}{\varepsilon}\right)^{2mp'} \left(1 + \frac{mp'q}{R}\right)^{m} e^{\frac{p'R}{2pq}}.$$

To optimize this bound, set $q = \frac{p'R}{pm}$. Then by Hölder's inequality,

$$\|\nabla^m f\|_{L^{p'}(\mu)}^{p'} \le e^{\frac{p'R}{2pq}} \left(\frac{1}{\varepsilon}\right)^{2mp'} \left(\frac{mR}{\varepsilon}\right)^{2mp'} \left(1 + \frac{mp'q}{R}\right)^{m}$$

$$\le e^{m/2} \left(\frac{1}{\varepsilon}\right)^{2mp'} \left(\frac{mR}{\varepsilon}\right)^{2mp'} \left(1 + \frac{mpq}{R}\right)^{m} \le \left(\frac{2mp}{R}\right)^{m}.$$

The first two terms give $R^{-O(p')} \left(\frac{mR}{\varepsilon}\right)^{2mp'}$, while the last term gives $R^{-O(p')} \left(\frac{m^2 R}{\varepsilon}\right)^{mp'}$. This completes the proof. □

7.4.3 Approximation with polynomial

Suppose we want to approximate $f = f_2$ (in equation 7.4) with a low-degree polynomial on the mixture distribution μ. We seek a low-degree polynomial g such that $\|f - g\|_{L^2(\mu)}$ is small. To do this, we first smooth f by the Ornstein-Uhlenbeck semigroup to obtain $P_t f$ (see equation 7.9), and then find a g that approximates $P_t f$. We can bound using Cauchy-Schwarz that

$$\|f - g\|_{L^2(\mu)}^2 \le 2\|f - P_t f\|_{L^2(\mu)}^2 + \|P_t f - g\|_{L^2(\mu)}^2$$

$$\le 2\|f - P_t f\|_{L^2(\mu)}^2 + \|P_t f - g\|_{L^2}^2 \cdot \|P_t f - g\|_{L^\infty}^{2-2/p'}.$$

Doing the smoothing ensures that we have better control over the term with higher-p' norm, $\|P_t f - g\|_{L^\infty}$. Choosing $t = \varepsilon$, $P_t f$ has exponential decay in coefficients with rate ε, so we can approximate $P_t f$ with a degree-$\tilde{O}(p')$ polynomial. To bound $\|f - P_t f\|_{L^2(\mu)}^2$ we bound its derivative:

$$-\frac{d}{dt}\|f - P_t f\|_{L^2(\mu)}^2 = 2\int_{\mathbb{R}^n} (f - P_t f)(-L P_t f)\, d\mu$$

$$= 2\|f - P_t f\|_{L^2(\mu)} \|L P_t f\|_{L^2(\mu)}$$

$$\le -\frac{d}{dt}\|f - P_t f\|_{L^2(\mu)} \|L P_t f\|_{L^2(\mu)}.$$

Focusing on the dependence of total error on ε, we can bound this with a change-of-measure inequality and Lemma 25, which, after integrating in t, gives error $O(\varepsilon)$. In order to obtain poly-logarithmic degree, we need a higher-order version of this argument. In preparation for Section 7.5, when we need to consider the norm with respect to a different measure, we state the following more generally.

Lemma 26. Suppose f is $O(r)$, γ-Gaussian-noise-stable and μ is a measure such that for all $a \in [0, \left\|\frac{d\mu}{d\gamma}\right\|_{L^{1+a}}]$, $B_\mu(e^{-ar^2/2})$ (for γ_2) (e.g., from Lemma 214, $\mu \ll N(0, I)$ where μ is supported on $B_r(0)$), where $r \geq 1$. There is a polynomial g of degree at most $O(\frac{1}{\varepsilon^4} \ln(c) \cdot \max(r^2, \ln(c)))$ such that

$$\|f - g\|^2_{\mu} \leq \varepsilon^2 \|f\|^2_{\mu} \quad \text{and} \quad \|g\|^2_{\mu} \leq \|f\|^2_{\mu}.$$

Proof. To get a better bound, we use a more clever smoothing strategy. For some δ to be defined, we will bound

$$\|f - g\|^2_{\mu} \leq 2\|f - f_\delta\|^2_{\mu} + 2\|f_\delta - g\|^2_{\mu}, \tag{7.14}$$

where g is a polynomial approximation of f_δ obtained by truncating the Hermite expansion.

To define f_δ, we approximate f with a numerical differentiation formula for a higher derivative. Define the finite difference by $\Delta_{x,h} g(x) = g(x) - g(x - h)$. We can write this as $\Delta_h g = T_h g - g$, where $T_h g(x) = g(x - h)$. Suppose that $g \in C^m$. By the Binomial Theorem on $T_h = \mathrm{id} - \Delta_h$ and Taylor's Theorem,

$$\begin{aligned}
\Delta_h^m g(0) &= \sum_{j=0}^{m} \binom{m}{j} (-1)^{m-j} \sum_{i=0}^{m-1} g^{(i)}(0) \frac{(jh)^i}{i!} + \frac{g^{(m)}(\xi_j)}{m!} (jh)^m \quad \text{for some } \xi_j \in (0, jh) \\
&\left| \sum_{i=0}^{m-1} \frac{g^{(i)}(0)}{i!} \Delta_{x,h} x^i \right| = 0 \leq h^m \sum_{j=0}^{m} \binom{m}{j} (-1)^{m-j} \frac{g^{(j)}(\xi_j)}{m!} j^m \\
&h^m \sum_{j=0}^{m} \binom{m}{j} (-1)^{m-j} \frac{g^{(j)}(\xi_j)}{m!} j^m
\end{aligned}$$

where in the last step we use the fact that finite differencing reduces the degree of a polynomial by 1, the mth finite difference of polynomials of degree $< m$ is 0. Hence

$$|\Delta_h^m g(0)| \leq h^m \sum_{j=0}^{m} \binom{m}{j} \left(\frac{ej}{m}\right)^m \max_{\xi \in (0, mh)} |g^{(m)}(\xi)| \leq h^m 2e^m \max_{\xi \in (0, mh)} |g^{(m)}(\xi)|$$

Let $f_\delta = \sum_{j=1}^{m} (-1)^{j-1} \binom{m}{j} P_{jh} f$. Note $f - f_\delta = (-1)^m \Delta_h^m P_t f = 0$. We then have (noting that P_s and L commute)

$$\begin{aligned}
\|f - f_\delta\|^2_{\mu} &\leq 2e^{2m} \int_{\mathbb{R}^n} \max_{t \in (0, mh)} \left| \frac{d^m}{dt^m} P_t f(x) \right|^2 d\mu(x) \\
&\leq 2e^{2m} \int_{\mathbb{R}^n} \left| \frac{d^m}{dt^m} P_t f(x) - 0 \right|^2 d\mu(x) \\
&\leq 2e^{2m} \int_{\mathbb{R}^n} \left| \int_0^{mh} \frac{d^{m+1}}{ds^{m+1}} P_s f(x)\, ds \right|^2 d\mu(x) \\
&\leq 2e^{2m} \int_{\mathbb{R}^n} \left| L^m f(x) \right|^2 d\mu(x) \cdot mh \int_0^{mh} \int_{\mathbb{R}^n} \left| P_s L^m f(x) \right|^2 d\mu(x)\, ds. \qquad (7.15)
\end{aligned}$$

First we bound the first term and the second term when $s = 0$. We use the Gaussian noise sensitivity assumption, the assumption on μ, and Hölder's inequality to obtain

$$\begin{aligned}
\int_{\mathbb{R}^n} \left| L^l f \right|^2 d\mu &\leq \|L^l f\|^2_{L^{p/(p-1)}} \cdot \left\| \frac{d\mu}{d\gamma} \right\|_{L^{\frac{p}{p-1}}} \\
&\leq r^{2l} \left(\frac{l^2}{2}\right)^l \max(r^2, l-1)^{\frac{p-2}{2p}} e^{\frac{r^2}{2p}}. \qquad (7.16)
\end{aligned}$$

To get a bound for the second term when $s > 0$, we bound the following derivative, again using the Gaussian noise sensitivity assumption and Hölder's inequality:

$$\begin{aligned}
\frac{d}{ds} \int_{\mathbb{R}^n} (P_s L^l f)^2 \, d\gamma &= \frac{d}{ds} \int_{\mathbb{R}^n} 2a P_s L^l f \cdot P_s L^{l-1} f \, d\gamma \\
&\le 2 \| P_s L^l f \|_{L^{2p/(p-1)}} \cdot \| P_s L^{l-1} f \|_{L^{2p/(p-1)}} \cdot \left\| \frac{d}{ds} \right\|_{L^{\frac{p}{p-1}}} \cdot \\
&\le 2 \| L^l f \|_{L^{2p/(p-1)}} \cdot \| L^{l-1} f \|_{L^{2p/(p-1)}} \cdot \left\| \frac{d}{ds} \right\|_{L^{\frac{p}{p-1}}} \cdot \qquad (7.17) \\
&\le 2 r^2 e^{-2s} \left(\frac{r^2}{2} \right)^{2l} \max(r^2, l)^{l-1} \, p \, e^{\frac{r^2}{2p}}; \qquad (7.18)
\end{aligned}$$

where in equation 7.17 we use the fact that for $q > 1$, $\|P_s g\|_{L^q}$ is monotonically decreasing in s. In both equation 7.16 and equation 7.18, we optimize the bound by taking $p = \frac{r^2}{l}$ to obtain

$$\begin{aligned}
\int_{\mathbb{R}^n} (L^l f)^2 \, d\gamma &\le r^2 e^{-2s} \left(\frac{r^2}{2} \right)^{2l} \max(r^2, l)^{l}, && l \ge m, m \ge 1 \\
\int_{\mathbb{R}^n} (L^l P_s f)^2 \, d\gamma &\le r^2 e^{-2s} \left(\frac{r^2}{2} \right)^{2l} \max(r^2, l)^{l}, && l \ge m \ge 1.
\end{aligned}$$

Integrating the last inequality twice and substituting into equation 7.15 gives

$$\begin{aligned}
\|f\|^2 - \|\mathbb{E} f\|^2 &\le \sum_{l=1}^{m-1} (2l+3)! \left(\frac{r^2}{2} \right)^{2l} \max(r^2, l)^{l} + (2m+3)! \left(\frac{r^2}{2} \right)^{2m} \max(r^2, m)^{m} \\
&\le r^2 e^{2m} \left(\frac{hrm^2 \max(r, m)}{2} \right)^{2m} \qquad (7.19)
\end{aligned}$$

when $mh \le 1$. We choose $m = \ln c$ and $h = \frac{c^2}{rm^2 \max(r, m)}$ for an appropriate constant c to get

$$\|f\|^2 - \|\mathbb{E} f\|^2 \le r^2 c^2.$$

Write $f = \sum_{k \in \mathbb{N}_0^d} a_k h_k$ where $a_k \in \mathbb{R}^d$, and let $p_l = \sum_{|k|=l} a_k h_k$, so that $f = \sum_{l=0}^{n} p_l$. Then

$$f = \sum_{k \in \mathbb{N}_0^n} b_k h_k \quad \text{for} \quad b_k = \sum_{j=1}^{m} (-1)^{j-1} \binom{m}{j} e^{j h |k| / 2} a_k = \left(1 - (1 - e^{-h|k|/2})^m\right) a_k. \qquad (7.20)$$

Let $q_l = \sum_{|k|=l} b_k h_k$. Note

$$\|q_l\|^2 = \sum_{|k|=l} n b_k^2 = \sum_{|k|=l} \left(1 - (1 - e^{-h|k|/2})^m\right)^2 n a_k^2 \le m^2 e^{2hl/2} \|p_l\|^2.$$

We approximate f_C with $g = \sum_{k \le L} b_k h_k$ where $L = C h^{-1} \ln(c/h)$ for an appropriate constant C. We have by Cauchy-Schwarz and Lemma 213 that

$$\begin{aligned}
\|f_C - g\|_{L^2(P_X)}^2 &= \int_{\mathbb{R}^n} \Big(\sum_{k > L} b_k h_k \Big)^2 dx \\
&\le \Big(\sum_{k > L} e^{-hk} \Big) \Big(\sum_{k > L} e^{hk} \int_{\mathbb{R}^n} |h_k|^2 dx \Big) \\
&\le \Big(\sum_{k > L} e^{-hk} \Big) \|f_C\|_{L^2(P_X)}^2 e^{2h} \\
&\le \big(e^{-hL} / (1 - e^{-h}) \big) \|f_C\|_{L^2(P_X)}^2 e^{2h} \\
&\le \max(1, e^{2h - hL}) \|f_C\|_{L^2(P_X)}^2 \le \max(1, e^{2h}) h^{-1} \|f_C\|_{L^2(P_X)}^2
\end{aligned} \tag{7.21}$$

when we take $L = C \frac{4r^2}{h^2} \ge r^{-4} \ln(c/h) \cdot 4 \max(r^{-2}, \ln(c/h))$ for an appropriate constant C. Note that the Gaussian-noise-sensitive property also implies $\|f_C\|_{L^2(P_X)} \le r^{-2}$. Plugging into equation 7.14 the inequalities equation 7.19 and equation 7.21,

$$\|f - g\|_{L^2(P_X)}^2 \le \|f - f_C\|_{L^2(P_X)}^2 + \|f_C - g\|_{L^2(P_X)}^2 \le \|f - f_C\|_{L^2(P_X)}^2 + r^{-2} h^2.$$

Choosing constants appropriately then gives the desired bound. Finally note that Equation equation 7.20 implies $|a_k| \le |b_k|$ for all $k > N_0^n$. Hence

$$\|g\|_{L^2(P_X)}^2 \le \|f_C\|_{L^2(P_X)}^2 + \sum_{k > N_0^n} n b_k^2 \le \sum_{k > N_0^n} n a_k^2 \le \|f\|_{L^2(P_X)}^2.$$

□

Combining Lemmas 25 and 26 implies a low degree approximation of order $\tilde{O}(n^{-R}; \ln(n)^{1/2})$ for the single cluster case.

7.5 From one to multiple clusters

To handle the general case of multiple clusters, we approximate the score function using a piecewise low-degree polynomial over a set of warm-starts. Before applying Lemma 26, we locally approximate the score within the Voronoi cell corresponding to each warm-start $\hat{\theta}$ by f_{loc}. We can then approximate f_{loc} using a low-degree polynomial by controlling the terms $\|L^m f_{loc}(\hat{\theta}) - \theta\|_{L^{p'}(\hat{\theta})}$, where f_{loc} is shifted by the warm start $\hat{\theta}$. We start with the definition of the warm starts.

Definition 21. Let $C > R$. We say C is a complete set of R-warm starts for Q_0 if

$$Q_0(\theta) \le \sum_{\hat{\theta} \in C} B_R(\hat{\theta}, \theta) \le 1.$$

In other words, for all θ in the support of Q_0, there exists $\hat{\theta} \in C$ such that $\theta \in B_R(\hat{\theta})$.

De nition 22. Let $C = \{\hat{A}_1, \ldots, \hat{A}_{k(e)}\}$. The Voronoi partition $V_1, \ldots, V_{k(e)}$ corresponding to C is de ned by $V_j = \{x \in \mathbb{R}^n : \|x - \hat{A}_j\| = \min_{1 \le j \le k(e)} \|x - \hat{A}_j\|\}$.

Note that up to the boundaries (which are a measure 0 set), this induces a partition of $\mathbb{R}^n$.

To tackle the multiple cluster setting, we rst suppose that we have a complete set of R-warm starts $C = \{\hat{A}_1, \ldots, \hat{A}_{k(e)}\}$, and let $V_1, \ldots, V_{k(e)}$ be the corresponding Voronoi partition. For a probability measure P and set S, de ne the unnormalized restriction by $P_S(A) = P(A \cap S)$ and the normalized restriction by $P_S(A) = \frac{P(A \cap S)}{P(S)}$. De ne $P^S = Q_0(S) * N(0, \sigma^2 I)$ that is, we do the restriction before the convolution. De ne

$$f_{S,\sigma^2}(y) = E_{Y \sim Q_0(S) * N(0,\sigma^2 I)}\left[\nabla \ln p^S(y) \cdot (Y - y)(Y - y)^T\right].$$

When σ is understood, we omit it from the subscript.

We need to show that we still have a good polynomial approximation for f under the measure $P(S_i) = Q_0 * N(0, \sigma^2 I)(V_i)$. By the analysis for one cluster, we have good approximation of f_{V_i,σ^2} under P^{V_i}. To obtain the result for multiple clusters, we need to show that this approximation is preserved even when we consider f instead of f_{V_i,σ^2} and $P(S_i)$ instead of P^{V_i}, i.e., deal with the leakage into V_i from the other Voronoi cells after Q_0 is convolved with $N(0, \sigma^2 I)$, in both the score function and the measure which the norm is with respect to. We do this in Sections 7.5.1 and 7.5.2, respectively.

7.5.1 Estimation with the "nearby" score

We will actually bound $\|f - f_{S_i}\|_{L^2(P_{V_i})}$, for an expanded neighborhood S_i of $\hat{A}_i$, allowing an extra "buffer region" where mass is allowed to leak in.

Lemma 27. Suppose that $C = \{\hat{A}_1, \ldots, \hat{A}_{k(e)}\}$ is a complete set of R-warm starts for Q_0, and let $V_1, \ldots, V_{k(e)}$ be the corresponding Voronoi partition. Suppose Q_0 is supported on $B(M, 0)$. Given $R(e) > R$, let $S_i = B(R(e), \hat{A}_i)$ and de ne

$$f^{C,R(e)}_{loc,\sigma^2}(y) = f_{S_i,\sigma^2}(y) \text{ when } y \in V_i,$$

$$\text{where } f_{S_i,\sigma^2}(y) = E_{Y \sim Q_0(S_i) * N(0,\sigma^2 I)}\left[\nabla \ln p^{S_i}(y) \cdot (Y - y)(Y - y)^T\right].$$

We write f_{loc,σ^2} when $C, R(e)$ are clear. Then for $R(e) \ge 3R + 2\sqrt{2}\sigma\sqrt{\ln(k(e)/\varepsilon)}$,

$$\int \|f - f_{loc,\sigma^2}\|^2_{L^2(P)} \le 32\varepsilon^2 + 6R(e)^2 \int \|s(x) + x/\sigma^2\|^2 \, dP(x) \cdot \exp\left(-\frac{R^2}{2\sigma^2}\right).$$

Note that

$$\int \|f - f_{loc,\sigma^2}\|^2_{L^2(P)} = \sum_{i=1}^{k(e)} \int \|f - f_{S_i,\sigma^2}\|^2_{L^2(P_{V_i})},$$

so this gives a bound for the L^2 norm within each Voronoi cell.

To prove this, we rst show that with high probability, Y does not stray too far from the Voronoi cell of $\hat{A}_i$.

Lemma 28. Let C ˜Â,:::; Âœ• and $V_1;::::;V_{k^{œ}}$ be the corresponding Voronoi partition. Suppose that n Âi BR. De ne Y where N ˆ;I •, and let i œbe such that Y >Viœ: Then with probability C1 ,

$$n \quad Â_{iœ}B3R \quad 2 \quad \overline{2} \quad ¾\sqrt{\ln‹\frac{k^{œ}}{—}•}:$$

In other words, with high probability, even if Y >Viœfor i œx i (i.e., adding a Gaussian brings the point to a di erent Voronoi cell), will not be too far from the center of the new cell Âiœ:

Proof. Let $v_{ii^{œ}} \quad \frac{Â_{iœ} Â_i}{n Â_{iœ} Â_i}$ be the unit vector pointing from Âi toward Âiœ: Since >Vi, we have

$$` \quad Â_i; v_{ii^{œ}}eB \frac{n Â_{iœ} \quad Â_i}{2}:$$

By Gaussian tail bounds and a union bound, with probability C1 , we have that for all iœ;

$$`Y \quad Â_i; v_{ii^{œ}}e \quad ` \quad Â_i; v_{ii^{œ}}eB` \quad Â_i; v_{ii^{œ}}e \quad o \quad \overline{2} \quad ¾\sqrt{\ln‹\frac{k^{œ}}{—}•} \quad BR \quad o \quad \overline{2} \quad ¾\sqrt{\ln‹\frac{k^{œ}}{—}•}$$

Therefore, if y >Viœ, then we have

$$\frac{n Â_{iœ} \quad Â_i}{2} B`Y \quad Â_i; v_{ii^{œ}}eBR \quad o \quad \overline{2} \quad ¾\sqrt{\ln‹\frac{k^{œ}}{—}•}$$

and

$$n \quad Â_{iœ}Bn \quad Â_i \quad n Â_i \quad Â_{iœ}B3R \quad 2 \quad \overline{2} \quad ¾\sqrt{\ln‹\frac{k^{œ}}{—}•}:$$

□

Proof of Lemma 27. Given , let i be such that y >Vi. We note

$$\begin{aligned} f_{\ 2}ˆy• \quad & E\ 1_{S_i}ˆ • \ SY \quad y \quad E\ 1_{S_i^c}ˆ • \ SY \quad y \\ f_{loc;\ 2}ˆy• \quad & Pˆ \ >S_i \ SY \quad y•f_{loc;\ 2}ˆy• \quad Pˆ \ ¶S_i \ SY \quad y•f_{loc;\ 2}ˆy• \\ & Pˆ \ >S_i \ SY \quad y•\frac{E\ 1_{S_i}ˆ • \ SY \quad y}{E\ 1_{S_i}ˆ • SY \quad y} \quad Pˆ \ ¶S_i \ SY \quad y•f_{loc;\ 2}ˆy• \\ & E\ 1_{S_i}ˆ • \ SY \quad y \quad E\ 1_{S_i^c}ˆ • E_{\ œQ_0S_i} \ œSY \quad y \ SY \quad y \end{aligned}$$

Hence, using the fact that $P_{\hat{}}(y \mid S_i) \leq B$, we have

$$\begin{aligned}
\mathbb{E}_{f_{\mathrm{loc}}} Z^2 &\leq L^2 P \left(\mathbb{E}_y \mathbb{E}_{Q_0} \mathbb{1}_{S_i^c}(\hat{}) \cdot \| \nabla \log q_{\hat{\theta}_i}(X, y) \|_2^2 \right) \\
&\leq B \mathbb{E}_{(y)} \left(\mathbb{1}_{S_i^c}(\hat{}) \cdot \| \nabla \log q_{\hat{\theta}_i}(X, y) \|_2^2 \right) \\
&\leq 2B \mathbb{E}_{(y)} \left(\mathbb{1}_{S_i^c}(\hat{}) \cdot \| \nabla \log q_{\hat{\theta}_i}(X, y) - \nabla \log q_{\hat{\theta}_i}(X, y) \|_2^2 \right) \\
&\leq 2B \mathbb{E}_{(y)} \left(\mathbb{1}_{S_i^c}(\hat{}) \cdot \| \nabla \log q_{\hat{\theta}_i}(X, y) \|_2^2 \right) \\
&\leq 2B \int_0^a P_{(y)}(\{ \| \nabla \log q_{\hat{\theta}_i}(X, y) \|_2^2 \geq t \}) \, dt + 6R^2 a \\
&\leq 2B \int_0^a P_{(y)}(\{ \| \nabla \log q_{\hat{\theta}_i}(X, y) \|_2 \geq R + \sqrt{t/2} \}) \, dt + 6R^2 a. \qquad (7.22)
\end{aligned}$$

But using Lemma 28, we have

$$P(\{ \| \nabla \log q_{\hat{\theta}_i}(X, y) \|_2 \geq 3R \}) \leq k e^{-\frac{r^2}{8}},$$

which implies

$$P_{(y)}(\{ \| \nabla \log q_{\hat{\theta}_i}(X, y) \|_2 \geq R + \sqrt{\tfrac{a}{2}} \}) \leq k e^{-\frac{(R + \sqrt{a/2})^2}{8} \ln \frac{k}{\delta}} \leq k e^{-\frac{8 \ln \frac{k}{\delta} \cdot a}{8}} = e^{-\frac{a}{16}}. \qquad (7.23)$$

Plugging Equation equation 7.23 into Equation equation 7.22,

$$\mathbb{E}_{f_{\mathrm{loc}}} Z^2 \leq L^2 P \left(2B \int_0^a e^{-\frac{t}{16}} \, dt + 6R^2 a \right) \leq 32 + 6R^2 a.$$

□

7.5.2 Approximation within a Voronoi cell

In the multiple cluster case, in order to apply Lemma 25 for approximating the score function in the Voronoi cell of a warm-start $\hat{\theta}_i$ by a low-degree polynomial, we also need to bound the Rényi divergence of the mixture measure P restricted to that Voronoi cell of $\hat{\theta}_i$, with respect to the Gaussian measure around $\hat{\theta}_i$.

Lemma 29. Let $C = \{\hat{\theta}_1, \ldots, \hat{\theta}_k\}$ and $V_1, \ldots, V_k$ be the corresponding Voronoi partition. Suppose that C is a complete set of R-warm starts. Then

$$S_{V_i}(\alpha) \leq \left\| \frac{dP_{V_i}}{d\mathcal{N}_{\hat{\theta}_i, 1}} \right\|_{\alpha}^{\alpha - 1} \leq \exp\left(\frac{\alpha R^2}{2} \right). \qquad (7.24)$$

Proof. We can write

$$P_{V_i} = Q_0(V_i) \cdot \frac{P^{V_i}}{Q_0(V_i)} + Q_0(V_i^c) \cdot \frac{P^{V_i^c}}{Q_0(V_i^c)},$$

note that $Q_0(V_i)$ and $Q_0(V_i^c)$ are the normalizing constants for the respective probability measures. By convexity, it suffices to show that equation 7.24 holds for P^{V_i} and $P^{V_i^c}$ in place of P_{V_i}.

For P_{V_i}, equation 7.24 follows directly from Lemma 214. For $P_{V_i}^c$, noting that it is a convex mixture of $N(\cdot; \cdot)$ for $\mu_j \in V_i$, again by convexity it suffices to show that equation 7.24 holds for one such $N(\cdot; \cdot)$. Suppose that $\mu_j \in V_j$, $j \neq i$. Note that by definition of the Voronoi cell, $\|\mu_j - \hat{\mu}_i\| \geq \|\mu_j - \hat{\mu}_j\|$ so $\frac{d\|\mu_j - \hat{\mu}_i\|}{d\|\hat{\mu}_i - \hat{\mu}_j\|} \geq 1$ on V_i. Then

$$\begin{aligned} \int_{V_i} \left(\frac{d\|\mu_j - \hat{\mu}_i\|}{d\|\hat{\mu}_i - \hat{\mu}_j\|}\right)^{p-1} d\hat{\mu}_i &\leq \int_{V_i} \left(\frac{d\|\mu_j - \hat{\mu}_i\|}{d\|\hat{\mu}_i - \hat{\mu}_j\|}\right)^{p} d\mu_j \\ &\leq \int_{V_i} \left(\frac{d\|\mu_j - \hat{\mu}_i\|}{d\|\hat{\mu}_i - \hat{\mu}_j\|}\right)^{p} d\mu_j \\ &\leq \int_{V_i} \left(\frac{d\|\mu_j - \hat{\mu}_i\|}{d\|\hat{\mu}_i - \hat{\mu}_j\|}\right)^{p-1} d\hat{\mu}_i \leq e^{\frac{aR^2}{2}} \end{aligned} \tag{7.25}$$

where the last step follows from Lemma 214, as $\|\mu_j - \hat{\mu}_i\| \leq R$. □

Lemma 30 (Existence of low-degree polynomial) Suppose that $f = f_2$ as in equation 7.4, Q_0 is supported on B_D, $C = \{\hat{\mu}_1, \ldots, \hat{\mu}_{k^\alpha}\}$ is a complete set of R-warm starts for Q_0, and let $V_1, \ldots, V_{k^\alpha}$ be the corresponding Voronoi partition. For each i, there is a polynomial g_i of degree at most $O\left(\frac{R}{\sigma}\sqrt{\ln(k^\alpha/\delta)}\sqrt{\ln(c/\epsilon)}^4\right)$ such that

$$\|f - g_i\|^2_{P_{V_i}} \leq \epsilon^2 R^2 \quad \text{and} \quad \|g_i\|_{\hat{\mu}_i, 2} \leq D.$$

Proof. Let $R' = 3R + 2\sqrt{2}\sqrt{\ln(k^\alpha/\delta)}$. First, note that from Lemma 27,

$$\sum_{i=1}^{k^\alpha} \|f\|^2_{S_i, 2} \|Z\|^2_{L^2(P_{V_i})} \leq \|f\|^2_{\text{loc}, 2} \|Z\|^2_{L^2(P)} \leq \epsilon^2 R^2 \sqrt{\ln(k^\alpha/\delta)}^2, \tag{7.26}$$

where S_i is the expanded neighborhood in Lemma 27. By Lemma 26 applied to P_{V_i}, $f_{\text{loc}, 2}$ is $(O(R'), \delta)$-Gaussian-noise-sensitive. By Lemma 29, for each i, $\left\|\frac{dP_{V_i}}{d\|\hat{\mu}_i, 2\|}\right\|_{L^1(P_{\hat{\mu}_i, 2})} \leq e^{\frac{a(R'-\epsilon)^2}{2}}$. Hence P_{V_i} satisfies the conditions of Lemma 26, and there exists a polynomial g_i of degree at most

$$O\left(\frac{R'}{\epsilon}\right)^4 \ln(c/\epsilon)^4 \max\left(\left(\frac{R'}{\epsilon}\right)^2; \ln(c/\epsilon)\right) \leq O\left(\frac{R'}{\epsilon}\right)^6 \ln(c/\epsilon)^4,$$

such that

$$\|f_{\text{loc}, 2}\|^2_{g_i} \leq \|f_{S_i}\|^2_{g_i} \leq \|f_{\text{loc}, 2}\|^2_{g_i} + \epsilon^2 R'^2. \tag{7.27}$$

Combining equation 7.26 and equation 7.27 gives the result. Finally, from Lemma 26,

$$\|g_i\|_{\hat{\mu}_i, 2} \leq \|f_{\text{loc}, 2}\|_{P_{V_i}} \leq D.$$

□

7.5.3 Piecewise polynomial regression

Thus far, we have demonstrated the existence of low-degree polynomials in the Voronoi cells corresponding to the warm starts, that provide a good approximation of the score function with respect to the mixture measure in each cell. Now we investigate how many samples we need, so that applying polynomial regression separately in each Voronoi cell enables us to recover the coe cients of these low-degree polynomials accurately. For (x_i, y_i) generated according to equation 7.2, by equation 7.6 we wish to find the least-squares solution to

$$y_i \approx s_t(y_i)\, x_i.$$

where by equation 7.4 we parameterize

$$f_\theta(y) = \sigma_t^2(y) \sum_{|k|\le d} b_k h_k(y) - \hat{s}(y);$$

where $\hat{s}$ is a warm start. Equivalently, we wish to find the least-squares solution to

$$\sum_{|k|\le d} b_k h_k(y_i) - \hat{s}(y_i) \approx \frac{x_i}{\sigma_t^2(y_i)} - \hat{s}(y_i) \approx \frac{x_i}{\sigma_t^2(y_i)} - \frac{y_i}{\sigma_t^2}.$$

We can restrict to $(b_k)_{|k|\le d} \in B_R$ (in $\mathbb{R}^{\binom{n+d}{d}}$), so we solve

$$\arg\min_{(b_k)_{|k|\le d} \in B_R} \hat{L}(b_k)_{|k|\le d} \quad \text{where} \quad \hat{L}(b_k)_{|k|\le d} = \frac{1}{N}\sum_{i=1}^{N} \Big(\sum_{|k|\le d} b_k h_k(y_i) - \hat{s}(y_i) \Big)^2 \approx \Big(\frac{x_i}{\sigma_t^2(y_i)} - \frac{y_i}{\sigma_t^2} \Big)^2.$$

Let $\bar{x}_i = \mathbb{E}_{X \sim p_0}(x_i \mid y_i) = y_i + \sigma_t^2 s_t(y_i)$ and $x_i = \bar{x}_i + \varepsilon_i$, so that conditioned on y_i, ε_i is mean-zero noise. Let $\sigma_i = \frac{\sigma_t^2}{\sigma_t^2}$. Let $z_i = \mathbb{E}_{X \sim p_0}(y_i) = \frac{1}{\sigma_t^2} y_i - \frac{\sigma_t^2}{\sigma_t^2} \bar{x}_i$. Then, for each Voronoi cell V_j, we calculate the empirical loss for samples where y_i falls into V_j,

$$\hat{L}^{(j)}(b_k)_{|k|\le d} = \frac{1}{N} \sum_{i:\, y_i \in V_j} \sum_{|k|\le d} b_k h_k(y_i) - \hat{s}_j - z_i - \varepsilon_i)^2;$$

and the empirical risk minimizer (ERM) for the jth Voronoi cell as

$$(\hat{b}_k^{(j)})_{|k|\le d} \in \arg\min_{(b_k)_{|k|\le d} \in B_M} \hat{L}^{(j)}(b_k)_{|k|\le d}. \tag{7.28}$$

We then combine these ERM solutions for all Voronoi cells and define the following piecewise polynomial function to approximate the score:

$$\hat{g}(y) = \sum_{j=1}^{k^{(n)}} 1_{V_j}(y) \sum_{|k|\le d} \hat{b}_k^{(j)} h_k(y) - \hat{s}_j, \tag{7.29}$$

where $k^{(n)}$ is the total number of Voronoi cells. To show that $\hat{g}$ is a decent approximation for the score function f_θ, we need to bound the generalization error of these ERM solutions equation 7.28. To do this, we derive moment bounds for the features and the noise. Let

$$L^{(j)}(b_k)_{|k|\le d} = \mathbb{E} \sum_{|k|\le d} 1_{V_j}(Y) \Big(\sum_{|k|\le d} b_k h_k(Y) - \hat{s}_j - Z - \varepsilon \Big)^2$$

denote the population risk in V_j. We further define the shifted error of the empirical loss with respect to the population loss as

$$\mathrm{err}^{(j)}(h_k(\cdot;S_{-V_j})) = \hat{L}^{(j)}(h_k(\cdot;S_{-V_j})) - L^{(j)}(h_k(\cdot;S_{-V_j})) = \frac{1}{N}\sum_{i:z_i\in V_j} \frac{1}{|V_j|}\sum_{z\in S_{-V_j}} \ell(z_i;z) - \frac{1}{N}\sum_{i:z_i\in V_j} \mathbb{E}_{S_{-V_j}}[\ell(z_i;z)]$$

$$= \frac{1}{N}\sum_{i:z_i\in V_j} \frac{1}{|V_j|}\sum_{z\in S_{-V_j}} \ell(h_k(\cdot;S_{-V_j}),z_i) - \mathbb{E}_{S_{-V_j}}[\ell(h_k(\cdot;S_{-V_j}),z_i)]$$

$$= \frac{1}{N}\sum_{i:z_i\in V_j} \left[\frac{1}{|V_j|}\sum_{z\in S_{-V_j}} \ell(h_k(\cdot;S_{-V_j}),z_i) - \mathbb{E}_{S_{-V_j}}[\ell(h_k(\cdot;S_{-V_j}),z_i)]\right]. \quad (7.30)$$

The following lemma is standard and relates the generalization gap of ERM to $\mathrm{err}^{(j)}(h_k(\cdot;S_{-V_j}))$. As we see shortly, the terms in the definition equation 7.30 that do not depend on $h_k(\cdot;S_{-V_j})$ cancel out in the calculation of the generalization gap.

Lemma 31 (Uniform convergence $\Rightarrow$ Generalization gap) The generalization gap for the ERM solution can be bounded as

$$L^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) - \hat{L}^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) \le \min_{h_k(\cdot;S_{-V_j})\in\mathcal{B}_M} L^{(j)}(h_k(\cdot;S_{-V_j})) + 2\max_{h_k(\cdot;S_{-V_j})\in\mathcal{B}_M} \mathrm{err}(h_k(\cdot;S_{-V_j})).$$

Proof. To bound the generalization gap of the ERM solution $\hat{h}_k^{(j)}(\cdot;S_{-V_j}), 1\le j\le n$ output by Algorithm ??, we compare its loss with the loss of an arbitrary point $h_k(\cdot;S_{-V_j})$ in $\mathcal{B}_L(0)$:

$$L^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) = L^{(j)}(h_k(\cdot;S_{-V_j})) + L^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) - \hat{L}^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) + \hat{L}^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) - \hat{L}^{(j)}(h_k(\cdot;S_{-V_j})) + \hat{L}^{(j)}(h_k(\cdot;S_{-V_j})) - L^{(j)}(h_k(\cdot;S_{-V_j}))$$

$$\le L^{(j)}(h_k(\cdot;S_{-V_j})) + \mathrm{err}^{(j)}(h_k(\cdot;S_{-V_j})) - \mathrm{err}^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) + \hat{L}^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) - \hat{L}^{(j)}(h_k(\cdot;S_{-V_j}))$$

$$\le L^{(j)}(h_k(\cdot;S_{-V_j})) + \mathrm{err}^{(j)}(h_k(\cdot;S_{-V_j})) - \mathrm{err}^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) + 0$$

$$\le L^{(j)}(h_k(\cdot;S_{-V_j})) + \mathrm{err}^{(j)}(h_k(\cdot;S_{-V_j})) + 2\max_{h_k(\cdot;S_{-V_j})\in\mathcal{B}_M} \mathrm{err}^{(j)}(h_k(\cdot;S_{-V_j})). \quad (7.31)$$

Note that $\hat{L}^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) - \hat{L}^{(j)}(h_k(\cdot;S_{-V_j})) \le 0$ by the definition of $\hat{h}_k^{(j)}(\cdot;S_{-V_j})$ as a minimizer of $\hat{L}^{(j)}(\cdot)$. Furthermore, by equation 7.30,

$$L^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) - \hat{L}^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) + \hat{L}^{(j)}(h_k(\cdot;S_{-V_j})) - L(h_k(\cdot;S_{-V_j}))$$

$$\le \mathrm{err}^{(j)}(h_k(\cdot;S_{-V_j})) - \mathrm{err}^{(j)}(\hat{h}_k^{(j)}(\cdot;S_{-V_j})) + 2\max_{h_k(\cdot;S_{-V_j})\in\mathcal{B}_M(0)} \mathrm{err}^{(j)}(h_k(\cdot;S_{-V_j})). \quad (7.32)$$

□

Based on Lemma 31 and Equation equation 7.30, to bound the generalization gap it suffices to bound

$$\max_{h_k(\cdot;S_{-V_j})\in\mathcal{B}_M} \mathrm{err}(h_k(\cdot;S_{-V_j})) \le \max_{h_k(\cdot;S_{-V_j})\in\mathcal{B}_M} \frac{1}{N}\sum_{i:z_i\in V_j} \frac{1}{|V_j|}\sum_{z\in S_{-V_j}} \ell(h_k(\cdot;S_{-V_j}),z_i) - \mathbb{E}_{S_{-V_j}}[\ell(h_k(\cdot;S_{-V_j}),z_i)]$$

$$+ \max_{h_k(\cdot;S_{-V_j})\in\mathcal{B}_M} \frac{1}{N}\sum_{i:z_i\in V_j} \left[\mathbb{E}_{S_{-V_j}}[\ell(h_k(\cdot;S_{-V_j}),z_i)] - \frac{1}{|V_j|}\sum_{z\in S_{-V_j}} \ell(h_k(\cdot;S_{-V_j}),z_i)\right]. \quad (7.33)$$

Let $h(y) : \mathbb{R}^n \to \mathbb{R}^{\binom{n+d}{d}}$ be given by $h(y)_k = h_k(y)$. Let $B : \mathbb{R}^n \to \mathbb{R}^{\binom{n+d}{d}}$ with columns b_k. Then $(b_k)_k = Y$ so

$$\max_{(b_k)_k : \|b_k\| \le B_M} \sum_{i,j} \left\| \sum_{y_i \in V_j} b_k h_k(y_i) - \hat{A}_{ij} \right\|_F = \max_{Y : \|Y\|_F \le M} \operatorname{Tr}\left(B \sum_{i,j} \sum_{y_i \in V_j} h(y_i) - \hat{A}_{ij} \right) \le M \left\| \sum_{i,j} \sum_{y_i \in V_j} h(y_i) - \hat{A}_{ij} \right\|_F. \quad (7.34)$$

We carefully bound the moments of each of the terms in Equation equation 7.33 in Lemmas 33 and 35, leading to high-probability bounds given in Lemmas 34 and 36, respectively. From this, we obtain the following high probability bound on the generalization gap.

Lemma 32 (Generalization gap) Suppose Q_0 is supported on $B_D(0)$ and C is a complete set of R-warm starts for Q_0 with S_{corr} and $d \ge R(\tilde{\epsilon}/2)^2$. Given N pairs of samples (x_i, y_i) distributed as $y_i = \cdot_i$, for $\cdot_i \sim Q_0$ and $\cdot_i \sim N(0, I)$ with

$$N \ge \frac{n \log(k_{\mathrm{corr}}) \left(1 + M D(2/\tilde{\epsilon})^2\right) \ln(k/\delta) \left(n+d+1\right)^{4(d+1)}}{\epsilon^4},$$

we have with probability at least $1 - \delta$ that

$$\max_{(b_k)_k : \|b_k\| \le B_M} \operatorname{err}_j(\hat{c}) \le \frac{B}{k_{\mathrm{corr}}}.$$

Furthermore, the piecewise polynomial approximation of the score defined in Equation equation 7.29 satisfies

$$\|\hat{s}\|_{L^2(P)} \le \|f\|_{L^2(P)} \le \|\hat{s}\|_{L^2(P)} + B/2.$$

for any other piece-wise polynomial g on the Voronoi cells whose coefficients satisfy $\|g_k^j\|_{(b_k)_k} \le B_M$ for each cell j.

We defer the proof to Section 7.5.3.

Moment and high-probability bounds

For the following lemmas, we assume the following. Q_0 is supported on $B_D(0)$, C is a complete set of R-warm starts for Q_0, and $V_1, \ldots, V_{k_{\mathrm{corr}}}$ is the corresponding Voronoi partition. We are given iid samples $(x_i, y_i), i = 1, \ldots, N$, where $y_i = \cdot_i + \cdot_i$, $\cdot_i \sim Q_0$, and $\cdot_i \sim N(0, I)$. Recall that h_k are the σ-rescaled Hermite polynomials and $h(y) = (h_k(y))_{k \le d}$. We moreover fix the Voronoi cell V_j corresponding to $\hat{A}_j$, and (to simplify notation) suppose that $\hat{A}_j = 0$.

Lemma 33 (Moment bound for first term). We have the following uniform convergence bound for p even:

$$\left\| \sum_{i=1}^{N} 1_{V_j}(x_i) h(y_i) \cdot_i \right\|_{2p} \le O(pn)^{1/2} \left(\frac{n}{d} + 1 \right)^{d/2} p^{p/2} \left(1 + \frac{d^p}{N} \right).$$

We will use properties of sub-exponential random variables; see Appendix H.4 for background.

Proof. In the following, the expectations without index refer to the mixture measure. Let $\hat{\epsilon}_i^{(\cdot)}$ be independent and identically distributed to $\epsilon_i \mid Y = y_i$. Without loss of generality suppose the samples in which h_i fall into V_j are exactly $y_1, \ldots, y_{N_j}$. Using the symmetrization technique with Jensen's inequality:

$$\mathbb{E}\left[\sum_{i=1}^{N_j} h(y_i)\cdot \epsilon_i\right]^{2p} \le \mathbb{E}\left[\sum_{i=1}^{N_j} h(y_i)\cdot(\epsilon_i - \mathbb{E}[\epsilon_i \mid Y = y_i])\right]^{2p} \tag{7.35}$$

$$\le \mathbb{E}\left[\sum_{i=1}^{N_j} h(y_i)\cdot(\epsilon_i - \hat{\epsilon}_i^{(\cdot)})\right]^{2p} \tag{7.36}$$

$$= \mathbb{E}\left[\sum_{i=1}^{N_j} h(y_i)\cdot\mathbb{E}_{\hat{\epsilon}}[\epsilon_i - \hat{\epsilon}_i^{(\cdot)}]\right]^{2p} \tag{7.37}$$

$$\le \mathbb{E}\left[\sum_{i=1}^{N_j} h(y_i)\cdot(\epsilon_i - \hat{\epsilon}_i^{(\cdot)})\right]^{2p}. \tag{7.38}$$

Note that conditioned on any fixed value of y_i, the variable $\epsilon_i - \hat{\epsilon}_i^{(\cdot)}$ is the difference between two iid random variables, each of whose norm is bounded by D. Therefore, given a fixed $\{y_i\}_{i=1}^{N_j}$, the variable $[\sum_{i=1}^{N_j} h(y_i)\cdot(\epsilon_i - \hat{\epsilon}_i^{(\cdot)})]^2$ is a quadratic form of a sub-Gaussian vector. From the properties of sub-Gaussian variables, for every $1 \le k \le n$ and $1 \le d$,

$$\left(\sum_{i=1}^{N_j} h_k(y_i)\cdot(\epsilon_i - \hat{\epsilon}_i^{(\cdot)})\right)^2$$

is sub-exponential with parameter $\left(O(\sqrt{d}\sum_{i=1}^{N_j} h_k(y_i)^2 D^2), O(\sum_{i=1}^{N_j} h_k(y_i)^2 D^2)\right)$. Therefore, noting we do not have independence when we enumerate the variables over k, by Appendix H.4, $\sum_{k=1}^{n}\left[\sum_{i=1}^{N_j} h_k(y_i)\cdot(\epsilon_i - \hat{\epsilon}_i^{(\cdot)})\right]^2$ is sub-exponential with parameters

$$\left(n\sqrt{d}\cdot O\left(\sum_{k=1}^{d}\sum_{i=1}^{N_j} h_k(y_i)^2 D^2\right), n\sqrt{d}\cdot O\left(\max_{k\le d}\sum_{i=1}^{N_j} h_k(y_i)^2 D^2\right)\right).$$

Finally summing over $1 \le k \le n$, from the independence of the coordinates of $\hat{\epsilon}^{(\cdot)}$, by Appendix H.4, the variable

$$X \left[\sum_{i=1}^{N_j} h(y_i)\cdot(\epsilon_i - \hat{\epsilon}_i^{(\cdot)})\right]^2 \sum_{k=1}^{n}\sum_{k=1}^{d}\left[\sum_{i=1}^{N_j} h_k(y_i)\cdot(\epsilon_i - \hat{\epsilon}_i^{(\cdot)})\right]^2$$

is sub-exponential with parameters

$$\hat{v}^2; \left(n\sqrt{d}\cdot O\left(\sum_{k=1}^{d}\sum_{i=1}^{N_j} h_k(y_i)^2 D^2\right), n\sqrt{d}\cdot O\left(\max_{k\le d}\sum_{i=1}^{N_j} h_k(y_i)^2 D^2\right)\right)$$

and has expectation

$$E\left(\sum_{i=1}^{N_j} h(y_i) - \hat{\theta}_i - \theta_i\right)^2 \le \sum_{i=1}^{N_j} n h(y_i)^2 E\left|\theta_i - \hat{\theta}_i\right|^2 \le 4D^2 \sum_{i=1}^{N_j} n h(y_i)^2.$$

But this implies

$$P(X \le E X + Ct) \ge 1 - e^{-t^2};$$

for parameter

$$C = \max\left(v, \sqrt{D^2 n \frac{n}{d} \sum_{i=1}^{N_j} n h(y_i)^2}\right).$$

Thus, using the second property for sub-exponential variables in Proposition 2, for some constant c_1 we have

$$E\left(\sum_{i=1}^{N_j} h(y_i) - \hat{\theta}_i - \theta_i\right)^{2p} \le \left(\sum_{i=1}^{N_j} |y_i - \theta_i|\right)^p \le E X^p + c_1 p D^2 n \frac{n}{d} \sum_{i=1}^{N_j} n h(y_i)^2 \Big)^p. \quad (7.39)$$

Then by Lemma 213,

$$E\left(\sum_{i=1}^{N_j} h(y_i) - \hat{\theta}_i - \theta_i\right)^{2p} E\left(\sum_{i=1}^{N_j} h(y_i) - \hat{\theta}_i - \theta_i\right)^{2p} \left(\sum_{i=1}^{N_j} |y_i - \theta_i|\right)^p$$
$$\le \left(E c_1 p D^2 n \frac{n}{d} \sum_{i=1}^{N_j} n h(y_i)^2\right)^p.$$

Therefore, given $\hat{\theta}_j$, θ_j is the Gaussian around the warm start point $\hat{\theta}_j$, and Lemma 213 applied to S and S (which is a polynomial as p is even) and Lemma 212,

$$\left\| \sum_{i=1}^{N} 1_{V_j}(\hat{\theta}_i - h(y_i) - \theta_i) \right\|_{2p} \le \left(c_1 p D \sqrt{n \frac{n}{d}}\right)^{1/2} \sqrt{\sum_{i=1}^{N_j} n h(y_i)^2}\Bigg\|_p$$
$$\le \left(c_1 p D \sqrt{n \frac{n}{d}}\right)^{1/2} \sqrt{\sum_{i=1}^{N_j} Z n h(y_i)^2 Z_p}$$
$$\le \left(c_1 p D \sqrt{n \frac{n}{d}}\right)^{1/2} \sqrt{\sum_{i=1}^{N_j} e^{2\frac{d-p}{d}R} \cdot Z n h(y_i)^2 Z_{L^p}}.$$
$$\le \left(c_1 p D \sqrt{n \frac{n}{d}}\right)^{1/2} \sqrt{\sum_{i=1}^{N_j} e^{2\frac{d-p}{d}R} \cdot \frac{p}{1-2d} Z n h(y_i)^2 Z_{L^2}}.$$
$$\le c_2 p D n^{1/2} \sqrt{\frac{n}{d}} \left(1 - \frac{d-2}{d}\right) e^{\frac{p}{d}} \sqrt{N};$$

for some universal constant c_2, and we used the upper bound $\|Z\|_k \le e\sqrt{k}$.

□

Lemma 34 (High probability bound for first term). We have

$$P\left(\left|\sum_{i=1}^{N} 1_{V_j}(y_i)\left(h(y_i)-\bar{h}(y_i)\right)\right| \geq t\right) \leq 2\exp\left(-\frac{t}{\bar{N}n^{1/2}}\left(\frac{n}{d}\right)^{1/2}\right).$$

Proof. By Lemma 33 and Markov's inequality,

$$P\left(\left|\sum_{i=1}^{N} 1_{V_j}(y_i)\left(h(y_i)-\bar{h}(y_i)\right)\right| \geq t\right) \leq P\left(\left|\sum_{i=1}^{N} 1_{V_j}(y_i)\left(h(y_i)-\bar{h}(y_i)\right)\right|^{2p} \geq t^{2p}\right)$$
$$\leq \frac{E\left|\sum_{i=1}^{N} 1_{V_j}(y_i)\left(h(y_i)-\bar{h}(y_i)\right)\right|^{2p}}{t^{2p}}$$
$$\leq \left(\frac{Dn^{1/2}\left(\frac{n}{d}\right)^{d/2}(2p)^{d}\bar{N}}{t}\right)^{2p}.$$

Picking

$$p = \left\lfloor \frac{\left(\frac{n}{d}\right)^{d/2} D^{d}\bar{N}n^{1/2}}{\left(\frac{n}{d}\right)^{d/2}} \right\rfloor$$

with appropriate constant, we have

$$P\left(\left|\sum_{i=1}^{N} 1_{V_j}(y_i)\left(h(y_i)-\bar{h}(y_i)\right)\right| \geq t\right) \leq 2\exp\left(-\frac{t}{\bar{N}n^{1/2}}\left(\frac{n}{d}\right)^{1/2}\right).$$

□

Lemma 35 (Moment bound for second term). Assume $C(R) \geq 2$. We have the following uniform convergence bound:

$$E\max_{b_k \in \{0,1\}^d} \left|\sum_{i=1}^{N} 1_{V_j}(y_i)\left(\sum_{k \in \{0,1\}^d} b_k h_k(y_i) - z_i\right)^2 - E\left(\sum_{k \in \{0,1\}^d} b_k h_k(y_i) - z_i\right)^2\right|^p$$
$$\leq O(1)^{d} M^{2p} D^{p} n^{p(1/2)} e^{d}\left(\frac{n}{d}\right)^{d} p^{2p} 4^{2p}\left(\frac{N}{n}\right)^{2d} \bar{N}^{p} D^{2p}.$$

Proof. Without loss of generality, suppose the first N_j samples $y_1, \ldots, y_{N_j}$ are in V_j. Again

using symmetrization for the y_i's,

$$\max_{b_k \cdot \mathbb{S}^{n-1}} \sum_{i=1}^{N_j} \left| b_k \cdot h_k(y_i) - z_i \right|^2 - \left| b_k \cdot h_k(y_i) - z_i \right|^2 \Big| :$$

$$\leq \max_{b_k \cdot \mathbb{S}^{n-1}} \sum_{i=1}^{N_j} \left| b_k \cdot h_k(y_i) - z_i \right|^2 - \left| b_k \cdot h_k(y_i) - z_i \right|^2 \Big|$$

$$\leq \max_{V \in \mathcal{V}_{n,m}} \operatorname{tr}\left(V^T \left(\sum_{i=1}^{N_j} h(y_i) h(y_i)^T - \sum_{i=1}^{N_j} h(y_i) h(y_i)^T \right) \right)$$

$$\leq \max_{V \in \mathcal{V}_{n,m}} \left| \sum_{i=1}^{N_j} z_i \langle V, h(y_i) \rangle - \sum_{i=1}^{N_j} z_i \langle V, h(y_i) \rangle \right| + \sum_{i=1}^{N_j} \left| \langle V, h(y_i) \rangle^2 - \langle V, h(y_i) \rangle^2 \right|$$

$$\leq 2 \max_{V \in \mathcal{V}_{n,m}} \left| \sum_{i=1}^{N_j} \langle V, h(y_i) \rangle - \sum_{i=1}^{N_j} \langle V, h(y_i) \rangle \right| + \sum_{i=1}^{N_j} \left| \langle V, h(y_i) \rangle^2 - \langle V, h(y_i) \rangle^2 \right|$$

$$\leq 2 \max_{V \in \mathcal{V}_{n,m}} \left| \sum_{i=1}^{N_j} z_i \langle V, h(y_i) \rangle - \sum_{i=1}^{N_j} z_i \langle V, h(y_i) \rangle \right| + \sum_{i=1}^{N_j} \left| \langle V, h(y_i) \rangle^2 - \langle V, h(y_i) \rangle^2 \right|$$

$$\leq 2M \left| \sum_{i=1}^{N_j} \langle V, h(y_i) \rangle - \sum_{i=1}^{N_j} \langle V, h(y_i) \rangle \right| + \sum_{i=1}^{N_j} \left| \langle V, h(y_i) \rangle^2 - \langle V, h(y_i) \rangle^2 \right|$$

$$\leq 2M \left| \sum_{i=1}^{N_j} z_i \langle V, h(y_i) \rangle - \sum_{i=1}^{N_j} z_i \langle V, h(y_i) \rangle \right| + \sum_{i=1}^{N_j} \left| \langle V, h(y_i) \rangle^2 - z_i^2 \right| + \left| z_i^2 - \langle V, h(y_i) \rangle^2 \right| :$$

For the third term, note that for each $1 \leq i \leq N$:

$$\left| z_i^2 - \langle V, h(y_i) \rangle^2 \right| \leq |z_i|^2 + |\langle V, h(y_i) \rangle|^2 \leq 2D^2,$$

and this has mean zero. Hence the third term is sub-Gaussian with parameter $\sigma^2 = 4D^4 N$. Therefore

$$\sum_{i=1}^{N_j} \left| z_i^2 - \langle V, h(y_i) \rangle^2 \right| - |z_i|^2 \Big| \leq O\left(D^2 \sqrt{pN} \right). \tag{7.40}$$

For all $1 \leq i \leq N_j$, $1 \leq k \leq n$, and $V \in \mathcal{V}_{n,m}$, given $A_{j,2}$ is the Gaussian around the warm start point A_j: by Cauchy-Schwarz, Lemma 213 applied to h_k^p, Lemma 212, and the assumption that $d \leq 2p$,

$$\left| z_{ik} h_k(y_i) - z_{ik}^2 h_k(y_i)^2 \right| \left| \begin{array}{l} \leq 2 |z_{ik} h_k(y_i)| \\ \leq 2 |z_{ik}| |h_k(y_i)| \\ \leq 2D^2 \sqrt{d} \|h_k(y_i)\|_{L^{2p}} \\ \leq 2D^2 \sqrt{d} \|h_k(y_i)\|_{L^{2p}} \\ \leq 2D^2 \sqrt{d} \cdot 2p \left(1 + d \|h_k(y_i)\|_{L^{2p}} \right) \\ \leq 2D^2 \sqrt{2p} \left(1 + d \right) : \end{array} \right.$$

Therefore, from the Rosenthal inequality [191] with constant given by Pinelis [192], there is a universal constant$c$ such that

$$\mathbb{E}\left|\sum_{i=1}^{N_j} z_{ik}h_k(y_i) - z_{ik}^2 h_k(y_i^2)\right|^p$$
$$\le c^p \max\left\{\sum_{i=1}^{N_j} \mathbb{E}|z_{ik}h_k(y_i) - z_{ik}^2 h_k(y_i^2)|^p, \left(\sum_{i=1}^{N_j} \mathbb{E}|z_{ik}h_k(y_i) - z_{ik}^2 h_k(y_i^2)|^2\right)^{p/2}\right\}$$
$$\le c^p \max\{N_j 2^p \|h_k\|_\infty^p + 1)^p, (N_j \|h_k\|_\infty^2 + 1)^{p/2}\}$$
$$\le (2c)^p (\|h_k\|_\infty^p + 1)^p.$$

which according to the norm property of$p$th norm $\|\cdot\|_p$ implies

$$\left\|\sum_{k=1}^{n}\left|\sum_{i=1}^{N_j} z_{ik}h_k(y_i) - z_{ik}^2 h_k(y_i^2)\right|^2\right\|_p \le \sum_{k=1}^{n}\left\|\sum_{i=1}^{N_j} z_{ik}h_k(y_i) - z_{ik}^2 h_k(y_i^2)\right\|_p^2$$
$$\le n\left(\frac{n}{d}+1\right)^2 (2c)^2 (\|h_k\|_\infty^2 + 1)^2.$$

Therefore, using$\|h_k\|_\infty \le \|e_k\|_\infty$,

$$\left\|\sum_{i=1}^{N_j} z_i h(y_i) - \sum_{i=1}^{N_j} z_i^2 h(y_i^2)\right\|_p \le n^{1/2}\left(\frac{n}{d}+1\right)^2 (2c)^2 (\|e_k\|_\infty^2 + 1). \quad (7.41)$$

For the first term, for all $k_1, k_2 \in [d]$, again by Cauchy-Schwarz, Lemma 213 applied toh_k^p, Lemma 212, and the assumption that$d \ge C(R\sigma)^2$,

$$\|[h_{k_1}(y_i)h_{k_2}(y_i) - h_{k_1}(y_i^2)h_{k_2}(y_i^2)]\|_p \le \|h_{k_1}(y_i)h_{k_2}(y_i)\|_p$$
$$\le \|h_{k_1}(y_i)\|_{2p}\|h_{k_2}(y_i)\|_{2p}$$
$$\le 2^{\sqrt{d/(2p)}}(R\sigma)^2 \|h_{k_1}(y_i)\|_{L^{2p}}\|h_{k_2}(y_i)\|_{L^{2p}}$$
$$\le 2^{\sqrt{d/(2p)}}(R\sigma)^2 (2p)^{2d}\|h_{k_1}(y_i)\|_{L^2}\|h_{k_2}(y_i)\|_{L^2}$$
$$\le (4p)^{2d}.$$

Therefore, again using Rosenthal's inequality,

$$\mathbb{E}\left|\sum_{i=1}^{N_j}(h_{k_1}(y_i)h_{k_2}(y_i) - h_{k_1}(y_i^2)h_{k_2}(y_i^2))\right|^p$$
$$\le c^p \max\left\{\sum_{i=1}^{N_j}\mathbb{E}|h_{k_1}(y_i)h_{k_2}(y_i) - h_{k_1}(y_i^2)h_{k_2}(y_i^2)|^p, \left(\sum_{i=1}^{N_j}\mathbb{E}|h_{k_1}(y_i)h_{k_2}(y_i) - h_{k_1}(y_i^2)h_{k_2}(y_i^2)|^2\right)^{p/2}\right\}$$
$$\le (cp)^4 (2p)^{2d}.$$

which implies

$$\left(\sum_{k_1,k_2\in d}\left(\sum_{i=1}^{N_j}\left(h_{k_1}(y_i)h_{k_2}(y_i)-\left(h_{k_1}(y_i^{(2)})h_{k_2}(y_i^{(2)})\right)\right)\right)^p\right)^{2}$$
$$\le\left(\sum_{k_1,k_2\in d}\left(\sum_{i=1}^{N_j}\left(h_{k_1}(y_i)h_{k_2}(y_i)-\left(h_{k_1}(y_i^{(2)})h_{k_2}(y_i^{(2)})\right)\right)\right)^p\right)^{2}$$
$$\le\left(\frac{n}{d}\right)^{2}\left(2cp(4p)^{2d}\overline{N}\right)^{2}.$$

Therefore, using $\|\cdot\|_k\le\|\cdot\|_{en/k}$,

$$\left(\sum_{i=1}^{N_j}\left(h(y_i)h(y_i)-\left(h(y_i^{(2)})h(y_i^{(2)})\right)\right)\right)^{2}\le\left(\frac{n}{d}\right)^{d}\left(2cp(4p)^{2d}\overline{N}\right)^{2}. \tag{7.42}$$

Combining Equations equation 7.42, equation 7.41, and equation 7.40 completes the proof. □

Lemma 36 (High-probability bound for second term) Assume $C(R-\cdot)\le\cdot^2$. We have

$$P\left(\max_{b:\|b\|_{B_M}\le 1}\sum_{i=1}^{N}\left(\sum_{k\in d}b_k h_k(y_i)-z_i\right)^2-E\left(\sum_{k\in d}b_k h_k(y_i)-z_i\right)^2\ge t\right)$$
$$\le\exp\left(-\left(\frac{n}{d}\right)^{1-2}\min\left(\frac{t}{n^{1-2}M(M+D\sqrt{\overline{N}})^{1-2(d-1)}},\left(\frac{t}{D^2\overline{N}}\right)^{2}\right)\right).$$

Proof. We have

$$P\left(\max_{b:\|b\|_{B_M}\le 1}\sum_{i=1}^{N}\left(\sum_{k\in d}b_k h_k(y_i)-z_i\right)^2-E\left(\sum_{k\in d}b_k h_k(y_i)-z_i\right)^2\ge t\right)$$
$$\le\frac{E\max_{b:\|b\|_{B_M}\le 1}\left|\sum_{i=1}^{N}\left(\sum_{k\in d}b_k h_k(y_i)-z_i\right)^2-E\left(\sum_{k\in d}b_k h_k(y_i)-z_i\right)^2\right|^{2p}}{t^{2p}}$$
$$\le\left(\frac{n^{1-2}M(M+D\sqrt{\overline{N}})+D\sqrt{\overline{N}}(4p)^{2d}\left(\frac{n}{d}\right)^{d}(4p)^{2d}\overline{N}+(4p)^{2d}\sqrt{\overline{pN}}D^2}{t}\right)^{2p}.$$

Picking

$$p=\left(\frac{c't\left(n^{1-2}M(M+D\sqrt{\overline{N}})^{1-(2d-1)}\left(\frac{n}{d}\right)^{1-2}\right)}{\left(\frac{n}{d}\right)^{1-2}},\left(\frac{c't}{D^2\overline{N}}\right)^{2}\right),$$

for universal constant c' small enough, the proof is complete. □

Generalization gap

Proof of Lemma 32. Note that from Equation equation 7.30, for every fixed Voronoi cell V_j:

$$\max_{h_k \in \mathcal{B}_M} \left| \frac{1}{N} \sum_{y_i \in V_j} h(y_i) - \mu_j \right| - \mathbb{E}_{\mathcal{D}} \left| \sum_{y_i \in V_j} h_k(y_i) - \mu_j \right| \leq \max_{h_k \in \mathcal{B}_M} \left| \sum_i \left(\mathbb{1}_{y_i \in V_j} h_k(y_i) - \mu_j \right)^2 - \mathbb{E}_{\mathcal{D}} \left(\mathbb{1}_{y_i \in V_j} h_k(y_i) - \mu_j \right)^2 \right|. \quad (7.43)$$

But from Lemma 34, given

$$N \geq \frac{k^2 \ln(2d+1) + k^2}{4},$$

samples, with probability at least $1 - 2e^{-k^2}$ the absolute value of the first term in Equation equation 7.43 is at most $2\epsilon - \epsilon^2$. Furthermore, based on Lemma 36, given

$$N \geq \frac{k^2 \ln(4d+1) + k^2 M^2 D^4 (2d+1)}{4},$$

with probability at least $1 - 2e^{-k^2}$ the second term is bounded by $2\epsilon - \epsilon^2$. Applying a union bound, the sum of the first and second terms is bounded by ϵ with probability at least $1 - e^{-k^2}$. This shows the first claim. To show the second claim, note that

$$\begin{aligned} L_j(h_k, \mathcal{D}) &= \mathbb{E}_{X,Y} \left(\mathbb{1}_{V_j}(Y) h_k(Y) - Z \right)^2 \\ &= \mathbb{E}_{X,Y} \left(\mathbb{1}_{V_j}(Y) h_k(Y) - Z \right)^2 - \mathbb{E} \left(\mathbb{1}_{V_j}(Y) \right)^2, \end{aligned}$$

where h_k is defined as in equation 7.29 and similarly

$$L_j(g_k, \mathcal{D}) = \mathbb{E}_{X,Y} \left(\mathbb{1}_{V_j}(Y) g_k(Y) - Z \right)^2 - \mathbb{E} \left(\mathbb{1}_{V_j}(Y) \right)^2,$$

where g_k is defined analogously. Therefore

$$L_j(h_k, \mathcal{D}) - L_j(g_k, \mathcal{D}) = \mathbb{E}_{X,Y} \left(\mathbb{1}_{V_j}(Y) h_k(Y) - Z \right)^2 - \mathbb{E}_{X,Y} \left(\mathbb{1}_{V_j}(Y) g_k(Y) - Z \right)^2. \quad (7.44)$$

Summing Equation equation 7.44 for $1 \leq j \leq k^2$ implies

$$\sum_{j=1}^{k^2} L_j(h_k, \mathcal{D}) - L_j(g_k, \mathcal{D}) = \|h_k - Y\|_{L^2(P)}^2 - \|g_k - Y\|_{L^2(P)}^2.$$

But from the previous claim and union bound, we know with probability at least $1 - \epsilon$ each $L_j(h_k, \mathcal{D}) - L_j(g_k, \mathcal{D})$ is bounded by $2\epsilon - \epsilon^2$. This completes the proof. □

7.5.4 Maintaining warm starts

We would like to maintain warm starts to the centers of all Gaussians (of non-negligible mass) as we decrease the noise level. By choosing the highest noise level large enough, 0 will be a warm start. The key observation is that with high probability, the score function points in a direction close to a mean; this remains true with the estimated score when the error is small.

We first show that we do not lose too much in the score estimate if we only consider the means that are close to the mean that a data point came from.

Lemma 37 (Good score estimation $\Rightarrow$ Warm starts). Consider a mixture of Gaussians satisfying Assumption 1 with $\sigma_0^2 \le 1$. There is a universal constant such that the following holds. Let $f(y,t) = \nabla \ln q(y,t)$. Suppose we are given a function g satisfying

$$\mathbb{E}_{y\sim q(\cdot,t)} \|g(y,t) - f(y,t)\|^2 \le \frac{R_0^2}{16} \cdot \varepsilon_{\min}^2. \tag{7.45}$$

Suppose we have $N \ge \frac{\ln(1/\delta)}{\varepsilon_{\min}}$ samples sampled from $q(\cdot,t)$. Let C be the output of ??. Then with probability $1-\delta$, for radius $R = CR_0\sqrt{\ln(c/\varepsilon_{\min})}$ with universal constant C, the support of Q_0 is contained in $\bigcup_{\mu\in C} B(\mu, R)$.

Proof. By equation 7.45 and Chebyshev's inequality,

$$\mathbb{P}_{y\sim q(\cdot,t)}\left[\|g(y,t) - f(y,t)\| \ge 4R_0\right] \le \mathbb{P}_{y\sim q(\cdot,t)}\left[\|g(y,t) - f(y,t)\|^2 \ge 16R_0^2\varepsilon^2\right] \le \frac{\mathbb{E}\|g(y,t) - f(y,t)\|^2}{16R_0^2\varepsilon^2} \le \frac{\varepsilon_{\min}^2}{16}.$$

Consider drawing Q_0, $N(0, I)$, and $Y = \sqrt{1-\bar\alpha_t}\, Q_0 + \sqrt{\bar\alpha_t}\, N(0,I)$. Let $C_o = \{\mu_1, \ldots, \mu_M\}$ be as in Assumption 1. Let $f_{\mathrm{loc}}(y,t) = f_{\mathrm{loc}}^{C, R\sqrt{\ln(k/\varepsilon)}}(y,t)$ be as in Lemma 27, where $R = 3R_0\sqrt{2\ln(k/\varepsilon)}$. Then

$$\mathbb{E}\|f(y,t) - f_{\mathrm{loc}}(y,t)\|^2 \le R_0^2 \cdot 2\ln\left(\frac{k}{\varepsilon}\right) \cdot \varepsilon.$$

By choosing $\varepsilon = \frac{c\,\varepsilon_{\min}}{\ln(c/\varepsilon_{\min})}$ for a small enough constant c and noting $\varepsilon_{\min} \le 1$, we obtain that $\mathbb{E}\|f(y,t) - f_{\mathrm{loc}}(y,t)\|^2 \le R_0^2 \cdot \varepsilon_{\min}^2$. Again by Chebyshev's inequality,

$$\mathbb{P}_{y\sim q(\cdot,t)}\left[\|f(y,t) - f_{\mathrm{loc}}(y,t)\| \ge 4R_0\right] \le \frac{\varepsilon_{\min}}{16}.$$

Hence

$$\mathbb{P}_{y\sim q(\cdot,t)}\left[\|g(y,t) - f_{\mathrm{loc}}(y,t)\| \ge 8R_0\right] \le \frac{\varepsilon_{\min}}{8}. \tag{7.46}$$

Let $V_1, \ldots, V_k$ be the Voronoi partition corresponding to C. Letting i be such that $y \in V_i$, we have that $f_{\mathrm{loc}}(y,t) \in \bar V_i \cap B(\mu_i, R\sqrt{\alpha_t})$. Hence, under the event in equation 7.46, by the triangle inequality,

$$\|g(y,t) - \bar V_i \cap B(\mu_i, R\sqrt{\alpha_t})\| \le 8R_0.$$

By assumption, $Q_0 \in B(\mu_i, R_0) \cap \bar V_i$, so letting $R' = R\sqrt{\alpha_t} + 8R_0$,

$$Q_0 \in B(g(y,t), R') \cap \bar V_i \quad \text{with probability} \ge \frac{7}{8}\varepsilon_{\min}$$

$$Q_0 \in \bigcup_{i=1}^{k} B(g(y,t), R') \cap \bar V_i \quad \text{with probability} \ge \left(1 - \frac{1}{8}\varepsilon_{\min}\right).$$

By the Chernoff bounds, for independent $Z_1, \ldots, Z_N \sim \text{Bernoulli}(p)$, for $c > 0$,

$$\Pr\left[\frac{Z_1 + \cdots + Z_N}{N} < (1-c)p\right] < e^{-\frac{c^2 pN}{2}}$$
$$\Pr\left[\frac{Z_1 + \cdots + Z_N}{N} > (1+c)p\right] < e^{-\frac{c^2 pN}{2+c}}.$$

For fixed c, it suffices to have $N \geq \frac{3\ln(1/\delta)}{p}$ to make this $< \delta$. Applying this to $p = \frac{7}{8}\gamma_{\min}$, $(1-c)p = \frac{3}{4}\gamma_{\min}$ and $p = \frac{1}{8}\gamma_{\min}$, $(1+c)p = \frac{1}{4}\gamma_{\min}$ respectively, by a union bound, given $N \geq \frac{3\ln(2k/\delta)}{\gamma_{\min}}$ iid draws $x_1, \ldots, x_m \sim Q$, we have

$$\Pr\left(\forall i \leq k, \left|\{j : g(y_j) > \beta_{R\alpha\alpha}\gamma_i\}\right| \geq \frac{3}{4}\gamma_{\min} N \text{ and } \left|\{j : g(y_j) > \sum_{i=1}^{k} \beta_{R\alpha\alpha}\gamma_i\}\right| \geq \left(k + 1 + \frac{1}{4}\gamma_{\min}\right)N\right) \geq 1 - \delta.$$

Suppose this event holds. Consider the set $\{y_j : g(y_j) \geq \beta_{R\alpha\alpha R_0} g(y_j)\}$. Choose C such that $R_0 < C < R_0 \sqrt{\ln\left(\frac{1}{1-\gamma_{\min}}\right)}$. For each i, choose j_i such that $g(y_{j_i}) > \beta_{R\alpha\alpha}\gamma_i$. Then $\beta_{R_0}\gamma_i \leq \beta_{R\alpha\alpha R_0} g(y_{j_i})$ so by the second event, these sets cover $1 - \frac{1}{4}\gamma_{\min}$ proportion of the $g(y_j)$, $1 \leq j \leq N$. To finish, apply Lemma 38 to obtain that the output of the algorithm covers $1 - \frac{1}{2}\gamma_{\min}$ proportion of the $g(y_j)$. In light of the first event, for each $1 \leq i \leq k$, it must contain some $g(y_j) > \beta_{R\alpha\alpha}\gamma_i$. This finishes the proof. □

Lemma 38. Let S be a set of subsets of X. Let k be the minimum number of sets in S required to cover $(1-\varepsilon)|X|$ elements of X. Consider the greedy algorithm where at each step, we take the set containing the most uncovered elements, as in [2]. Then the greedy algorithm finds $O(k \ln(1/\varepsilon))$ sets covering $(1-2\varepsilon)|X|$ elements of X.

The proof is based on the classic proof of the approximation ratio for set cover [193].

Proof. Let $S_1, S_2, \ldots$ be the sets chosen by the greedy algorithm, let $U_i = \bigcup_{j=1}^{i} S_j$, and let $x_1, x_2, \ldots$ be the covered elements in order. Define the cost of an element x_j as follows: Let S_i be the first set where x_j appears, and set

$$c_j = |S_i \setminus U_{i-1}|.$$

Now suppose that $U_{i-1} = \{x_1, \ldots, x_{j-1}\}$, and consider the cost of x_j. We claim that

$$c_j \leq \frac{k}{(1-\varepsilon)|X| - (j-1)}.$$

To see this, let $A_1, \ldots, A_k$ be an optimal cover of $(1-\varepsilon)|X|$ elements of X. Then they must cover $(1-\varepsilon)|X| - (j-1)$ elements of $X \setminus U_{i-1}$, so

$$\sum_{\ell=1}^{k} |A_\ell \setminus U_{i-1}| \geq (1-\varepsilon)|X| - (j-1)$$

By optimality of S_i, $|S_i \setminus U_{i-1}| \leq \frac{k}{(1-\varepsilon)|X| - (j-1)}$, which shows the claim. Then the number of sets required to cover the first $(1-2\varepsilon)|X|$ elements is given by the sum of costs of

those elements. It is the ceiling of

$$\sum_{j=1}^{\lceil 1-2\cdot n\rceil} c_j \le \sum_{j=1}^{\lceil 1-2\cdot n\rceil} \frac{k}{\lceil 1-\cdot n\rceil - j + 1} \le k \sum_{j=0}^{\lceil 1-2\cdot n\rceil} c n - j \le O(k \ln(c\cdot\cdot)).$$

□

7.5.5 Proof of Theorem 20 & Corollary 8

Combining all the pieces, we prove Theorem 20 and Corollary 8.

Proof of Theorem 20. Here we show that Algorithm ?? successfully samples from $P_0 \star Q_0 \pm N(0, \cdot^2 I)$ with the claimed time and sample complexity. Let $M_2 \ge \mathbb{E}_{x\sim P_0}\|x\|^2$. Let $\hat{P}_{t_1}$ be the output of Algorithm ??. By the triangle and Pinsker's inequality,

$$TV(P_0, \hat{P}_{t_1}) \le TV(P_0, P_{t_1}) + TV(P_{t_1}, \hat{P}_{t_1}) \le TV(P_0, P_{t_1}) + \sqrt{\tfrac{1}{4} c^2 D_{KL}(P_{t_1} \| \hat{P}_{t_1})}.$$

Choose the starting time $t_1 \le \frac{\cdot^2}{2} \wedge \frac{\cdot^2}{n}$, so that by Lemma 215 we have $TV(P_0, P_{t_1}) \le \frac{\cdot^2}{2}$. Hence it is sufficient to prove $D_{KL}(P_{t_1} \| \hat{P}_{t_1}) \le \frac{\cdot^2}{2}$; actually it suffices to find parameters to make $D_{KL}(P_{t_1} \| \hat{P}_{t_1}) \le O(\cdot^2)$ as the exact constant can be adjusted by rescaling by a constant. We implement Algorithm ?? with the step size schedule obtained recursively by the equality version of condition 3 in Theorem 21, i.e. $t_{k+1} = t_k + \min(\frac{\cdot^2}{M_2 n \ln(T/t_k)}, \frac{\cdot^2}{\ln(T/t_k)})$, with ending time $t_{N_{\text{step}}} = T - \frac{M_2 n}{\cdot^2}$, and number of iterations $N_{\text{step}} = O(\frac{1}{\cdot^2} \ln(\frac{T}{t_1}))$. Then, given that we pick $\cdot^2 \le \frac{\cdot^2}{\ln(T/t_1)}$, Theorem 21 tells us that $D_{KL}(P_{t_1} \| \hat{P}_{t_1}) \le O(\cdot^2)$, as needed.

Hence the problem reduces to giving sufficiently accurate estimates of the score function, to guarantee an accuracy of $\cdot^2 \le \frac{\cdot^2}{\ln(T/t_1)}$ for all time steps $t_1, \ldots, t_{N_{\text{step}}}$. Obtaining such accurate scores is our main contribution and the proof consists of showing by backwards induction on $\ell = N_{\text{step}}, \ldots, 1$ that the following hold:

1. The set C_ℓ is a complete set of R_ℓ-warm starts for Q_0 (see Definition 21), where
$$R_\ell = C \sqrt{R_0^2 + \ell \ln\left(\frac{R_0}{\ell} + 1\right) \frac{k}{\min}}$$
for a large enough universal constant C. Moreover, $|C_\ell| \le \exp(O(\ln(c\cdot) \cdot \cdot))$.
2. The score estimate s_{t_ℓ} is ℓ-accurate in $L^2(P_{t_\ell})$, for $\ell^2 \le \frac{\cdot^2}{\ln(T/t_\ell)}$.

More precisely, we will show that 1 holds for $\ell = N_{\text{step}}, \ldots, N_{\text{step}} - \ell_0$ and 2 holds for $\ell = N_{\text{step}}, \ldots, N_{\text{step}} - \ell_0 + 1$ with probability at least $1 - \frac{\ell_0}{N_{\text{step}}}$.

The base case of the induction ($\ell = N_{\text{step}}, \ell_0 = 0$) is to show that $C_{N_{\text{step}}}(\cdot)$ is a complete set of $R_{N_{\text{step}}}$-warm-starts. But note that the assumption $\cdot^2 \ge \frac{M_2 n}{D^2}$ implies the variance at step N_{step} satisfies $\cdot^2_{N_{\text{step}}} = T - t_0 \le T \le \frac{M_2 n}{\cdot^2} \le D^2$, which given that we pick $C \ge 1$ means

$R_{N_{step}}$ C $_{N_{step}}$ CD. Therefore, from the de nition of D, we have that Q_0 is supported on B_D b $B_{R_{N_{step}}}$.

Next, we show the induction step; here, the hypothesis of induction for stèpis that the set C is a complete set of $R_{`}$-warm starts for $Q_0$. Then, we show that 2 for ` and 1 for ` 1 are satis ed, after excluding an event of probability at most $\frac{}{N_{step}}$. First we handle 2. Using Lemma 30 with

$$Ç^2 \quad \frac{2}{S\%\frac{R_{`}}{}Ž^2 \quad \ln S\frac{k\text{œ}R_{`}\hat{}M_2 \quad n\bullet}{`}\bullet\bullet \ln\hat{}T \quad 1\bullet};$$

there exists a piece-wise polynomial g_C on the Voronoi partition of C that approximates the score function with the desired accuracy,

$$e\text{g}_C \quad f \ {}^2_{`P_t} \ ß Ç^2 \langle R^2 \quad {}^2 \ln \langle \frac{k\text{œ}}{Ç}\bullet\bullet \ ß \frac{2\hat{} \ {}^2_{`} \quad 1\bullet}{\ln\hat{}T \quad 1\bullet}: \tag{7.47}$$

We note

$$\ln\hat{}c\text{G}\bullet \quad O_{„} \ln \frac{\%\frac{R_{`}}{}Ž\hat{}M_2 \quad n\bullet\text{“}}{\bullet} \quad O\hat{}\ln\hat{}c \bullet\bullet$$

assuming that B min ^{TM}c2; $\frac{0}{R_0}$; cD; cn; $_{min}$ ž and noting $_0$ B `, $_{min}$ Bck. The degree of g_C is at most

$$\begin{aligned} d_{`} \quad & O_{–„} \frac{R_{`}}{`} \quad {}^{¾}\!\!/_{4} \frac{}{\ln \langle \frac{k\text{œ}}{Ç}\bullet\text{“}^6 \ln\hat{}c\text{G}\bullet^4\text{—}}_{\bullet} \\ & O_{–„} \frac{R_0}{`} \quad {}^{¾}\!\!/_{4} \frac{}{\ln \langle\langle 1 \quad \frac{R_0}{`}\bullet \frac{k}{min}\bullet} \quad {}^{¾}\!\!/_{4} \frac{}{\ln \langle \frac{1}{–}\bullet\text{“}^6 \ln\hat{}c \bullet^4\text{—}}_{\bullet} \\ & O\text{CE}\text{G}\text{E}\langle \frac{1}{–}\bullet^3 \ \langle \frac{R_0}{0}\bullet^6 \text{‘} \ln \langle \frac{1}{–}\bullet^4 \text{‘}: \end{aligned} \tag{7.48}$$

Now for an arbitrary warm start point $Â_i$ `• > C, by Lemma 30, there exists a polynomial $g_C$ P $_{k\ \text{s}\ d_{`}}$ b_k $h_k\hat{}y$ $Â_i$ `••, where h_k $h_{k;\ ^2}$ are the Hermite polynomials with variance ^{2}, and the coe cients satisfy

$$n\hat{}b_k \quad Ç\hat{}i`\bullet_{k\ \text{s}\ d_{`}} \quad e\text{g}_C \ {}_{Â_i;\ `^2} \ BD:$$

But letting $d_{`}$ be the RHS of equation 7.48 with appropriate constants, the condition $d_{`}$ C$\hat{}R_{`}$ ~ $_{`}$•^{2} is satis ed, and hence we can use Lemma 32 with parameters D, M both set to D; the implication is that given

$$N \quad \text{œ}^4 n \%k\hat{}D \quad 1\bullet^2 \ln\hat{}c \ {}_{min}\bullet Ž^2 \langle n \ln \langle \frac{k\text{º}N_{step}}{}\bullet\bullet^{cd_{`}}$$

samples for $\alpha^2 \frac{2\hat{}\ \backslash\ 1\bullet}{\ln\hat{}T\ \ 1\bullet}$ and some universal constant c, we have the guarantee

$$\left\|\hat{g}_{\mathrm{fit}} - f\right\|^2_{L^{2\hat{}}P_{t\backslash}\bullet} \le \left\|G - f\right\|^2_{L^{2\hat{}}P_{t\backslash}\bullet} + B\,\alpha^2$$

after excluding an event of probability at most $\frac{\delta}{2N_{\mathrm{step}}}$. Combining this with Equation equation 7.47, we obtain

$$\left\|\hat{g}_{\mathrm{fit}} - f\right\|^2_{L^{2\hat{}}P_{t\backslash}\bullet} \le B\,O\hat{}\ \alpha^2\bullet$$

as desired.

Next, we handle 1, i.e. show that C is a complete set of $R_{\backslash\ 1}$-warm starts for Q_0. Note that when $\backslash$ changes by a constant factor, 1 is still satisfied with a modified constant, so we only have to update C each time $t_{\backslash}\ 1$ halves (as done in??). According to Lemma 37, it is sufficient to find an estimate g for the score which satisfies $\left\|g\right\|^2_{L^{2\hat{}}P_{t\backslash}\bullet} \le B\hat{}R_0\ \ \backslash\bullet^2\ _{\min}$. But from our assumption we have $\alpha^2\ \frac{2\hat{}\ \backslash\ 1\bullet}{\ln\hat{}T\ \ 1\bullet}\ B\hat{}R_0\ \ \backslash\bullet^2\ _{\min}$, which means the polynomial G that we obtained for proving part 2 in the induction already satisfies the required accuracy for updating the warm starts as well; hence we can use the same degree of polynomial for our regression here. Again we exclude an event of probability at most $\frac{\delta}{2N_{\mathrm{step}}}$. Note that the sample size required by?? is $O\left(\frac{\ln\hat{}N_{\mathrm{step}} \sim \bullet k}{\ _{\min}}\bullet\right)$, which is negligible compared to the number of samples needed for regression. This completes the induction step.

Multiplying N by the number of steps N_{step} and dropping lower-order terms (recalling the assumption on), we get that the total sample complexity is

$$N^{\mathrm{total}}\ \ \hat{}n\ln\hat{}c\bullet\bullet^{O\hat{}d\backslash\bullet}\ \ \hat{}n\ln\hat{}c\bullet\bullet^{O\ll\ln\%\frac{1}{\delta}\check{Z}^3}\left(\frac{R_0}{\ _0}\right)^{\bullet 6}\bullet\ln\%\frac{1}{\delta}\check{Z}^4\bullet\ :$$

Note that the regression problems that we solve to obtain score estimates in Algorithm?? can still use the same batch of samples, because we apply a union bound. However, we do need to use fresh samples each time for updating the warm starts, separate from the ones that we use for score estimates. This is because our uniform convergence bounds in Lemma 32 require the condition that the samples are independent from the randomness used in the construction of the Voronoi partition. □

Proof of Corollary 8. This follows from Theorem 20 with $_{\min}\ \ \frac{}{C^T}$ and substituting $\min\left(\frac{}{C^T};c2;cR_0;cn;cD\check{z}\right.$ for $\ $ in Theorem 20, noting $\ln\left(\frac{C^l}{\bullet}\right)\ \ O\hat{}l\ \ \ln\hat{}c\bullet\bullet$. □

Chapter 8

The Inductive Bias of Flatness Regularization for Deep Matrix Factorization

Abstract

Recent works on over-parameterized neural networks have shown that the stochasticity in optimizers has the implicit regularization effect of minimizing the sharpness of the loss function (in particular, the trace of its Hessian) over the family zero-loss solutions. More explicit forms of flatness regularization also empirically improve the generalization performance. However, it remains unclear why and when flatness regularization leads to better generalization. This work takes the first step toward understanding the inductive bias of the minimum trace of the Hessian solutions in an important setting: learning deep linear networks from linear measurements, also known as *deep matrix factorization*. We show that for all depth greater than one, with the standard Restricted Isometry Property (RIP) on the measurements, minimizing the trace of Hessian is approximately equivalent to minimizing the Schatten 1-norm of the corresponding end-to-end matrix parameters (i.e., the product of all layer matrices), which in turn leads to better generalization. We empirically verify our theoretical findings on synthetic datasets.

8.1 Introduction

Modern deep neural networks are typically over-parametrized and equipped with huge model capacity, but surprisingly, they generalize well when trained using stochastic gradient descent (SGD) or its variants [194]. A recent line of research suggested the *implicit bias* of SGD as a possible explanation to this mysterious ability. In particular, [11, 195–199] have shown that SGD can implicitly minimize the *sharpness* of the training loss, in particular, the trace of the Hessian of the training loss, to obtain the final model. However, despite the strong empirical evidence on the correlation between various notions of sharpness and generalization [200–203]

and the effectiveness of using sharpness regularization on improving generalization [204, 207], the connection between penalization of the sharpness of training loss and better generalization still remains majorly unclear [208, 209] and has only been proved in the context of two-layer linear models [11, 210, 211]. To further understand this connection beyond the two layer case, we study the inductive bias of penalizing the trace of the Hessian of training loss and its effect on the generalization in an important theoretical deep learning setting: deep linear networks (or equivalently, deep matrix factorization [212]). We start by briefly describing the problem setup.

Deep Matrix Factorization. Consider an L-layer deep network where $L > N$; $L \in 2$ is the depth of the model. Let $W_i > R^{d_i \ d_{i-1}}$ and d_i denote the layer weight matrix and width of the i^{th} ($i > L$) layer respectively. We use W to denote the concatenation of all the parameters $(W_1; : : : ; W_L)$ and define the end-to-end matrix of W as

$$E(W) < W_L W_{L-1} \cdots W_1: \tag{8.1}$$

In this paper, we focus on models that are linear in the space of the end-to-end matrix $E(W)$. Suppose $M^{\ddagger} > R^{d_L \times d_0}$ is the target end-to-end matrix, and we observe n linear measurements (matrices) $A_i > R^{d_L \times d_0}$ and the corresponding labels $y_i = \langle A_i, M^{\ddagger} \rangle$. The training loss of W is the mean-squared error (MSE) between the prediction $\langle A_i, W_L W_{L-1} \cdots W_1 \rangle$ and the observation y_i:

$$L(W) < \frac{1}{n} \sum_{i=1}^{n} (\langle A_i, W_L W_{L-1} \cdots W_1 \rangle - y_i)^2: \tag{8.2}$$

Throughout this paper, we assume that $d_i \in \min(d_0, d_L)$ for each $i > L$ and, thus, the image of the function $E(\cdot)$ is the entire $R^{d_L \times d_0}$. In particular, this ensures that the deep models are sufficiently expressive in the sense that $\min_W L(W) = 0$. For this setting, we aim to understand the structure of the trace of the Hessian minimization, as described below. The trace of Hessian is the sum of the eigenvalues of Hessian, which is an indicator of sharpness and it is known that variants of SGD, such as label noise SGD or 1-SAM, are biased toward models with a smaller trace of Hessian [11, 198].

Min Trace of Hessian Interpolating Solution. Our primary object of study is the interpolating solution with the minimum trace of Hessian, defined as:

$$W^{\ddagger} > \arg\min_{W \ L(W) = 0} \operatorname{tr}(\nabla^2 L(W)): \tag{8.3}$$

As we shall see shortly, the solution to the above optimization problem is not unique. We are interested in understanding the underlying structure of any minimizer $W^{\ddagger}$. This will, in turn, inform us about the generalization nature of these solutions.

8.1.1 Main Results

Before delving into the technical details, we state our main results in this section. This also serves the purpose of highlighting the primary technical contributions of the paper. First,

since the generalization of W only depends on its end-to-end matrix $E(W)$, it is informative to derive the properties of $E(W^\dagger)$ for any min trace of the Hessian interpolating solution $W^\dagger$ defined in equation 8.3. Indeed, penalizing the trace of Hessian in the W space induces an equivalent penalization in the space of the end-to-end parameters. More concretely, given an end-to-end parameter M, let the induced regularizer $F(M)$ denote the minimum trace of Hessian of the training loss at W among all W's that instantiate the end-to-end matrix M i.e., $E(W) = M$.

Definition 23 (Induced Regularizer) Suppose $M \in \mathbb{R}^{d_L \times d_0}$ is an end-to-end parameter that fits the training data perfectly (that is, $\langle A_i, M \rangle = y_i, \forall i \in [n]$). We define the induced regularizer as

$$F(M) = \min_{W: E(W) = M} \operatorname{tr}\left[\nabla^2 L(W)\right] \tag{8.4}$$

Since the image of $E(\cdot)$ is the entire $\mathbb{R}^{d_L \times d_0}$ by our assumption that $d_i \geq \min(d_0, d_L)$, function F is well-defined for all $M \in \mathbb{R}^{d_L \times d_0}$. It is easy to see that minimizing the trace of the Hessian in the original parameter space (see equation 8.3) is equivalent to penalizing $F(M)$ in the end-to-end parameter. Indeed, the minimizers of the implicit regularizer in the end-to-end space are related to the minimizers of the implicit regularizer in the W space, i.e.,

$$\arg\min_{M:\, L(M) = 0} F(M) \quad \text{and} \quad E(W^\dagger) \in \arg\min_{W:\, L(W) = 0} \operatorname{tr}\left[\nabla^2 L(W)\right],$$

where for any $M \in \mathbb{R}^{d_L \times d_0}$, we define $L(M) = \frac{1}{n}\sum_{i=1}^{n} \left(\langle A_i, M \rangle - y_i\right)^2$ and thus $L(W) = L(E(W))$. This directly follows from the definition of F in equation 8.4. Our main result characterizes the induced regularizer $F(M)$ when the data satisfies the RIP property.

Theorem 22 (Induced regularizer under RIP) Suppose the linear measurements $\{A_i\}_{i=1}^{n}$ satisfy the $(1, \delta)$-RIP condition.

1. For any $M \in \mathbb{R}^{d_L \times d_0}$ such that $\langle A_i, M \rangle = y_i, \forall i \in [n]$, it holds that

$$(1-\delta)\, L\left(d_0 d_L\right)^{\frac{1-\gamma}{2}} \|M\|_\dagger^{2(1-\gamma)} \leq F(M) \leq (1+\delta)\, L\left(d_0 d_L\right)^{\frac{1-\gamma}{2}} \|M\|_\dagger^{2(1-\gamma)}. \tag{8.5}$$

2. Let $W^\dagger \in \arg\min_{W: L(W) = 0} \operatorname{tr}\left[\nabla^2 L(W)\right]$ be an interpolating solution with minimal trace of Hessian. Then $E(W^\dagger)$ roughly minimizes the nuclear norm among all interpolating solutions of $L(\cdot)$. That is,

$$\|E(W^\dagger)\|_\dagger \leq \frac{1}{1-\delta} \min_{L(M) = 0} \|M\|_\dagger.$$

However, for more general cases, it is challenging to compute the closed-form expression of F. In this work, we derive closed-form expressions for it in the following two cases: (1) depth L is equal to 2 and (2) there is only one measurement, i.e., $n = 1$ (see Table 8.1). Leveraging the above characterization of induced regularzier, we obtain the following result on the generalization bounds:

Settings	Induced Regularizer $R(W)$	Theorem
$(1, \delta)$-RIP	$\|W\|_F^2 / \sqrt{d_0 d_L}\, \delta \|M\|_*^{2/L}$	Theorem 22
$L \geq 2$	$\left[\frac{1}{n}\sum_{i=1}^n A_i A_i^\top, \frac{1}{n}\sum_{i=1}^n A_i A_i^\top\right]_{2/L}$	Theorem 26 ([211])
$n = 1$	$\|W^\top M^{1/L}\|_{S_{2/L}}$	Theorem 28

Table 8.1: Summary of properties of the induced regularizer in the end-to-end matrix space. Here $\|\cdot\|_{S_p}$ denotes the Schatten-p norm for $p \geq 1$, and Schatten p-quasinorm for $p \in (0, 1)$ (see Definition 24). $\|\cdot\|_*$ denotes the Schatten 1-norm, also known as the nuclear norm.

Theorem 23 (Recovery of the ground truth under RIP). Suppose the linear measurements $\{A_i\}_{i=1}^n$ satisfy the $(2, \delta_n)$-RIP (Definition 25). Then for any $W^* \in \arg\min_{W : \mathcal{L}(W) = 0} \operatorname{tr}(\nabla^2 L(W))$, we have

$$\|\mathcal{E}(W) - \mathcal{E}(W^*)\|_F^2 \leq \frac{8(1+\delta_n)}{(1-\delta_n)^2} \|M - M^*\|_*^2. \tag{8.6}$$

where δ_n depends on the number of measurements n and the distribution of the measurements.

If we further suppose $\{A_i\}_{i=1}^n$ are independently sampled from some distribution over $\mathbb{R}^{d_L \times d_0}$ satisfying that $\mathbb{E}_A[\langle A, M\rangle^2] = \|M\|_F^2$, e.g., the standard multivariate Gaussian distribution, denoted by $\mathcal{G}_{d_L, d_0}$, we know $\delta_n = \mathcal{O}\left(\sqrt{\frac{d_L d_0}{n}}\right)$ from [213] (see Section 8.5.1 for more examples).

Theorem 24. For $n \geq C(d_0 + d_L)$, with probability at least $1 - \exp(-(d_0 + d_L))$ over the randomly sampled $\{A_i\}_{i=1}^n$ from multivariate Gaussian distribution $\mathcal{G}$, for any minimum trace of Hessian interpolating solution $W^* \in \arg\min_{W : \mathcal{L}(W) = 0} \operatorname{tr}(\nabla^2 L(W))$, the population loss $\bar{L}(W) = \mathbb{E}_{A \sim \mathcal{G}}[\langle A, W\rangle - \langle A, M\rangle]^2$ satisfies that

$$\bar{L}(W^*) = \|\mathcal{E}(W^*) - \mathcal{E}(M)\|_F^2 \leq \mathcal{O}\left(\frac{d_0 + d_L}{n}\right) \|M\|_*^2 \log^3 n.$$

Next, we state a lower bound for the conventional estimator for overparameterized models that minimizes the norm. The lower bound states that, to achieve a small error, the number of samples should be as large as the product of the dimensions of the end-to-end matrix $d_0 d_L$ as opposed to $d_0 + d_L$ in case of the min trace of Hessian minimizer. It is proved in Appendix .6.

Theorem 25 (Lower bound for ℓ_2 regression). Suppose $\{A_i\}_{i=1}^n$ are randomly sampled from multivariate Gaussian distribution $\mathcal{G}$, let $W^* = \arg\min_{W : \mathcal{L}(W) = 0} \|\mathcal{E}(W)\|_F$ to be the minimum Frobenius norm interpolating solution, then the expected population loss is

$$\mathbb{E}[\bar{L}(W^*)] \geq \left(1 - \frac{\min\{n, d_0 d_L\}}{d_0 d_L}\right) \|M\|_F^2.$$

The lower bound in Theorem 25 shows in order to obtain an $O(1)$-relatively accurate estimates of the ground truth in expectation, namely to guarantee $\mathbb{E}\|W - W^*\|_F^2 \le O(1)\|W^*\|_F^2$, the minimum Frobenius norm interpolating solution needs at least $\Omega(d_0 d_L)$ samples. In contrast, the minimizer of trace of Hessian in the same problem only requires $O(d_0 + d_L)\|W^*\|_2^2/\|W^*\|_F^2)$ samples, which is at least $O(\min\{d_0, d_L\}/r)$ times smaller. We further illustrate experimentally the superior generalization ability of sharpness minimization algorithms like label noise SGD [11, 195, 214] compared to vanilla mini-bactch SGD Figure 8.1. Due to the space limits, we defer the full setting for experiments into Appendix I.1.

8.2 Related Work

Connection Between Sharpness and Generalization. Research on the connection between generalization and sharpness dates back to [215]. [200] famously observe that when increasing the batch size of SGD, the test error and the sharpness of the learned solution both increase. [201] extend this observation and found that there is a positive correlation between sharpness and the ratio between learning rate and batch size. [203] perform a large-scale empirical study on various notions of generalization measures and show that sharpness-based measures correlate with generalization best. [199] find that among language models with the same validation pretraining loss, those that have smaller sharpness can have better downstream performance. On the other hand, [208] argue that for networks with scaling invariance, there always exist models with good generalization but with arbitrarily large sharpness. We note this does not contradict our main result here, which only asserts the interpolation solution with a minimal trace of Hessian generalizes well, but not vice versa. Empirically, sharpness minimization is also a popular and effective regularization method for overparametrized models [205–207, 216–221].

Implicit Bias of Sharpness Minimization. Recent theoretical works [11, 195, 199, 214] show that SGD with label noise is implicitly biased toward local minimizers with a smaller trace of Hessian under the assumption that the minimizers locally connect as a manifold. Such a manifold setting is empirically verified by [222, 223] in the sense that the set of minimizers of the training loss is path-connected. It is the same situation for the deep matrix factorization problem studied in this paper, although we do not study the optimization trajectory. Instead, we directly study properties of the minimum trace of Hessian interpolation solution.

Sharpness-reduction implicit bias can also happen for deterministic GD. [196] show that normalized GD implicitly penalizes the largest eigenvalue of the Hessian. [224] argues that such sharpness reduction phenomena can also be caused by a multi-scale loss landscape. [197] show that GD with weight decay on a scale-invariant loss function implicitly decreases the spherical sharpness, i.e., the largest eigenvalue of the Hessian evaluated at the normalized parameter. Another line of work focuses on the sharpness minimization effect of a large learning rate in GD, assuming that it converges at the end of training. This has been studied mainly through linear stability analysis [225–228]. Recent theoretical analysis [229, 230] showed that the sharpness minimization effect of a large learning rate in GD does not necessarily rely on convergence and linear stability, through a four-phase characterization of

the dynamics at the Edge of Stability regime [226].

8.3 Preliminaries

Notation. We use $[n]$ to denote $\{1, 2, \ldots, n\}$ for every $n \in \mathbb{N}$. We use $\|M\|_F$, $\|M\|_*$, $\|M\|_2$ and $\mathrm{tr}(M)$ to denote the Frobenius norm, nuclear norm, spectral norm and trace of matrix M respectively. For any function f defined over set S such that $\min_{x \in S} f(x)$ exists, we use $\arg\min_S f$ to denote the set $\{y \in S : f(y) = \min_{x \in S} f(x)\}$. Given a matrix M, we use h_M to denote the linear map $A \mapsto \langle A, M \rangle$. We use $\mathcal{H}_r$ to to denote the set $\mathcal{H}_r \triangleq \{h_M : \|M\|_* \le r\}$. M_i and M_j are used to denote the ith row and jth column of the matrix M.

The following definitions will be important to the technical discussion in the paper.

Rademacher Complexity. Given n data points $\{A_i\}_{i=1}^n$, the empirical Rademacher complexity of function class $\mathcal{H}$ is defined as

$$\mathcal{R}_n(\mathcal{H}) \triangleq \frac{1}{n} \mathbb{E}_{\epsilon_{1:n}} \sup_{h \in \mathcal{H}} \sum_{i=1}^n \epsilon_i h(A_i).$$

Given a distribution P, the population Rademacher complexity is defined as follows: $\overline{\mathcal{R}}_n(\mathcal{H}) \triangleq \mathbb{E}_{A_i \overset{iid}{\sim} P} \mathcal{R}_n(\mathcal{H})$. This is mainly used to upper bound the generalization gap of SGD.

Definition 24 (Schatten p-(quasi)norm). Given any $d, d' \in \mathbb{N}$, $p > 0$, and a matrix $M \in \mathbb{R}^{d \times d'}$ with singular values $\sigma_1(M), \ldots, \sigma_{\min\{d,d'\}}(M)$, we define the Schatten p-(semi)norm as

$$\|M\|_{S_p} \triangleq \left(\sum_{i=1}^{\min\{d,d'\}} \sigma_i^p(M) \right)^{1/p}.$$

Note that in this definition $\|\cdot\|_{S_p}$ is a norm only when $p \ge 1$. When $p \in (0, 1)$, the triangle inequality does not hold. Note that when $p \in (0, 1)$, $\|A + B\|_{S_p} \le 2^{1/p - 1} (\|A\|_{S_p} + \|B\|_{S_p})$ for any matrices A and B, however, $2^{1/p - 1} \le 1$.

We use L to denote the depth of the linear model and $W = (W_1, \ldots, W_L)$ to denote the parameters, where $W_i \in \mathbb{R}^{d_i \times d_{i-1}}$. We assume that $d_i \ge \min\{d_0, d_L\}$ for each $i \in [L-1]$ and, thus, the image of $E(W)$ is the entire $\mathbb{R}^{d_L \times d_0}$. Following is a simple relationship between nuclear norm and Frobenius norm that is used frequently in the paper.

Lemma 39. For any matrices A and B, it holds that $\|AB\|_* \le \|B\|_F \|A\|_F \le \|B\|_F$.

8.4 Exact Formulation of Induced Regularizer by Trace of Hessian

In this section, we derive the exact formulation of trace of Hessian for ℓ_2 loss over deep matrix factorization models with linear measurements as a minimization problem over W. We shall later approximate this formula by a different function in Section 8.5, which allows us to calculate the implicit bias in closed-form in the space of end-to-end matrices.

We first introduce the following simple lemma showing that the trace of the Hessian of the loss is equal to the sum of squares of norms of the gradients of the neural network output.

Lemma 40. For any twice-differentiable function $f(W) : \mathbb{R}^{n} \to \mathbb{R}^{n}$, real-valued labels $y_1, \ldots, y_n$, loss function $L(W) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(W), y_i)$, and any W satisfying $L(W) = 0$, it holds that

$$\mathrm{tr}(\nabla^2 L(W)) = \frac{2}{n}\sum_{i=1}^{n} \|\nabla f(W)\|_F^2.$$

Using Lemma 40, we calculate the trace of Hessian for the particular loss defined in equation 8.2. To do this, we consider W in Lemma 40 to be the concatenation of matrices $W_1, \ldots, W_L$, and we set $f(W)$ to be the linear measurements $\langle A_i, E(W) \rangle$, where $E(W) = W_L \cdots W_1$ (see equation 8.1). To calculate the trace of Hessian, according to Lemma 40, we need to calculate the gradient of $L(W)$ in equation 8.2. To this end, for a fixed i, we compute the gradient of $\langle A_i, E(W) \rangle$ with respect to one of the weight matrices W_j.

$$\begin{aligned}\nabla_{W_j} \langle A_i, E(W) \rangle &= \nabla_{W_j} \mathrm{tr}(A_i^\top W_L \cdots W_1) \\ &= \nabla_{W_j} \mathrm{tr}(W_{j-1} \cdots W_1 A_i^\top W_L \cdots W_{j-1} W_j) \\ &= (W_{j-1} \cdots W_1 A_i^\top W_L \cdots W_{j-1})^\top.\end{aligned}$$

According to Lemma 40, trace of Hessian is given by

$$\mathrm{tr}(\nabla^2 L(W)) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{L} \|\nabla_{W_j} \langle A_i, E(W) \rangle\|_F^2 = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{L} \|W_{j-1} \cdots W_1 A_i^\top W_L \cdots W_{j-1}\|_F^2.$$

As mentioned earlier, our approach is to characterize the minimizer of the trace of Hessian among all interpolating solutions by its induced regularizer in the end-to-end matrix space. The above calculation provides the following more tractable characterization of induced regularizer F in equation 8.12:

$$F(M) = \min_{E(W) = M} \sum_{i=1}^{n}\sum_{j=1}^{L} \|W_{j-1} \cdots W_1 A_i^\top W_L \cdots W_{j-1}\|_F^2. \tag{8.7}$$

In general, we cannot solve F in closed form for general linear measurements $\{A_i\}_{i=1}^{n}$; however, interestingly, we show that it can be solved approximately under reasonable assumption on the measurements. In particular, we show that the induced regularizer, as defined in equation 8.7, will be approximately proportional to a power of the nuclear norm of $E(W)$ given that the measurements $\{A_i\}_{i=1}^{n}$ satisfy a natural norm-preserving property known as the Restricted Isometry Property (RIP) [213, 231].

Before diving into the proof of the general result for RIP, we first illustrate the connection between nuclear norm and the induced regularizer for the depth-two case. In this case, fortunately, we can compute the closed form of the induced regularizer. This result was first proved by [211]. For self-completeness, we also provide a short proof.

Theorem 26 ([211]). For any $M \in \mathbb{R}^{d_L \times d_0}$, it holds that

$$F(M) < \min_{W_2 W_1 = M} \mathrm{tr}(\nabla^2 L(W)) = 2\|\tfrac{1}{n}\textstyle\sum_i A_i A_i^\top\|_*^{1/2} \, \|M\|_*^{1/2}. \tag{8.8}$$

Proof of Theorem 26. We first define $B_1 = [A_1^\top, \dots, A_n^\top]^{\frac{1}{2}}$ and $B_2 = [A_1^\top A_1, \dots, A_n^\top A_n]^{\frac{1}{2}}$. Therefore we have that

$$\operatorname{tr}\left[\nabla^2 L(W)\right] = \sum_{i=1}^{n} \|A_i^\top W_2\|_F^2 \cdot \|W_1 A_i^\top\|_F^2 \le \|B_1 W_2\|_F^2 \cdot \|W_1 B_2\|_F^2 .$$

Further applying Lemma 39, we have that

$$\begin{aligned} F(M) &= \min_{W_2 W_1 = M} \operatorname{tr}\left[\nabla^2 L(W)\right] = \min_{W_2 W_1 = M} \sum_{i=1}^{n} \|A_i^\top W_2\|_F^2 \cdot \|W_1 A_i^\top\|_F^2 \\ &\ge \min_{W_2 W_1 = M} \|B_1 W_2 W_1 B_2\|_*^2 = \|B_1 M B_2\|_*^2 . \end{aligned}$$

Next we show this lower bound of $F(M)$ can be attained. Let $U \Sigma V^\top$ be the SVD of $B_1 M B_2$. The equality condition happens for $W_2^\ddagger = B_1^\dagger U \Sigma^{1/2}, W_1^\ddagger = \Sigma^{1/2} V^\top B_2^\dagger$, where we have that $\sum_{i=1}^{n} \|A_i^\top W_2^\ddagger\|_F^2 \cdot \|W_1^\ddagger A_i^\top\|_F^2 = 2\|\Sigma\|_*^2 = 2\|B_1 M B_2\|_*^2$. This completes the proof. □

The right-hand side in equation 8.8 will be very close to the nuclear norm of M if the two extra multiplicative terms are close to the identity matrix. It turns out that $\{A_i\}_{i=1}^n$ satisfying the $(1, \delta)$-RIP exactly guarantees the two extra terms are $O(\delta)$-close to identity. However, the case for deep networks where depth is larger than two is fundamentally different from the two-layer case, where one can obtain a closed form for F. To the best of our knowledge, it is open whether one obtain a closed form for the induced-regularizer for the trace of Hessian when $L \ge 2$. Nonetheless, in Section 8.5.1, we show that under RIP, we can still approximate it with the nuclear norm.

8.5 Results for Measurements with Restricted Isometry Property (RIP)

In this section, we present our main results for the generalization benefit of flatness regularization in deep linear networks. We structure the analysis as follows:

1. In Section 8.5.1, we first recap some preliminaries on the RIP property.
2. In Section 8.5.2, we prove that the induced regularizer by trace of Hessian is approximately the power of nuclear norm for $(1, \delta)$-RIP measurements (Theorem 22).
3. In Section 8.5.3, we prove that the minimum trace of Hessian interpolating solution with $(2, \delta)$-RIP measurements can recover the ground truth $M^\ddagger$ up to error $\|M^\ddagger\|_*^2$. For $\{A_i\}_{i=1}^n$ sampled from Gaussian distributions, we know $\delta \le \sqrt{\frac{d_0 d_L}{n}}$.
4. In Section 8.5.4, we prove a generalization bound with faster rate of $\frac{d_0 d_L}{n} \|M^\ddagger\|_*^2$ using local Rademacher complexity based techniques from [232].

Next, we discuss important distributions of measurements for which the RIP property holds.

8.5.1 Preliminaries for RIP

De nition 25 (Restricted Isometry Property (RIP)). A family of matrices $\{A_i\}_{i=1}^n$ satis es the (r, δ)-RIP i for any matrix X with the same dimension and rank at most r:

$$(1-\delta)\|X\|_F^2 \le \frac{1}{n}\sum_{i=1}^n \langle A_i, X\rangle^2 \le (1+\delta)\|X\|_F^2. \tag{8.9}$$

Next, we give two examples of distributions where $r(d_0 + d_L)$ samples guarantee $(r, O(1))$-RIP. The proofs follow from Theorem 2.3 in [213].

Example 1. Suppose for every $(j, :, :)$, each entry in the matrix A_i is an independent standard Gaussian random variable (i.e., $A_i \overset{i.i.d.}{\sim} \mathcal{G}_{d_L \times d_0}$. For every constant $\delta \in (0, 1)$, if $n \ge (r(d_0 + d_L))$, then with probability $1 - e^{-n}$, $\{A_i\}_{i=1}^n$ satis es (r, δ)-RIP.

Example 2. If each entry of A_i is from a symmetric Bernoulli random variable with variance 1, i.e. for all (i, k, ℓ), entry $A_{i(k,\ell)}$ is either equal to 1 or -1 with equal probabilities, then for any r and δ, (r, δ)-RIP holds with same probability as in Example 1 if the same condition there is satis ed.

8.5.2 Induced Regularizer of Trace of Hessian is Approximately Nuclear Norm

This section focuses primarily on the proof of Theorem 23. Our proof consists of two steps: (1) we show that the trace of Hessian of training loss at the minimizer W is multiplicatively $(1 \pm \delta)$-close to the regularizer $R(W)$ de ned below (Lemma 41) and (2) we show that the induced regularizer of R, $F_{\mathcal{M}}(\cdot)$, is proportional to $\|M\|_*^{2/L}$ (Lemma 42).

$$R(W) = \|W_L \cdots W_2\|_F^2 d_0 \sum_{j=2}^{L-1} \|W_L \cdots W_{j+1}\|_F^2 \|W_{j-1} \cdots W_1\|_F^2 \|W_{L-1} \cdots W_1\|_F^2 d_L. \tag{8.10}$$

Lemma 41. Suppose the linear measurements $\{A_i\}_{i=1}^n$ satisfy (r, δ)-RIP. Then, for any W such that $L(W) = 0$, it holds that

$$(1-\delta)R(W) \le \mathrm{tr}(\nabla^2 L(W)) \le (1+\delta)R(W).$$

Since $\mathrm{tr}(\nabla^2 L(W))$ closely approximates $R(W)$, we can study R instead of $\mathrm{tr}(\nabla^2 L)$ to understand the implicit bias up to a multiplicative factor $(1 \pm \delta)$. In particular, we want to solve the induced regularizer of $R(W)$ on the space of end-to-end matrices $F_{\mathcal{M}}$:

$$F_{\mathcal{M}}(M) = \min_{W:\, W_L \cdots W_1 = M} R(W). \tag{8.11}$$

Surprisingly, we can solve this problem in closed form.

Lemma 42. For any $M \in \mathbb{R}^{d_L \times d_0}$, it holds that

$$F_{\mathcal{M}}(M) = \min_{W:\, W_L \cdots W_1 = M} R(W) = L(d_0 d_L)^{1-1/L} \|M\|_*^{2/L}. \tag{8.12}$$

Proof of Lemma 42. Applying the L-version of the AM-GM to Equation equation 8.10:

$$\hat{R}\hat{W}\bullet\sim L\bullet^{L}\leq d_0\|W_L\cdots W_2\|_F^2\|W_1\|_F^2\|W_L\cdots W_3\|_F^2\|W_{L-1}\cdots W_1\|_F^2\|W_L\cdots W_{j+1}\|_F^2\|W_j\cdots W_1\|_F^2 d_L: \tag{8.13}$$

$$d_0 d_L\prod_{j=1}^{L-1}\%\|W_L\cdots W_{j+1}\|_F^2\|W_j\cdots W_1\|_F^2\check{Z}$$

Now using Lemma 39, we have for every $1\leq j\leq L-1$:

$$\|W_L\cdots W_{j+1}\|_F^2\|W_j\cdots W_1\|_F^2\leq\|W_L\cdots W_1\|_\ddagger^2+\|M\|_\ddagger^2: \tag{8.14}$$

Multiplying Equation equation 8.14 for all $1\leq j\leq L-1$ and combining with Equation equation 8.13 implies

$$\min_{\sim W\in W_L\times\cdots\times W_1=M\bullet}\hat{R}\hat{W}\bullet\leq CL\hat{d}_0 d_L\bullet^{1\sim L}\|M\|_\ddagger^{2\hat{L}(1\bullet\sim L)}: \tag{8.15}$$

Now we show that equality can indeed be attained. To construct an example in which the equality happens, consider the singular value decomposition of M: $M=U\Sigma V^T$, where Σ is a square matrix with dimension $\operatorname{rank}\hat{M}\bullet$.

For $1\leq i\leq L$, we pick $Q_i\in\mathbb{R}^{d_i\times\operatorname{rank}\hat{M}\bullet}$ to be any matrix with orthonormal columns. Note that $\operatorname{rank}\hat{M}\bullet$ is not larger than d_i for all $1\leq i\leq L$, hence such orthonormal matrices Q_i exist. Then we define the following with $\alpha,\beta>0$ being constants to be determined:

$$\begin{aligned}
W_L&\coloneqq\hat{\alpha}\hat{L}\bullet\sim 2\Sigma^{1\sim 2}Q_{L-1}^T\in\mathbb{R}^{d_L\times d_{L-1}};\\
W_i&\coloneqq Q_iQ_{i-1}^T\in\mathbb{R}^{d_i\times d_{i-1}};\quad\forall 2\leq i\leq L-1;\\
W_1&\coloneqq\alpha^{1-\hat{L}\bullet\sim 2}Q_1^{1\sim 2}V^T\in\mathbb{R}^{d_1\times d_0}.
\end{aligned}$$

Note that Σ is a square matrix with dimension $\operatorname{rank}\hat{M}\bullet$. First of all, note that the defined matrices satisfy

$$W_LW_{L-1}\cdots W_1=\alpha^{L-2\hat{L}\bullet\sim 2}U\Sigma^{1\sim 2}\Sigma^{1\sim 2}V^T=M.$$

To gain some intuition, we check that the equality case for all the inequalities that we applied above. We set the value of α in a way that these equality cases can hold simultaneously. Note that for the matrix holder inequality that we applied in Equation equation 8.14:

$$\|W_L\cdots W_{j+1}\|_F^2\|W_j\cdots W_1\|_F^2=\|W_L\cdots W_1\|_\ddagger^2=\|\Sigma^{1\sim 2}\|_F^2,$$

independent of the choice of α. It remains to check the equality case for the AM-GM inequality that we applied in Equation (8.13). We have for all $2\leq j\leq L-1$:

$$\begin{aligned}
&\|W_L\cdots W_{j+1}\|_F\|W_j\cdots W_1\|_F\\
&\quad j=2\Rightarrow\hat{\alpha}\hat{L}\bullet\sim 2\|\Sigma\|_F\|\Sigma^{1\sim 2}\|_F=\|\Sigma\|_F^{1\sim 2}\|\Sigma^{1\sim 2}\|_F^2;
\end{aligned}\tag{8.16}$$

Hence, equality happens for all of them. Moreover, for cases $j=1$ and $j=L$, we have

$$d_0\|W_L\cdots W_2\|_F=\|\Sigma^{1\sim 2}\|_Fd_0=\alpha^{L-2}\hat{\alpha}\hat{L}\bullet\sim 2\|\Sigma^{1\sim 2}\|_Fd_0=\alpha^{\hat{L}\bullet\sim 2}: \tag{8.17}$$

$$d_L\|W_{L-1}\cdots W_1\|_F=\|\Sigma^{1\sim 2}\|_Fd_L=\alpha^{1-L-2}\hat{\alpha}\hat{L}\bullet\sim 2\|\Sigma^{1\sim 2}\|_Fd_L=\alpha^{1-\hat{L}\bullet\sim 2}: \tag{8.18}$$

Thus it suffices to set $\alpha=\hat{\frac{d_L}{d_0}}\bullet^{1\sim 2}$ and $\beta=\hat{\frac{\|\Sigma\|_F^{1\sim 2}\|\Sigma\|_F}{d_0d_L}}\bullet^{2\sim L}=\hat{\frac{\|M\|_\ddagger}{d_0d_L}}\bullet^{1\sim L}$ so that the left-hand sides of equation 8.16, equation 8.17, and equation 8.18 are equal, which implies that the lower bound in Equation equation 8.15 is actually an equality. The proof is complete. □

Now we can prove Theorem 22 as an implication of Lemma 42.

Proof of Theorem 22. The first claim is a corollary of Lemma 41. We note that

$$\begin{aligned} F(M) &\le \min_{W_1,\dots,W_L} \|M\|_1 \quad \text{s.t.} \quad \|\mathcal{A}(W_1,\dots,W_L) - y\|_2 \le \epsilon \\ F(M) &\le \min_{W_1,\dots,W_L} \|M\|_1 \quad \text{s.t.} \quad \|\mathcal{A}(W_1,\dots,W_L) - y\|_2 \le \epsilon. \end{aligned}$$

For the second claim, pick W that minimizes $R(W)$ over all W's that satisfy the linear measurements, thus we have that

$$R(W) \le (\sqrt{2}(1+\delta_L)d_0 d_L)^{L} \|E(W) - Y\|_2^{\sqrt{2}(1+\delta_L)} \le (\sqrt{2}(1+\delta_L)d_0 d_L)^{L} \min_{L(M)=0} \|M - Y\|_2^{\sqrt{2}(1+\delta_L)}. \tag{8.19}$$

Now from the definition of $E(W^\ddagger)$,

$$\operatorname{tr}(\mathcal{C}^2(L(W^\ddagger)) \le \operatorname{tr}(\mathcal{C}^2(L(W)) \le \|\mathcal{C}(L(W))\|_1 \le R(W), \tag{8.20}$$

where the last inequality follows from the definition of W. On the other hand

$$\operatorname{tr}(\mathcal{C}^2(L(W^\ddagger)) \ge \|\mathcal{C}(L(W^\ddagger))\|_1 \ge R(W) \ge (\sqrt{2}(1+\delta_L)d_0 d_L)^{L} \|E(W^\ddagger) - Y\|_2^{\sqrt{2}(1+\delta_L)}. \tag{8.21}$$

Combining equation 8.19, equation G.29 and equation G.30,

$$\|E(W^\ddagger) - Y\|_2 \le \left(\frac{1}{1 - \frac{L}{\sqrt{2}(1+\delta_L)}}\right) \min_{L(M)=0} \|M - Y\|_2.$$

The proof is completed by noting that $\frac{L}{\sqrt{2}(1+\delta_L)} \le 1$ for all $L \ge 2$. □

Thus combining Example 1 and Theorem 22 with $\epsilon = 1/2$, we have the following corollary.

Corollary 9. Let $\{A_i\}_{i=1}^n$ be sampled independently from Gaussian distribution $\mathcal{G}(d_L, d_0)$ where $n \ge C(\delta_L d_0 d_L)^2$, with probability at least $1 - \exp(-\delta n)$, we have

$$\|E(W^\ddagger) - Y\|_2 \le 3 \min_{L(M)=0} \|M - Y\|_2 \le 3\|E(W^\ddagger) - Y\|_2.$$

8.5.3 Recovering the Ground truth

In this section, we prove Theorem 23. The idea is to show that under RIP, the empirical loss $L(W)$ is a good approximation for the Frobenius distance of $E(W)$ to the ground truth $M^\ddagger$. To this end, we first introduce a very useful Lemma 43 below, whose proof is deferred to Appendix .5.

Lemma 43. Suppose the measurements $\{A_i\}_{i=1}^n$ satisfy the $(2, \delta)$-RIP condition. Then for any matrix $M \in \mathbb{R}^{d_L \times d_0}$, we have that

$$\left|\frac{1}{n}\|\mathcal{A}(M)\|_2^2 - \|M\|_F^2\right| \le 2\delta\|M\|_2^2.$$

We note that if $\{A_i\}_{i=1}^n$ are i.i.d. random matrices with each coordinate being independent, zero mean, and unit variance (like standard Gaussian distribution), then $\|W - M^*\|_F^2$ is the population squared loss corresponding to W. Thus, Theorem 23 implies a generalization bound for this case. Now we are ready to prove Theorem 23.

Proof of Theorem 23. Note that from Theorem 22,

$$\|E[W^{(1)}]\|_2 \le \frac{1}{1-\epsilon} \min_{L \in \mathbb{R}^{d_1 \times 0}} \|M_1\|_2 \le \frac{1}{1-\epsilon}\|M^*\|_2,$$

which implies the following by triangle inequality,

$$\|E[W^{(1)}] - M^*\|_2 \le \|E[W^{(1)}]\|_2 + \|M^*\|_2 \le \frac{2}{1-\epsilon}\|M^*\|_2. \tag{8.22}$$

Combining equation 8.22 with Lemma 43 (with $M = E[W^{(1)}] - M^*$):

$$\frac{1}{n}\sum_{i=1}^n \langle A_i, E[W^{(1)}] - M^* \rangle^2 \le \|E[W^{(1)}] - M^*\|_F^2 \le \frac{8}{(1-\epsilon)^2}\|M^*\|_2^2.$$

Since W^* satisfies the linear constraints $\mathrm{tr}[A_i E[W^{(1)}]] = b_i$, $\frac{1}{n}\sum_{i=1}^n \langle A_i, E[W^{(1)}] - M^* \rangle^2 \le \frac{1}{n}\sum_{i=1}^n \xi_i \langle A_i, E[W^{(1)}] - M^* \rangle = \frac{1}{n}\sum_{i=1}^n \xi_i (b_i - \langle A_i, M^* \rangle) = 0$, which completes the proof. □

8.5.4 Generalization Bound

In this section, we prove the generalization bound in Theorem 24, which yields a faster rate of $O\left(\frac{d_0 d_L}{n}\|M^*\|_2^2\right)$ compared to $O\left(\sqrt{\frac{d_0 d_L}{n}}\|M^*\|_2^2\right)$ in Theorem 23. The intuition for this is as follows: By Corollary 9, we know that with very high probability, the learned solution has a bounded nuclear norm for its end-to-end matrix, no larger than $3\|M^*\|_2$, where M^* is the ground truth. The key mathematical tool is Theorem 27, which provides an upper bound on the population error of the learned interpolation solution that is proportional to the square of the Rademacher complexity of the function class $\mathcal{H}_{3\|M^*\|_2} = \{h_M : \|M\|_2 \le 3\|M^*\|_2\}$. Theorem 27 (Theorem 1, [232]) Let $\mathcal{H}$ be a class of real-valued functions and $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ be a differentiable non-negative loss function satisfying that (1) for any x and y, $|\ell'(x, y)|$ with respect to its first coordinate is H-Lipschitz and (2) $\sup_{x,y} \ell(x, y) \le B$, where H, B are some positive constants. Then for any $\delta > 0$, we have that with probability at least $1 - \delta$ over a random sample of size n, for any $h \in \mathcal{H}$ with zero training loss,

$$L(h) \le O\left(H \log^3(n) R_n^2(\mathcal{H}) + \frac{B \log(1/\delta)}{n}\right). \tag{8.23}$$

One technical difficulty is that Theorem 27 only works for bounded loss functions, but the ℓ_2 loss on Gaussian data is unbounded. To circumvent this issue, we construct a smoothly truncated variant of ℓ_2 loss equation 41 and apply Theorem 27 on that. Finally, we show that with a carefully chosen threshold, this truncation happens very rarely and, thus, does not change the population loss significantly. The proof can be found in Appendix .5.

Figure 8.1: Train and test loss. Label noise SGD leads to better generalization results due to the sharpness-minimization implicit biases (as shown in Figure I.1), while mini-batch SGD without label noise finds solutions with much larger test loss.

8.6 Result for the Single Measurement Case

Quite surprisingly, even though in the general case we cannot compute the closed-form of the induced regularizer in equation 8.12, we can find its minimum as a quasinorm function of the $E\hat{}W \bullet$ which only depends on the singular values of $E\hat{}W \bullet$. This yields the following result for multiple layers L (possibly $L A 2$) with a single measurement.

Theorem 28. Suppose there is only a single measurement matrix A, i.e., $n 1$. For any $M > R^{d_L d_0}$, the following holds:

$$F\hat{}M \bullet \min_{W_L :::W_1 M} \operatorname{tr} \left(\hat{}2\right) L\hat{}W \bullet L \backslash \check{S} A^T M \bullet^{L 1} A^T \backslash^{2\sim L}_{S_{2\sim L}} : \tag{8.24}$$

To better illustrate the behavior of this induced regularizer, consider the case where the measurement matrix A is identity and M is symmetric with eigenvalues $\tilde{} _i \bullet^{d}_{i 1}$. Then, it is easy to see that $F\hat{}M \bullet$ in equation 8.24 is equal to $F\hat{}M \bullet P_i {}^{2\hat{}L 1\bullet\sim L}_i$. Interestingly, we see that the value of $F\hat{}M \bullet$ converges to the Frobenius norm of M and not the nuclear norm as L becomes large, which behaves quite differently (e.g. in the context of sparse recovery). This means that beyond RIP, the induced regularizer can behave very differently, and perhaps the success of training deep networks with SGD is closely tied to the properties of the dataset.

8.7 Conclusion and Future Directions

In this paper, we study the inductive bias of the minimum trace of the Hessian solutions for learning deep linear networks from linear measurements. We show that trace of Hessian regularization of loss on the end-to-end matrix of deep linear networks roughly corresponds to nuclear norm regularization under restricted isometry property (RIP) and yields a way to

recover the ground truth matrix. Furthermore, leveraging this connection with the nuclear norm regularization, we show a generalization bound which yields a faster rate than Frobenius (or ℓ_2 norm) regularizer for Gaussian distributions. Finally, going beyond RIP conditions, we obtain closed-form solutions for the case of a single measurement. Several avenues for future work remain open, e.g., more general characterization of trace of Hessian regularization beyond RIP settings and understanding it for neural networks with non-linear activations.

Acknowledgement

TM and ZL would like to thank the support from NSF IIS 2045685. KG and SJ acknowledge support by NSF award CCF-2112665 (TILOS AI Institute) and NSF award 2134108.

.1 Experiments

In this section, we examine our theoretical results with controlled experiments via synthetic data. The experiments are based on mini-batch SGD and label noise SGD [214]. Both use the standard update rule $W_{t+1} = W_t - \eta \nabla L_t(W_t)$, but with different objectives:

- Mini-batch loss: $L_t^{\text{mini-batch}}(W) = \frac{1}{B}\sum_{i\in B_t}(f_i(W) - y_i)^2$;
- Label-noise loss: $L_t^{\text{label-noise}}(W) = \frac{1}{B}\sum_{i\in B_t}(f_i(W) - y_i - \sigma_{t,i})^2$,

where B_t is the batch of size B independently sampled with replacement at step t and $\sigma_t \in \mathbb{R}^d$ are i.i.d. multivariate zero-mean Gaussian random variables with unit variance.

It is known that with a small learning rate, label noise SGD implicitly minimizes the trace of Hessian of the loss, after reaching zero loss [11, 195]. In particular, [11] show that after reaching zero loss, in the limit of step size going to zero, label noise SGD converges to a gradient flow according to the negative gradient of the trace of Hessian of the loss. As a result, we expect label noise SGD to be biased to regions with smaller trace of Hessian. We also compare the label noise SGD with vanilla SGD without label noise as a baseline, which can potentially find a solution with large sharpness when the learning rate is small. Note this is not contradictory with the common belief that mini-batch SGD prefers flat minimizers and thus benefits generalization [200, 201]. For example, assuming the convergence of mini-batch SGD, [225] shows that the solution found by SGD must have a small sharpness, bounded by a certain function of the learning rate. However, there is no guarantee when the learning rate is small and the upper bound of sharpness becomes vacuous.

In our synthetic experiments, we sample $n = 600$ input matrices $\{A_i\}_{i=1}^n$, where $A_i \in \mathbb{R}^{d\times d}$ with $d = 60$. Each entry $A_i^{(j,k)}$ is i.i.d. sampled from normal distribution $\mathcal{N}(0,1)$. The ground truth matrix M^* is constructed by $M^* = M_1 M_2^\top$, where $M_1 \in \mathbb{R}^{d\times r}$ and $M_2 \in \mathbb{R}^{r\times d}$ and r is the rank of M^*. The entries in M_1 and M_2 are again i.i.d. sampled from $\mathcal{N}(0,1)$ and the rank r is set to 3. The corresponding label is therefore computed via $\langle A_i, M^*\rangle$. The parameters $\{W_1, \ldots, W_L\}$ are sampled from a zero-mean normal distribution for depth $L = 2, 3, 5$, and 10. For label noise SGD, we optimize the parameter via SGD with label noise drawn from

Figure 2: Trace of Hessian and Nuclear Norm. Label noise SGD recovers the min nuclear norm solution via its sharpness-minimization implicit regularization and thus leads to better generalization (see Figure 8.1).

$N\hat{}0;1\bullet$ and batch size50. The learning rate is set to0:01.

We examine our theory by plotting the training and testing loss along with the nuclear norm and the trace of Hessian of the label noise SGD solutions in Figures 8.1 and I.1. As the gure illustrates, the trace of the Hessian exhibits a gradual decrement, eventually reaching a state of convergence over the course of the training process. This phenomenon co-occurs with the decreasing of the nuclear norm of the end-to-end matrix. In particular, we further plot the nuclear norm of the min nuclear norm solution obtained via solving convex optimization in Figure I.1 and demonstrate that label noise SGD converges to the minimal nuclear norm solution, as predicted by our theorem Theorem 22. As a consequence of this sharpness-minimization implicit bias, the test loss decreases drastically.

Interestingly, there are a few large spikes in the training loss curve of mini-batch SGD without label noise even after the training loss becomes as small as 10^{-12} and its generalization improves immediately after recovering from the spike. Meanwhile, the trace of hessian and the nuclear decrease during this process. We do not have a complete explanation for such spikes. One possible explanation from the literature [233] is that the loss landscape around the minimizers is too sharp and thus mini-batch SGD is not linear stable around the minimizer, so it escapes eventually. However, this explanation does not explain why minibatch SGD can nd a atter minimizer each time after escaping and re-converging.

.2 Additional Related Work

Implicit Bias of Gradient Descent on Matrix Factorization. At rst glance, over-tting could happen when the number of linear measurements is less than the size of the groundtruth matrix. Surprisingly, a recent line of works [212, 234 240] has shown that GD

starting from small initialization has a good implicit bias towards solutions with approximate recovery of ground truth. Notably, [234] show that for depth 2, GD from in nitesimal initialization is implicitly biased to the minimum nuclear norm solution under commuting measurements and [212] generalize this results to deep matrix factorization for any depth. This is very similar to our main result that for all depth (C2) the implicit regularization is minimizing nuclear norm, though the settings are di erent. Moreover, when the measurements satisfy RIP, [241, 242] show that GD exactly recovers the ground truth.

Provable Generalization of Flatness Regularization for Two-layer Models. To our best knowledge, most existing generalization analysis for at regularization are for two-layer models,e.g., [11] shows that the min trace of hessian interpolating solution of 2-layer diagonal linear networks can recover sparse ground truth on gaussian or boolean data, and [210] proves a generalization bound for the interpolating solutions with the smallest maximum eigenvalue of Hessian for non-centered data. [211] is probably the most related work to ours, which shows that the trace of Hessian implicit bias for two-layer matrix factorization is a rescaled version of the nuclear norm of the end-to-end matrix. Using this formula, they further prove that the attest solution in this problem recovers the low-rank ground truth. However, matrix factorization with more than two layers is fundamentally more challenging compared to the depth two case; while we managed to obtain a formula for the trace of Hessian for deeper networks given a single measurement (see Theorem 28), as far as we know, one in general cannot obtain a closed-form solution for the trace of Hessian regularizer as a function of the end-to-end matrix for multiple measurements. In this work, we discover a way to bypass this hardness by showing that minimizing the trace of Hessian regularizer for a xed end-to-end matrix approximately amounts to the nuclear norm of the end-to-end matrix, when the linear measurements satisfy the RIP property. As a cost of this approximation, we are not able to show the exact recovery of the low-rank ground truth, but only up to a certain precision.

Sharpness Minimization in Deep Diagonal Linear Network. [211] show that the minimizer of trace of Hessianin a deep diagonal matrix factorization model with Gaussian linear measurements becomes the Schatten 2-L norm of a rescaled version of the end to end matrix. At rst glance, their result might seem contradictory to our result in the RIP setup, as their implicit regularization is not always the Nuclear norm the sparsity regularization vanishes when $L = a$. Similar results have been obtained by [210] for minimizing a di erent notion of sharpness among all interpolating solutions, the largest eigenvalue of Hessian, on the same diagonal linear models. The subtle di erence is that since we consider the more standard setting without assuming the weight matrices are all diagonal, then in the calculation of the trace of Hessian of the loss we need to also di erentiate the loss with respect to the non-diagonal entries, even though their values are zero, which is quite di erent from ℓ_p norm regularization. This curiously shows the complicated interplay between the geometry of the loss landscape and the implicit bias of the algorithm.

Sharpness-related Generalization Bounds. Most existing sharpness-related generalizations depend on not only the sharpness of the training loss but also other complexity measures like a norm of the parameters or even undesirable dependence on the number of

parameters [207, 216, 243 245]. In contrast, our result only involves the trace of Hessian but not parameter norm or the number of parameters, e.g., our result holds for any (large) width of intermediate layers, $d_1, \ldots, d_{L-1}$.

.3 Proof of Lemma 41

Proof. For a fixed $j \in \{2, \ldots, L-1\}$ and vectors $x \in \mathbb{R}^{d_0}$ and $y \in \mathbb{R}^{d_L}$ we apply the RIP property in Definition 25 for the rank one matrix $X = xy^T$. As a result we get

$$\|A(xy^T)\|_2^2 - \frac{1}{n}\sum_{i=1}^{n} \langle A_i, xy^T \rangle^2 \le \|A(xy^T)\|_2^2,$$

or equivalently

$$\|A(xx^T)\|_2^2 \|yy^T\|_2^2 - \frac{1}{n}\sum_{i=1}^{n} (x^T A_i y)^2 \le \|A(xx^T)\|_2^2 \|yy^T\|_2^2. \tag{25}$$

Now for arbitrary indices $1 \le \ell \le d_{j-1}$ and $1 \le k \le d_j$, we pick x, y in Equation equation 25 equal to the ℓth row of the matrix $W_{j-1} \cdots W_1$ and the kth column of the matrix $W_L \cdots W_{j-1}$:

$$\begin{aligned} &\|A(xx^T)\|_2^2 \|yy^T\|_2^2 \\ &- \frac{1}{n}\sum_{i=1}^{n} \langle A_i, xx^T \rangle (W_{j-1} \cdots W_1 e_\ell)^2 (e_k^T W_L \cdots W_{j-1})^2 \\ &\le \|A(xx^T)\|_2^2 \|yy^T\|_2^2. \end{aligned} \tag{26}$$

Summing this over all ℓ, k, we obtain that the sum of Frobenius norm of matrices $W_{j-1} \cdots W_1 A_i W_L \cdots W_{j-1}$ concentrate around $\|W_{j-1} \cdots W_1\|_F^2 \|W_L \cdots W_{j-1}\|_F^2$.

$$\begin{aligned} &\|W_{j-1} \cdots W_1\|_F^2 \|W_L \cdots W_{j-1}\|_F^2 \\ &- \frac{1}{n}\sum_{i=1}^{n} \|W_{j-1} \cdots W_1 A_i W_L \cdots W_{j-1}\|_F^2 \\ &\le \|W_{j-1} \cdots W_1\|_F^2 \|W_L \cdots W_{j-1}\|_F^2. \end{aligned} \tag{27}$$

For $j = 1$, we apply Equation equation 25 with $x = (W_{j-1} \cdots W_1 e_\ell)$ and $y = e_k$, where e_k is the kth standard vector:

$$\|W_{L-1} \cdots W_1 e_\ell\|_2^2 - \frac{1}{n}\sum_{i=1}^{n} (W_{L-1} \cdots W_1 e_\ell)^T A_i e_k)^2 \le \|W_{L-1} \cdots W_1 e_\ell\|_2^2.$$

Summing this for all k, ℓ

$$d_0 \|W_{L-1} \cdots W_1\|_F^2 - \frac{1}{n}\sum_{i=1}^{n} \|W_{L-1} \cdots W_1 A_i\|_F^2 \le d_0 \|W_{L-1} \cdots W_1\|_F^2. \tag{28}$$

Similarly for $j = L$

$$d_L \|W_{L-1} \cdots W_2\|_F^2 - \frac{1}{n}\sum_{i=1}^{n} \|A_i W_{L-1} \cdots W_2\|_F^2 \le d_L \|W_{L-1} \cdots W_2\|_F^2. \tag{29}$$

Combining Equations equation 27, equation 28, and equation 29

$$\|R(W)\|^2 - \mathrm{tr}(\Phi^2 L(W)) \le \|R(W)\|^2.$$

□

.4 Proof of Theorem 28

Proof of Theorem 28. Recall that we hope to characterize the solution with a minimal trace of hessian given that the end-to-end matrix $E(W) = W_L \cdots W_1$ is equal to some fixed matrix M, namely,

$$\min_{E(W)=M} \sum_{i=1}^{L} \|W_{i-1} \cdots W_1 A^T W_L \cdots W_{i-1}\|_F^2 .$$

Let W be any minimizer of the above objective. For arbitrary matrix $C \in \mathbb{R}^{d_i \times d_i}$, define

$$U(t) = \exp(tC) = \sum_{i=0}^{a} \frac{(tC)^i}{i!};$$

For any i, we multiply W_i from left by $U(t)$ and multiply W_{i-1} by $U(t)^{-1}$ from right,

$$W_i(t) = U(t) W_i,$$
$$W_{i-1}(t) = W_{i-1} U(t)^{-1}.$$

For convenience, below we drop the dependence of $W_i(t), W_{i-1}(t)$ over t, that is, only W_i and W_{i-1} are implicitly functions of t, while the rest W_j are independent of t. Then, note that for any $j < i-1$ we have

$$W_{j-1} \cdots W_1 A^T W_L \cdots W_{i-1} U(t)^{-1} U(t) W_i \cdots W_{j-1} = W_{j-1} \cdots W_1 A^T W_L \cdots W_{j-1},$$

and for $j > i-2$:

$$W_{j-1} \cdots W_{i-1} U(t)^{-1} U(t) W_i \cdots W_1 A^T W_L \cdots W_{j-1} = W_{j-1} \cdots W_{i-1} W_i \cdots W_1 A^T W_L \cdots W_{j-1}.$$

So the only terms that actually change as a function of t correspond to $j = i$,

$$\|W_{i-1} \cdots W_1 A^T W_L \cdots W_{i-1}\|_F^2 = \mathrm{tr}(W_{i-1} \cdots W_1 A^T W_L \cdots W_{i-1} W_{i-1}^T \cdots W_L^T A W_1^T \cdots W_{i-1}^T), \tag{30}$$

and to $j = i-1$,

$$\|W_i \cdots W_1 A^T W_L \cdots W_{i-2}\|_F^2 = \mathrm{tr}(W_i \cdots W_1 A^T W_L \cdots W_{i-2} W_{i-2}^T \cdots W_L^T A W_1^T \cdots W_i^T). \tag{31}$$

Now taking derivative of $U(t)$ with respect to t,

$$U'(0) = C.$$

Now for every $j \in \{1, \ldots, L\}$ we define

$$\tilde{W}_j = W_{j-1} \cdots W_1 A^T W_L \cdots W_{j-1},$$

where we use $W_{i-1} \cdots W_1$ and $W_L \cdots W_{i-1}$ to denote identity for $i = 1$ and $i = L$ respectively.

Then, if we take derivative from the terms equation 30 and equation 31 with respect to

$$\frac{d}{dt}\|W_{i-1}\cdots W_1 A^T W_L\cdots W_{i-1}\|_F^2\Big|_{t=0}$$
$$\operatorname{tr}\big(C\,C^T\cdot W_{i-1}^T\cdots W_L^T A W_1^T\cdots W_{i-1}^T W_{i-1}\cdots W_1 A^T W_L\cdots W_{i-1}\big);\qquad (32)$$
$$\operatorname{tr}\big(C\,C^T\cdot \tilde{W}_i^T\tilde{W}_i\big):$$

and

$$\frac{d}{dt}\|W_i\cdots W_1 A^T W_L\cdots W_{i-2}\|_F^2\Big|_{t=0}$$
$$\operatorname{tr}\big(C\,C^T\cdot \tilde{W}_{i-1}\tilde{W}_{i-1}^T\big)\qquad (33)$$

Now from the optimality of W, the following equality holds for every matrix $C\in\mathbb{R}^{d\times d}$:

$$\frac{d}{dt}\operatorname{tr}\big(\nabla^2 L(W(t))\big)\Big|_{t=0}=\operatorname{tr}\big(C\,C^T\cdot(\tilde{W}_i^T\tilde{W}_i+\tilde{W}_{i-1}\tilde{W}_{i-1}^T)\big)=0.\qquad (34)$$

Now since C is arbitrary and the matrices $\tilde{W}_i^T\tilde{W}_i$ and $\tilde{W}_{i-1}\tilde{W}_{i-1}^T$ are symmetric, we must have

$$\tilde{W}_i^T\tilde{W}_i=-\tilde{W}_{i-1}\tilde{W}_{i-1}^T.\qquad (35)$$

Equation (35) implies that all $\tilde{W}_i$ for $1\le i\le L$ have the same set of singular values. Moreover, there exists matrices $\{U_i\}_{i=0}^{L}$ where the columns of each matrix are orthogonal, such that for each $1\le i\le L$,

$$\tilde{W}_i=W_{i-1}\cdots W_1 A^T W_L\cdots W_{i-1}=U_{i-1}U_i^T.\qquad (36)$$

Multiplying Equation equation 35 for all $1\le i\le L$ (in the case $i=1$ we take $W_1\cdots W_{i-1}$ as identity), we get

$$(A^T\nabla^2 L(W)\cdot)^{L-1}A^T=(A^T W_L\cdots W_1\cdot)^{L-1}A^T=U_0-U_L^T;\qquad (37)$$

or in case where A is positive semi-definite,

$$A^{1/2}(A^{1/2}\nabla^2 L(W)\cdot A^{1/2}\cdot)^{L-1}A^{1/2}=U_0-U_L^T.\qquad (38)$$

But having access to Equations equation 36, we can write $\nabla^2 L(W\cdot)$ at the minimizer point $W=(W_1,\dots,W_L)$ as

$$\prod_{i=1}^{L}\|W_{i-1}\cdots W_1 A W_L\cdots W_{i-1}\|_F^2=\|Y\|_F^{2L}\|Y\|_{S_{2/L}}^{2-L}.$$

which based on Equation equation 37 is equal to

$$L\,\|(A^T\nabla^2 L(W)\cdot)^{L-1}A^T\|_{S_{2/L}}^{2-L};$$

or in the symmetric case is equal to

$$L \setminus A^{1-2}\check{S}A^{1-2}E\hat{}W \bullet A^{1-2}\bullet^{L\ 1} A^{1-2}\setminus_{S_{2-L}}^{2-L} :$$

This is the induced regularizer of the trace of Hessian over all interpolating solutions for linear network with depth L in the space of end-to-end matrices.

□

.5 Other Omitted Proofs

.5.1 Proof of Theorem 24

Proof. By Corollary 9, we know that with probability at least $1 \quad \exp\hat{} \quad \hat{}n\bullet\bullet$,

$$YE\hat{}W^{\ddagger}\bullet Y_{\ddagger} B 3YM^{\ddagger}Y_{\ddagger}:$$

Note by assumption,$n \quad \hat{}d_0 \quad d_L\bullet$. Thus it su ces to show that with probability at least $1 \quad \exp\hat{} \hat{}d_0 \quad d_L\bullet\bullet$, for all interpolating solutions in $H_{3YM^{\ddagger}Y_{\ddagger}} \quad \tilde{}h_M \ S \ MY_{\ddagger} B 3YM^{\ddagger}Y_{\ddagger}\bullet$, Equation (8.23) holds.

Recall $L\%E\hat{}W \bullet \check{Z}$ is the population squared loss of the end-to-end matrix$E\hat{}W \bullet > R^{d_0 \ d_L}$. Namely,

$$L\hat{}E\hat{}W \bullet\bullet < E_A\hat{}\grave{}A; E\hat{}W^{\ddagger}\bullet e \quad \grave{}A; M^{\ddagger}e\bullet^2 \quad E_A\grave{}A; E\hat{}W \bullet \quad M^{\ddagger}e^2 \quad YE\hat{}W^{\ddagger}\bullet \quad M^{\ddagger}Y_F^2 :$$

First, we bound the population Rademacher complexity of$H_{3YM^{\ddagger}Y_{\ddagger}}$. The empirical Rademacher complexity on$\tilde{}A_i\bullet_{i\ 1}^n$is

$$\begin{aligned} R_n\hat{}H_{3YM^{\ddagger}Y_{\ddagger}}\bullet \quad & \frac{1}{n}E_{\ \tilde{}\ 1\bullet^n} \sup_{h>H_{3YM^{\ddagger}Y_{\ddagger}}} \overset{n}{Q}_{i\ 1} {}_i h\hat{}A_i\bullet \\ & \frac{1}{n}E_{\ \tilde{}\ 1\bullet^n} \sup_{M\ YM\ Y_{\ddagger}B3YM^{\ddagger}Y_{\ddagger}} \overset{n}{Q}_{i\ 1} \grave{}A_i; M e \quad 3\tilde{}n \ YM^{\ddagger}Y_{\ddagger}Y\overset{n}{Q}_{i\ 1} {}_i A_i Y_2: \end{aligned}$$

Note that the matrix $A_{sum} \quad P_{i\ 1}^n {}_i A_i$ itself is an iid Gaussian matrix where each entry is sampled from$N\hat{}0; n\bullet$. Hence, from Proposition 2.4 in [246], we have the following tail bound on the spectral norm ofA_{sum}

$$P\hat{}YA_{sum}Y_2 \ Cc_1\sqrt{n}\hat{} \sqrt{d_0} \quad \sqrt{d_L}\bullet \quad \sqrt{n}t\bullet B 2e^{\ c_2t^2}; \tag{39}$$

This implies $EYA_{sum}Y_2 \quad O\%\sqrt{n}\hat{} \sqrt{d_0} \quad \sqrt{d_L}\bullet\check{Z}$, which in turn bounds the Rademacher complexity

$$\overline{R}_n\hat{}H_{3YM^{\ddagger}Y_{\ddagger}}\bullet \quad ER_n\hat{}H_{3YM^{\ddagger}Y_{\ddagger}}\bullet \quad O\left(\frac{\sqrt{d_0} \quad \sqrt{d_L}}{\sqrt{n}}\right) YM^{\ddagger}Y_{\ddagger}\text{‘} : \tag{40}$$

Note that the Gaussian distribution $G_{d_L\ d_0}$ is unbounded, which makes the value of the squared loss unbounded, while it is convenient to bound the generalization gap when the value of the loss is bounded. To cope with this fact, for a given threshold c, we define a truncated version of the loss denoted by $l_c(x, y) = \ell_c(x - y)$, plotted in Figure 3, which is a smooth approximation of the squared loss.

$$l_c(x, y) = \ell_c(x - y) = \begin{cases} (x-y)^2, & \text{if } |x-y| \le c, \\ 4c|x-y| - 2c^2, & \text{if } |x-y| \le 2c, \\ 2c^2, & \text{if } |x-y| > 2c. \end{cases} \tag{41}$$

It is easy to verify ℓ_c is 2-lipschitz in x. Also it is clear that $l_c(x, y) \le \max(2c^2, l(x, y))$ for all x, y and $l_c(x, y) = l(x, y)$ only when $|x - y| \le c$. Next, we define the c-cap population loss L_c with respect to ℓ_c:

$$L_c(M) = E_{A \sim G_{d_L\ d_0}} [\ell_c(\langle A, M \rangle - \langle A, M^\ddagger \rangle)].$$

Thus we have

$$L(M) - L_c(M) \le \mathbb{E}[\mathbb{1}_{|\langle A, M - M^\ddagger \rangle| > c} \ell_c(\langle A, M - M^\ddagger \rangle)].$$

But note that the variable $\langle A, M - M^\ddagger \rangle$ is a Gaussian variable with variance $\|M - M^\ddagger\|_F^2 \le \|M - M^\ddagger\|_*^2$. Hence, from Lemma 44, picking $c = \sqrt{\log n} \|M - M^\ddagger\|_*$, for all $M > H_3 \|M - M^\ddagger\|_*$,

$$0 \le L(M) - L_c(M) \le O\left(\frac{\|M - M^\ddagger\|_*^2 \log n}{n} \right).$$

Now using the Rademacher complexity bound in equation 40 and applying Theorem 27, we have for all interpolating solutions $M > H_3 \|M - M^\ddagger\|_*$ with probability at least $1 - \exp(-(d_0 + d_L))$:

$$\begin{aligned} L_c(M) &\le O\left(H \log^3(n) R_n^2 \left(\frac{c^2 (d_0 + d_L)}{n} \right) \right) \\ &\le O\left(\|M - M^\ddagger\|_*^2 \frac{(d_0 + d_L) \log^3 n}{n} \right), \end{aligned} \tag{42}$$

where H is the gradient smoothness of the loss ℓ_c which is 2 and L_c is the empirical loss defined in equation 8.2 with square loss substituted by ℓ_c. Above, we used the fact that ℓ_c is bounded by $2c^2$. □

Lemma 44. For standard Gaussian variable X, we have

$$\mathbb{E}[\mathbb{1}_{|X| > c} X^2] \le e^{c^2/2} \frac{2(c^2 + 2)}{c\sqrt{2\pi}}.$$

Figure 3: The smooth surrogate loss$\grave{s}_c$ as de ned in Equation equation 41 with parameter c 2.

Proof of Lemma 44.

$$
\begin{aligned}
E1_{s\,x\,s\,c}X^{2} \quad & 2\tilde{} \quad \frac{o}{2} \quad \int_{x\ c}^{a} x^{2} e^{\frac{x^{2}}{2}} dx \\
& B2\tilde{} \quad \frac{o}{2} \quad \int_{x\ c}^{a} \frac{x^{3}}{c} e^{\frac{x^{2}}{2}} dx \\
& 1\tilde{}\hat{c} \quad \frac{o}{2} \bullet \int_{x\ a\ c} x^{2} e^{\frac{x^{2}}{2}} dx^{2} \\
& 1\tilde{}\hat{c} \quad \frac{o}{2} \bullet \int_{x\ c^{2}} x e^{\frac{x}{2}} dx \\
& 1\tilde{}\hat{c} \quad \frac{o}{2} \bullet \hat{} \quad 0 \quad \hat{} \quad 2e^{\,c^{2}\tilde{}2\hat{}c^{2}} \quad 2\bullet\bullet\bullet \\
& e^{\,c^{2}\tilde{}2} \frac{2\hat{}\zeta_{0}^{2} \quad 2\bullet}{c \quad \overline{2}}:
\end{aligned}
$$

□

.5.2 Proof of Lemma 43

Proof of Lemma 43 . Consider its SVD decomposition of M, $M = \sum_{i=1}^{d} \sigma_i u_i v_i^T$, where σ_i's are the singular values and $\{u_i\}_{i=1}^{d}, \{v_i\}_{i=1}^{d}$ each is an orthonormal basis for $\mathbb{R}^d$. We can write

$$\begin{aligned}\sum_{i=1}^{n} \langle A_i, M \rangle^2 &= \frac{1}{n} \sum_{i=1}^{n} \Big(\sum_{j=1}^{d} \sigma_j u_j^T A_i v_j \Big)^2 \\ &= \frac{1}{n} \sum_{i=1}^{n} \sum_{j,k=1}^{d} \sigma_j \sigma_k \operatorname{tr}(A_i v_j u_j^T) \operatorname{tr}(A_i v_k u_k^T) \\ &= \sum_{j,k=1}^{d} \sigma_j \sigma_k \Big(\frac{1}{4n} \sum_{i=1}^{n} \operatorname{tr}(A_i (v_j u_j^T - v_k u_k^T))^2 - \operatorname{tr}(A_i (v_j u_j^T - v_k u_k^T))^2 \Big).\end{aligned}$$

But again using the $(2, \delta)$-RIP of $\{A_i\}_{i=1}^{n}$,

$$\begin{aligned}&(1-\delta) \|v_j u_j^T - v_k u_k^T\|_F^2 \le \frac{1}{4n} \sum_{i=1}^{n} \operatorname{tr}(A_i (v_j u_j^T - v_k u_k^T))^2 - \operatorname{tr}(A_i (v_j u_j^T - v_k u_k^T))^2 \le (1+\delta) \|v_j u_j^T - v_k u_k^T\|_F^2 \\ &(1-\delta) \|v_j u_j^T - v_k u_k^T\|_F^2 \le \frac{1}{4n} \sum_{i=1}^{n} \operatorname{tr}(A_i (v_j u_j^T - v_k u_k^T))^2 - \operatorname{tr}(A_i (v_j u_j^T - v_k u_k^T))^2 \le (1+\delta) \|v_j u_j^T - v_k u_k^T\|_F^2.\end{aligned}$$

This implies

$$\begin{aligned}&\frac{1}{4n} \sum_{i=1}^{n} \operatorname{tr}(A_i (v_j u_j^T - v_k u_k^T))^2 - \operatorname{tr}(A_i (v_j u_j^T - v_k u_k^T))^2 \\ &\le \frac{1}{2} (\|v_j u_j^T\|_F + \|v_k u_k^T\|_F) \cdot (1+\delta) \|v_j u_j^T - v_k u_k^T\|_F \le (1+\delta) \langle v_j u_j^T, v_k u_k^T \rangle.\end{aligned}$$

Summing this from j to k and noting that $\langle v_j u_j^T, v_k u_k^T \rangle$ is zero for $j \neq k$:

$$\sum_{i=1}^{n} \langle A_i, M \rangle^2 \le (1+\delta) \sum_{j=1}^{d} \sigma_j^2 + (1+\delta) \sum_{j \neq k} \sigma_j \sigma_k \le (1+\delta) \|M\|_F^2 + \delta \|M\|_*^2. \tag{43}$$

Similarly we obtain

$$\sum_{i=1}^{n} \langle A_i, M \rangle^2 \ge (1-\delta) \|M\|_F^2 - \delta \|M\|_*^2. \tag{44}$$

Combining Equations equation F.61 and equation C.24:

$$(1-\delta) \|M\|_F^2 - \delta \|M\|_*^2 \le \sum_{i=1}^{n} \langle A_i, M \rangle^2 \le (1+\delta) \|M\|_F^2 + \delta \|M\|_*^2. \tag{45}$$

This completes the proof. □

.6 Proof of Theorem 25

Proof of Theorem 25. Here we view matrices in $\mathbb{R}^{d_0 \times d_L}$ as $d_0 d_L$ dimensional vectors, hence by rotating a matrix with an orthogonal transformation we mean to rotate the corresponding

vector. Note that the minimum ℓ_2 solution of the regression problem is given by $\hat{M}$ defined as

$$\hat{M} = \sum_{i=1}^{n} A_i \left(\left\langle A_i, A_j \right\rangle_{1 \le i,j \le n} \right)^{-1} b_i .$$

First, note that if we rotate the ground-truth matrix $M^\ddagger$ with an arbitrary orthogonal matrix U, then $\hat{M}$ rotates according to the same U. Combining this with the fact the distribution on the measurement matrices is Gaussian and rotationally symmetric, we conclude that the population loss $L(\hat{M})$ is the same for all $M^\ddagger$. Hence, to lower bound the population loss, we can further assume that the entries of $M^\ddagger$ are sampled from standard Gaussian distribution. Hence, for any $M^\ddagger$ we can write

$$\begin{aligned}
\mathbb{E}_{\{A_i\}_{i=1}^n} L(\hat{M}) &= \mathbb{E}_{M^\ddagger} \mathbb{E}_{\{A_i\}_{i=1}^n} L(\hat{M}) \\
&= \mathbb{E}_{\{A_i\}_{i=1}^n} \mathbb{E}_{M^\ddagger} \left\| \hat{M} - M^\ddagger \right\|_F^2 \\
&= \mathbb{E}_{\{A_i\}_{i=1}^n} \left(1 - \frac{n}{d_0 d_L} \right) \left\| M^\ddagger \right\|_F^2 \\
&= \left(1 - \frac{n}{d_0 d_L} \right) \left\| M^\ddagger \right\|_F^2 .
\end{aligned}$$

where we used the fact that $\hat{M}$ is the projection of $M^\ddagger$ onto the subspace spanned by $\{A_i\}_{i=1}^n$. □

Appendix A

Simplicity Bias via Global Convergence of Sharpness Minimization

Abstract

The remarkable generalization ability of neural networks is usually attributed to the implicit bias of SGD, which often yields models with lower complexity using simpler (e.g. linear) and low-rank features [247]. Recent works have provided empirical and theoretical evidence for the bias of particular variants of SGD (such as label noise SGD) toward flatter regions of the loss landscape. Despite the folklore intuition that flat solutions are 'simple', the connection with the simplicity of the final trained model (e.g. low-rank) is not well understood. In this work, we take a step toward bridging this gap by studying the simplicity structure that arises from minimizers of the sharpness for a class of two-layer neural networks. We show that, for any high dimensional training data and certain activations, with small enough step size, label noise SGD always converges to a network that replicates a single linear feature across all neurons; thereby, implying a simple rank one feature matrix. To obtain this result, our main technical contribution is to show that label noise SGD always minimizes the sharpness on the manifold of models with zero loss for two-layer networks. Along the way, we discover a novel property – a local geodesic convexity – of the trace of Hessian of the loss at approximate stationary points on the manifold of zero loss, which links sharpness to the geometry of the manifold. This tool may be of independent interest.

A.1 Introduction

Overparameterized neural networks trained by stochastic gradient descent (SGD) have demonstrated remarkable generalization ability. The emergence of this ability, even when the network perfectly fits the data and without any explicit regularization, still remains a mystery [194]. A line of recent works have attempted to provide a theoretical basis for such

observations by showing that SGD has an implicit bias toward functions with "low complexity" among all possible networks with zero training loss [202, 234, 248 254]. In particular, it is noted that SGD tends to pick simpler features to fit the data over more complex ones [255 258]. This tendency is typically referred to as *simplicity bias*. Notably, [247] empirically observes that the feature matrix on the training set tends to become low-rank and that this property is amplified in deeper networks.

On the other hand, recently, a different form of implicit bias of SGD based on the sharpness of loss landscape has garnered interest. The study of flatness of the loss landscape and its positive correlation with generalization is not new (e.g. [215]), and has been supported by a plethora of empirical and theoretical evidence [11, 195, 197, 200 204, 207, 210, 211, 225, 227, 243, 244, 253, 259, 260]. It has been suggested that, after reaching zero loss, neural networks trained using some variants of SGD are biased toward regions of the loss landscape with low sharpness [11, 195 198, 259, 261, 262]. In particular, [259] discovered the sharpness-reduction implicit bias of a variant of SGD called *label noise SGD*, which explicitly adds noise to the labels. They proved that SGD locally diverges from points that are not stationary for the trace of the Hessian of the loss after fitting the training data. Along this line, [11] show that in the limit of step size going to zero, label noise SGD converges to a gradient flow according to the trace of Hessian of the loss after reaching almost zero loss. This flow can be viewed as a Riemannian gradient flow with respect to the sharpness on the manifold of zero loss, i.e., the flow following the gradient of sharpness projected onto the tangent space of the manifold. Furthermore, a standard Bayesian argument based on minimum description length suggests such flat minimizers correspond to simple models.

From a conceptual point of view, both simplicity bias and sharpness minimization bias suggest that SGD learns simple solutions; however, the nature of "simplicity" is different. Simplicity bias is defined in the feature space, meaning the model learns simple features, e.g., low-rank features, but low sharpness is defined in the parameter space, meaning the loss landscape around the learned parameter is flat, such that the model is resilient to random parameter perturbations and admits a short and simple description from a Bayesian perspective. It is thus an interesting question whether these two notions of simplicity for neural networks have any causal relationship. Existing works that attempt to study this connection are restricted to somewhat contrived settings (e.g. [253, 259, 263, 264]). For instance, [259] study 1-D two-layer ReLU networks and two-layer sigmoid networks trained on a single data point, where they show the stationary points of the trace of Hessian on the manifold of zero loss are solutions that are simple in nature. They further prove that label noise SGD locally diverges from non-stationary points. As another example, [11] and [265] independently show that label noise SGD can recover the sparse ground truth, but only for a simple overparameterized quadratic linear model. In this paper, we study the following fundamental question:

> (1) Is there a non-linear general setting where the sharpness-reduction implicit bias provably implies simplicity bias?

However, even if we can show that a model with minimum flatness is a simple model, a key open question is whether the sharpness-reduction implicit bias, characterized by the Riemannian gradient flow of sharpness in [11], even successfully minimizes the sharpness in the first place. The convergence of the gradient flow so far is only known in very simple

Results	Assumptions	Basic Description
Theorem 30	α; $\alpha\alpha\alpha\alpha$A0 or x^3 X is full-rank	The global minimizer of the squared loss, all neurons collapse into the same vector in the data subspace
Theorem 31	α; $\alpha\alpha\alpha\alpha$A0 X $\bar{}$X C I -normality [4]	The Riemannian gradient ow of sharpness converges to the global minimizer of sharpness on the manifold of zero loss
Theorem 29	α; $\alpha\alpha\alpha\alpha$A0 or x^3 X $\bar{}$X C I -normality [4]	label noise SGD with small enough step size converges to the global minimizer of the sharpness on the manifo

Table A.1: Summary of main theoretical results and key assumptions

settings, i.e., quadratically overparametrized linear nets [11]. In this regard, a central open question in the study of sharpness minimization for label noise SGD is:

> (2) Does label noise SGD converge to the global minimizers of the sharpness on the manifold of zero loss, and if it does, how fast does it converge? In particular, for what class of activations does global convergence happen?

In this paper, we show sharpness regularization provably leads to simplicity bias for two-layer neural networks with certain activation functions. We further prove that the Riemannian gradient ow on the manifold linearly converges to the global minimizer of the sharpness for all initialization . To our best knowledge, this is the rst global linear convergence result for the gradient ow of trace of Hessian on manifolds of minimizers.

More formally, we consider the mean squared loss

$$\grave{}\hat{} \bullet \quad Q_{i\ 1}^{n}\hat{}Q_{j\ 1}^{m}\ \hat{}\ _{j}\bar{}x_{i}\bullet \quad y_{i}\bullet^{2} \tag{A.1}$$

for a two-layer network model where the weights of the second layer are xed to one and $\tilde{}x_i; y_i\bullet_{i\ 1}^{n}$ are the training dataset.[1] We consider label noise SGD on , that is, running gradient descent on the loss $P_{i\ 1}^{n}\hat{}P_{j\ 1}^{m}\ \hat{}\ _{j}\bar{}x_{i}\bullet \quad y_{i} \quad _{t;i}\bullet^{2}$ at each step t, in which independent Gaussian noise $_{t;i} \quad N\hat{}0; \ ^{2}\bullet$ is added to the labels. The following is a corollary of the main results of our paper, which proves the asymptotic convergence of label noise SGD to the global minimizer of the sharpness. This result holds under Assumptions 2, 3, 4) and for su ciently small learning rate.

Theorem 29 (Convergence to simple solutions from any initialization) Given any A0 and arbitrary initial parameter 0 , for label noise SGD with any noise variance^2 A0 initialized at 0 , there exists a threshold $_0$ such that for any learning rate B $_0$, running label noise SGD with step size and number of iterations$T \quad O\hat{}\log\hat{}1\tilde{}\bullet \quad ^{2} \quad ^{2}\bullet$, with probability at least 1 ,

1. $\grave{}\hat{} \ T \bullet B$;
2. $\mathrm{Tr}\hat{}D^{2}\grave{}\hat{} \ T \bullet\bullet B\inf_{\grave{}} \bullet_{\ 0}\mathrm{Tr}\hat{}D^{2}\grave{}\hat{} \bullet\bullet \quad$;
3. For all neurons $_j$ T and all data pointsx_i, T$_j$ T $\bar{}x_i \quad ^{1}\hat{}y_i\tilde{}m\bullet TB$;

[1] We discuss how our result generalizes to the case of unequal weights at the end of Section A.4.2.

where $D^2\grave{}$ is the Hessian of the loss and we hide constants about loss and initialization 0 in $O\hat{}\bullet$.

Informally, Theorem 29 states that, rst, label noise SGD reaches zero loss and then (1) successfully minimizes the sharpness to its global optima (2) and more importantly, it recovers the simplest possible solution that ts the data, in which the feature matrix is rank one (see Appendix J.2.3 for proof details). In particular, Theorem 29 holds for pure label noise SGD without any additional projection.

- Simplicity bias: We prove that at every minimizer of the trace of Hessian, the pre-activation of di erent neurons becomes the same for every training data. In other words, all of the feature embeddings of the dataset with respect to di erent neurons collapse into a single vector, e.g., see Figure A.1. This combined with our convergence result mathematically proves that the low-rank feature observation by [247] holds in our two-layer network setting trained by sharpness minimization induced regularizer algorithms, such as label noise SGD and 1-SAM.

- Convergence analysis: Under an additional normality assumption on the activation (Assumption 3), we show that the gradient ow of the trace of Hessian converges exponentially fast to the global optima on the manifold of zero loss, and as a result, label noise SGD with a su ciently small step size successfully converges to the global minimizer of trace of Hessian among all the models achieving zero loss. Importantly, we do not assume additional technical conditions (such as PL inequality and non-strict saddle property used in prior works) about the landscape of the loss or its sharpness, and our constants are explicit and only depend on the choice of the activation and coherence of the data matrix. Moreover, our convergence results hold in the strong sense i.e., the convergence holds for the last iterations of label noise SGD.

  We have summarized our main results in Table A.1. The novelty of our approach (after that we show it converges to zero loss) is that we characterize the convergence on the manifold of zero loss in two phases (1) the rst phase where the algorithm is far from stationary points, trace of Hessian decreases with a at least constant rate and there is no convexity in trace of Hessian, (2) the second phase where the algorithm gets close to stationarity, we prove a novel g-convexity property of trace of Hessian which holds only locally on the manifold, but we show implies an exponentially fast convergence rate by changing the Lyapunov potential from value of trace of Hessian to the norm squared of its gradient on the manifold. We further prove that approximate stationary points are close to a global optimum via a semi-monotonicity property.

Interestingly, we observe this simplicity bias even beyond the regime of our theory; namely when the number of data points exceeds the ambient dimension, instead of a single vector, the feature embeddings of the neurons cluster and collapse into a small set of vectors (see Figure B.1).

Figure A.1: The rank of the feature and weight matrices go to one as Theorem 29 predicts in the high dimensional setting $d \asymp n$.

A.2 Related Work

Implicit Bias of Sharpness Minimization. Recent theoretical investigations, including those by [214], [195], and [11], have indicated that Stochastic Gradient Descent (SGD) with label noise intrinsically favors local minimizers with smaller Hessian traces. This is under the assumption that these minimizers are connected locally as a manifold and these analyses focus on the final phase of training when iterates are close to the manifold of minimizes. The effect of label noise or mini-batch noise in the central phase of training, i.e., when the loss is still substantially above zero is more difficult and is only known for simple quadratically overparametrized linear models [265, 266]. Further, [196] demonstrated that normalized Gradient Descent (GD) intrinsically penalizes the Hessian's largest eigenvalue. [227] proposes that such sharpness reduction phenomena could also be triggered by a multi-scale loss landscape. In the context of scale-invariant loss functions, [197] found that GD with weight decay implicitly reduces the spherical sharpness, defined as the Hessian's largest eigenvalue evaluated at the normalized parameter.

Another line of research examines the sharpness minimization effect of large learning rates, assuming that (stochastic) gradient descent converges at the end of training. This analysis primarily hinges on the concept of linear stability, as referenced in works by [225], [226], [227], and [228]. More recent theoretical analyses, such as those by [229] and [230], suggest that the sharpness minimization effect of large learning rates in gradient descent does not necessarily depend on the convergence assumption and linear stability. Instead, they propose a four-phase characterization of the dynamics in the so-called Edge of Stability regime, as detailed in [226].

Sharpness-aware Minimization (SAM, [204]) is a highly effective regularization method that improves generalization by penalizing the sharpness of the landscape. SAM was independently developed in [206, 207]. Recently, [198, 262] proved that SAM successfully minimizes the worst-direction sharpness under the assumption that the minimizers of training loss connect

as a smooth manifold. [198] also analyze SAM with batch size 1 for regression problems and show that the implicit bias of SAM becomes penalizing average-direction sharpness, which is approximately equal to the trace of the Hessian of the training loss. [267] observe that the SAM update rule for ReLU networks is locally biased toward sparsifying the features leading to low-rank features. However, their analysis is local and does not provide a convergence proof. They further con rm empirically that deep networks trained by SAM are biased to produce low-rank features.

Figure A.2: The feature embeddings of neurons collapse into clusters in the low dimensional regime d @n. The plot visualizes this phenomenon using the rst and second principal components of the feature embedding ‡X .

A.3 Main Results

Problem Setup. In this paper, we focus on the following two-layer neural network model:

$$r_{;NN}\hat{x}\bullet < \sum_{j\ 1}^{m} \hat{\ }_j \bar{x}\bullet: \tag{A.2}$$

Here $\hat{\ }_{1;:::;\ m}\bullet$ and are the set of parameters and activation of the neural network, respectively. As illustrated in Equation equation A.2, we assume that all the second-layer weights are equal to one. At the end of section A.4.2 we discuss how our approach in Theorem 30 generalizes to any xed choice of weights in the second layer. We denote the matrix of the weights by $>R^{m\ d}$, whosejth row is $_j$. Given a training dataset $\hat{x}_i; y_i\bullet_{i\ 1}^{n}$, we study the mean squared loss

$$`\hat{\ }\bullet < \sum_{i\ 1}^{n} \hat{r}_{;NN}\hat{x}_i\bullet \quad y_i\bullet^2: \tag{A.3}$$

For simplicity, we assume that the data points have unit norm, i.e.,$\|x_i\| = 1$. Let M denote the zero level set of L: $M = \{W \in \mathbb{R}^{m \times d} \mid L(W) = 0\}$. As we show in Lemma 225 in the Appendix, M is well-defined as a manifold due to a mild non-degeneracy condition implied by Assumption 2. We equip M with the standard Euclidean metric in $\mathbb{R}^{md}$.

The starting point of our investigation is the work [11], which characterizes the minimizer of the trace of the Hessian of the loss, $\mathrm{Tr}\,\nabla^2 L$, as the implicit bias of label noise SGD by showing that, in the limit of step size going to zero, label noise SGD evolves on the manifold of zero loss according to the following deterministic gradient flow:

$$\frac{d}{dt} W(t) = -\nabla \mathrm{Tr}\,\nabla^2 L(W(t)). \tag{A.4}$$

Here, ∇ is the Riemannian gradient operator on the manifold M, which is the projection of the normal Euclidean gradient onto the tangent space of M at $W(t)$. We use this result of [11] to relate the convergence of label noise SGD to the convergence of the flow equation A.4. Our main object of study in this paper is then the global convergence of this flow. Note that starting from a point $W(0) \in M$, $W(t)$ remains on M for all times t (for a quick recap on basic notions in Differential Geometry such as covariant derivative and gradient on a manifold, we refer the reader to Appendix ??.) On a manifold, similar to Euclidean space, having a zero gradient $\nabla \mathrm{Tr}\,\nabla^2 L(W^*) = 0$ (the gradient defined on the manifold) at some point W^* is a necessary condition for being a local minimizer, or equivalently the projection of Euclidean gradient onto the tangent space at W^* has to vanish. More generally, we call an ε-stationary point on M if $\|\nabla \mathrm{Tr}\,\nabla^2 L(W)\| \le \varepsilon$.

A.3.1 Characterization of Stationary Points

Before characterizing the stationary points of $\mathrm{Tr}\,\nabla^2 L$, we introduce an important assumption on the activation function, namely the convexity and positivity of its derivative.

Assumption 2 (Strict positivity and convexity of σ). For all $z \in \mathbb{R}$, $\sigma'(z) \ge \alpha_1 > 0$, $\sigma''(z) \ge \alpha_2 > 0$.

For example, Assumption 2 is satisfied for $\sigma(x) = x^3 + x$ with constants $\alpha_1 = 1, \alpha_2 = 1$. In Lemma 224, given the condition $\sigma'' \ge \alpha_2 > 0$ we show that $W(t)$ remains in a bounded domain along the gradient flow, which means having the weaker assumption that $\sigma'' > 0$ automatically implies Assumption 2 for some positive constants α_1 and α_2. Another activation of interest to us that happens to be extensively effective in NLP applications is the cube (x^3) activation [268]. Although x^3 does not satisfy the strict positivity required in Assumption 2, we separately prove Theorem 30 for x^3 (see Lemma 227 in the Appendix.)

Next, let $X = [x_1 \ldots x_n]$ be the data matrix. We make a coherence assumption on X which requires the input dimension d to be at least as large as the number of data points, denoted by n.

Assumption 3 (Data matrix coherence). The data matrix X satisfies $X^\top X \approx I$.

Assumption 2 implies that M is well-defined as a manifold (see Lemma 225 for a proof). Under aforementioned Assumptions 2 and 3, we show that the trace of the Hessian regularizer has a unique stationary point on the manifold.

Theorem 30 (First-order optimal points). Under Assumptions 2 and 3, the first order optimal points and global optimums of Tr hess on M coincide and are equal to the set of all $\ddagger$ $\ddagger_j{}_{j=1}^{m}$ such that for all $i > n$ and $j > m$ satisfy the following:

$$\ddagger_j \bar{x}_i \quad 1\hat{}y_i \sim m\bullet: \tag{A.5}$$

Note that instead of Assumption 3, just having the linear independence of $\tilde{o}x_i\bullet_{i=1}^{n}$ suffices to prove Theorem 30, as pointed out in section A.4.2. In particular, Theorem 30 means that the feature matrix is rank one at a global optimum $\ddagger$, which together with our main convergence result Theorem 29 proves the low-rank bias conjecture proposed by [247] in the two-layer network setting for label noise SGD and 1-SAM. Notice the effect of the number of neurons on the global minimizers of the implicit bias in Equation equation A.5. While Theorem 30 concerns networks with equal second-layer weights, the case of unequal weights can be analyzed similarly, as we discuss after the proof of Theorem 30 in Section A.4.2.

A.3.2 Convergence Rate

Next, to state our convergence result, we introduce the key-normality assumption, under which we bound the rate of convergence of the trace of Hessian to its global minimizer.

Assumption 4 (-normality). For all $z > R$ the second derivative of $\hat{}z\bullet$ can be bounded by the first and third derivatives as:

$$\text{œ}\text{º}z\bullet \; B \; \text{œ}\hat{}^2z\bullet \; \text{œ}\text{œ}\text{ß}z\bullet:$$

An example class of activations that satisfies both Assumptions 2 and 4 is of the form $\hat{}z\bullet \quad z^{2k\ 1} \quad z$ for $\quad A0$, which is well-known in the deep learning theory literature [249, 251, 269, 270]. Notably, these activations satisfy Assumption 4 with normality coefficient $\min\tilde{}\frac{1}{\hat{}2k\ 1\bullet^2\hat{}2k\ 1\bullet}; \ ^2\bullet$.

Under Assumptions 2, 3, and 4, we show that $\hat{}t\bullet$ converges to $\ddagger$ exponentially fast in Theorem 31. This results from strong local g-convexity of trace of Hessian for approximate stationary points (see Lemmas 48 and 219), plus a semi-monotonicity property for the trace of Hessian (Lemma 50).

Theorem 31 (Convergence of the gradient flow) Consider the limiting flow of Label noise SGD on the manifold of zero loss, which is the gradient flow in equation A.4. Then, under Assumptions 2,3, and 4,

(C.1) The gradient flow reaches -stationarity for all times

$$t \; C\frac{\text{Tr}\,\text{hess}\hat{}\ \hat{}0\bullet\bullet}{^2} \quad \frac{\log\hat{}\ ^2\sim\hat{}\%_1^2\%_2^2\ ^2\bullet Y1Y\bullet}{\%_1\%_2}:$$

(C.2) For all $j > m$ and $i > n$, the dot product of the jth neuron to the ith data point becomes -close to that of any global optimum $\ddagger$:

$$T_j\,\hat{}t\bullet\bar{x}_i \quad \ddagger_j\bar{x}_i TB \; ; \tag{A.6}$$

where $\ddagger$ is defined in Theorem 30.

The formal proof of Theorem 31 is given in Section J.2.3. We sketch its proof in Section A.4.3. Note that the rate is logarithmic in the accuracy parameter , i.e., the ow has exponentially fast convergence to the global minimizer.

A.4 Proof Sketches

Before delving into the technical details of the proofs, we discuss some necessary background and additional notation used in the proofs.

A.4.1 Additional Notation

We use $f_i(\cdot)$ to denote the output of the network on the ith input x_i, i.e.,

$$f_i(\cdot) = r_{;NN}(x_i;\cdot). \tag{A.7}$$

We use $f(\cdot) = (f_1(\cdot);\dots;f_n(\cdot))$ to denote the array of outputs of the network on $\{x_i\}_{i=1}^n$. We use D for the Euclidean directional derivative or Euclidean gradient, and ∇ for the covariant derivative on the manifold or the gradient on the manifold. We denote the Jacobian of f at point θ by $Df(\theta)$ whose ith row is $Df_i(\theta)$, the Euclidean gradient of f_i at θ. Recall the definition of the manifold of zero loss as the zero level set of the loss, which is the intersection of zero level sets of functions $\{s_i(\theta) : i = 1,\dots,n\}$, where $s_i(\theta) = (f_i(\theta) - y_i)^2$. The tangent space $T(\theta)$ of M at point θ can be identified by the tangents to all curves on M passing through θ, and the normal space $T^N(\theta)$ in this setting is just the orthogonal subspace to $T(\theta)$. We denote the projection operators onto $T(\theta)$ and $T^N(\theta)$ by P and P^N, respectively. For a set of vectors $\{v_i\}_{i=1}^n$ where for all $i = 1,\dots,n$, $v_i \in \mathbb{R}^d$, we use the notation $(v_i)_{i=1}^n$ to denote the vector in $\mathbb{R}^{nd}$ obtained by stacking vectors $\{v_i\}_{i=1}^n$.

A.4.2 Proof Sketch of Theorem 30

In this section, we describe the high-level proof idea of Theorem 30. Note that Theorem 30 characterizes the first order stationary points of Trhess on M, i.e., points θ^* on M for which $\nabla\text{Trhess}(\theta^*) = 0$. The starting point of the proof is the observation that the gradients $Df_i(\theta)$ form a basis for the normal space at.

Lemma 45 (Basis for the normal space) For every , the set of vectors $(Df_i(\theta))_{i=1}^n$ form a basis for the normal space $T^N(\theta)$ of M at .

For a stationary point θ^* with $\nabla F(\theta^*) = 0$, the Euclidean gradient at θ^* should be in the normal space $T^N_{\theta^*}(\theta)$. But by Lemma 45, this means there exist coefficients $(\lambda_i)_{i=1}^n$ such that

$$D(\text{Trhess})(\theta^*) = \sum_{i=1}^n 2\lambda_i Df_i(\theta^*). \tag{A.8}$$

To further understand what condition equation A.8 means for our two-layer network equation A.2, we first state a result from prior work in Lemma 46 (see e.g. [259]) regarding the

trace of the Hessian of the mean squared loss for some parameter $\theta \in M$ (see Section J.3.2 for a proof).

Lemma 46. For the loss on the dataset defined in Equation equation A.3 and parameters with $\ell(\theta) < 0$, we have

$$\mathrm{Tr}\,\mathrm{hess}(\theta) = \sum_{i=1}^{n} \|Df_i(\theta)\|^2.$$

Using Lemma 46, we can explicitly calculate the trace of Hessian for our two-layer network in equation A.2.

Lemma 47 (Trace of Hessian in two-layer networks) For the neural network model equation A.2 and the mean squared loss in Equation equation A.3, for any θ with $\ell(\theta) < 0$ we have

$$\mathrm{Tr}\,\mathrm{hess}(\theta) = \sum_{i=1}^{n}\sum_{j=1}^{m} \sigma'(\langle w_j, x_i\rangle)^2. \tag{A.9}$$

Now we are ready to prove Theorem 30.

Proof of Theorem 30. From Equation equation A.8 and by explicitly calculating the gradients $\{Df_i(\theta)\}_{i=1}^n$ using Lemma 47, we have

$$\sum_{i=1}^{n} 2\,\sigma(\langle w_j, x_i\rangle)\,\sigma'(\langle w_j, x_i\rangle)\,x_i \Big|_{j=1}^{m}$$
$$\sum_{i=1}^{n} 2\,y_i\,\sigma'(\langle w_j, x_i\rangle)\,x_i \Big|_{j=1}^{m}.$$

But using our assumption that the data points $\{x_i\}_{i=1}^n$ are linearly independent, we have for all $i > n$ and $j > m$, $\sigma'(\langle w_j, x_i\rangle) \neq y_i$. Now, because σ' is positive and σ is strictly monotone, its inverse is well-defined:

$$\langle w_j, x_i\rangle < (\sigma')^{-1}(|y_i|); \tag{A.10}$$

This implies $|\langle w_j, x_i\rangle| < |(\sigma')^{-1}(|y_i|)|$. Therefore,

$$y_i \sum_{j=1}^{m} |\langle w_j, x_i\rangle| \geq m\,|(\sigma')^{-1}(|y_i|)|. \tag{A.11}$$

But note that from strict monotonicity of σ from Assumption 2, we get that it is invertible. Therefore, Equation equation A.11 implies

$$\max_i |\langle w_j, x_i\rangle| \geq |(\sigma')^{-1}(|y_i|)|, \quad \text{and} \quad \max_i |\langle w_j, x_i\rangle| \geq |(\sigma')^{-1}(|y_i|)|.$$

This characterizes the first order optimal points of $\mathrm{Tr}\,\mathrm{hess}$.

On the other hand, note that $\mathrm{Tr}\,\mathrm{hess} \geq 0$, so its infimum over M is well-defined. Moreover, from Lemma 224 the set of θ's with $\mathrm{Tr}\,\mathrm{hess}(\theta) \leq Bc$ for arbitrary c is bounded in $\mathbb{R}^{md}$. Therefore $\mathrm{Tr}\,\mathrm{hess}$ has a global minimizer on M. But the global minimizers are also stationary points and from Equation equation A.10 we see that all of the first order optimal points have the same value of $\mathrm{Tr}\,\mathrm{hess}$. Therefore, all first order optimal points are global optima and satisfy Equation equation A.10. □

Results for General Second Layer Weights: Note that if the weights of the second layer are xed arbitrarily to $\hat{w}_j \bullet_j^m 1$, then Equation equation A.10 simply turns into $\ddagger\bar{x}_i$ $\text{œ}\text{c}\text{d}^{\wedge}$ $_i \sim w_j \bullet$; namely, the $\text{œ}\text{e}\text{e}$ mage of the feature matrix $\ddagger X$ is rank one. Interestingly, this shows that in the general case, the right matrix to look for obtaining a low-rank structure is the embedding of the feature matrix by the second derivative of the activation (for the cube activation this embedding is the feature matrix itself.) Next, we investigate the convergence of the gradient ow in Equation equation A.4.

A.4.3 Proof Sketch of Theorem 31

The rst claim (C.1) of Theorem 31 is that after time $t \; C \; \tilde{}\hat{}\log\hat{}1 \sim \bullet\bullet$, the gradient will always be smaller than . It is actually trivial to prove that there exists some time at most $t \quad O\hat{}1 \sim {}^2\bullet$ such that $\Y\text{©}\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet\Y\ B$ without any assumption, based on a standard argument which is a continuous version of descent lemma. Namely, if the gradient of $\mathrm{Tr}\,\mathrm{hess}$ remains larger than some $A0$ along the ow until time $t \; A\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}0\bullet\bullet \sim {}^2$, then we get a contradiction:

$$\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}0\bullet\bullet \quad \mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet$$
$$S_0^t \; [\text{©}\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet Z^2 \; A\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}0\bullet\bullet :$$

Our novelty here is a new characterization of the loss landscape under Assumptions 2, 3 (Lemma 48 proved in Appendix J.2.1).

Lemma 48 (PSD Hessian when gradient vanishes)Suppose that activation satis es Assumptions 2, 3, and 4. Consider a point on the manifold where the gradient is small, $\text{namely}\ \Y\text{©}\ \mathrm{Tr}\,\mathrm{hess}\hat{}\ \bullet\Y\ B \quad \bar{}$. Then, the Hessian of $\mathrm{Tr}\,\mathrm{hess}$ on the manifold is PSD at point , or equivalently $\mathrm{Tr}\,\mathrm{hess}$ is locally g-convex on M around .

Lemma 48 implies that whenever the gradient is su ciently small, the time derivative of the squared gradient norm will also be non-positive:

$$\frac{d}{dt}\Y\text{©}\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet\Y^2$$
$$\text{©}\ \mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet \bar{}\text{©}^2\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet \text{©}\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet \; B0;$$

Therefore, once the gradient is su ciently small at some t, it will always remain small for all $t\text{œ}At$. In fact, we show that the Hessian of $\mathrm{Tr}\,\mathrm{hess}$ on the manifold is strictly positive with a uniform $\%_1\%_2$ lower bound in a suitable subspace which includes the gradient (Lemma 219). Then

$$\frac{d\Y\text{©}\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet\Y^2}{dt}$$
$$\text{©}\ \mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet \bar{}\text{©}^2\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet \text{©}\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet$$
$$B \quad \%_1\%_2 \quad \Y\text{©}\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet\Y^2;$$

which further implies a linear convergence of gradient norm:

$$\Y\text{©}\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}t\bullet\bullet\Y^2 \; B \; \Y\text{©}\mathrm{Tr}\,\mathrm{hess}\hat{}\ \hat{}\ \hat{}t_0\bullet\bullet\Y^2 e^{\hat{}t \; t_0\bullet\%_1\%_2} :$$

Next, we state the high level ideas that we use to prove Lemma 48, which is the key to proving Theorem 31. Before that, we review some necessary background.

Computing the Hessian on M. To prove Lemma 48, we calculate the Hessian of $F < Tr\,hess$ on the manifold in Lemma 49. The Hessian of a function F on R is defined as $D^2F(u,w) = \grave{D}\hat{D}F\hat{}(\cdot\cdot)(u,w)$, where $DF\hat{}(\cdot)$ is the usual Euclidean gradient of F at $\cdot$, and $D\hat{}(\cdot)(u)$ denotes directional derivative. To calculate the Hessian on the manifold, one needs to substitute the Euclidean gradient $DF\hat{}(\cdot)$ by the gradient $\nabla F\hat{}(\cdot)$ on the manifold. Moreover, $D\hat{}DF\hat{}(\cdot\cdot)(u)$ has to be substituted by the covariant derivative $\nabla_u\nabla F\hat{}(\cdot)$, a different differential operator than the usual derivative. We recall the characterization of the covariant derivative as the projection of the Euclidean directional derivative onto the tangent space. For more background on covariant differentiation, we refer the reader to Appendix ??.

Fact 2. For vector fields V, W on M, we have $\nabla_V W\hat{}(\cdot) = P\hat{}DW\hat{}(\cdot\cdot)(V)$.

We also recall the definition of Hessian $\nabla^2 F$ on M based on covariant derivative.

Fact 3. The Hessian of F at point $\cdot$ on M is given by $\nabla^2 F\hat{}(w,u) = \grave{\nabla}_w\nabla F(u)$.

Next, we state Lemma 49, which is proved in Appendix J.2.1.

Lemma 49 (Hessian of the implicit regularizer on the manifold). Recall that $\{Df_i\hat{}(\cdot)\}_{i=1}^n$ is a basis for the normal space $T^N\hat{}M(\cdot)$ according to Lemma 45. Let $\{\alpha\hat{}_i(\cdot)\}_{i=1}^n$ be the coefficients representing $P^N\hat{}D\hat{}Tr\,hess(\cdot\hat{}\cdot\cdot) > T^N\hat{}M(\cdot)$ in the basis $\{Df_i\hat{}(\cdot)\}_{i=1}^n$, i.e.

$$P^N\check{S}D\hat{}Tr\,hess(\cdot\hat{}\cdot\cdot) = \sum_{i=1}^n \alpha_i Df_i\hat{}(\cdot).$$

Then, the Hessian of $Tr\,hess$ on M can be explicitly written (in the Euclidean chart R) using α as

$$\begin{aligned}\nabla^2 Tr\,hess\hat{}(\cdot)(u,w) &= D^2 Tr\,hess\hat{}(\cdot)(u,w) - \sum_{i=1}^n \alpha_i D^2 f_i\hat{}(\cdot)(u,w), \end{aligned} \tag{A.12}$$

for arbitrary $u, w > T\hat{}M(\cdot)$, where D^2 and ∇^2 denote the normal Euclidean Hessian and Hessian over the manifold.

Observe in the formula of $\nabla^2\hat{}Tr\,hess(\cdot)$ in Equation equation A.12 the first term is just the normal Euclidean Hessian of $Tr\,hess$ while we get the second projection term due to the difference of covariant differentiation with the usual derivative. Finally, to prove the second claim (C.2) of Theorem 31, we prove Lemma 50 in Appendix J.3.5 which shows that the small gradient norm of $Y\nabla Tr\,hess\hat{}(\cdot)Y$ implies the (approximate) alignment of features.

Lemma 50 (Small gradient implies close to optimum). Suppose $Y\nabla Tr\,hess\hat{}(\cdot)Y$. Then, for all $i > n, j > m$, $U_j\bar{}x_i \ddagger\bar{}x_i U$ or $\tilde{}\hat{}\bar{}\%_1\%_2(\cdot)$; for $\ddagger$ in Equation equation A.5.

A.5 Experimental Setup

We examine our theory on the convergence to a rank one feature matrix in Figure A.1 via a synthetic experiment by considering a network with $m = 10$ neurons on ambient dimension

$d = 3$ and $n = 3$ data points. We further pick learning rate $= 0.05$ and noise variance $= 0.03$ for implementing label noise SGD. Each entry of the data points is generated uniformly on $[0, 1)$, which is the same data generating process in all the experiments. As Figure A.1 shows, the second and third eigenvalues converge to zero which is predicted by Theorem 30. On the other hand, in a similar setting with the same learning rate $= 0.05$ but larger $= 0.2$ we run a synthetic experiment with $d = 5$ this time smaller than the number of data points $n = 6$. In this case, the feature matrix cannot be rank one anymore. However, one might guess that it still converges to something low-rank even though our theoretical result is shown only for the high dimensional case; in this case, the smallest possible rank to fit the data is indeed two, and surprisingly we see in Figure J.1 that except the first and second eigenvalues of the feature matrix, the rest go to zero. To further test extrapolating our theory to the low dimensional case $d = O(n)$, this time we use a higher number of data points $n = 15$ with ambient dimension $d = 10$ and 30 neurons, with learning rate $= 0.1$ and noise variance $= 0.2$. We observe a very interesting property: even though the feature matrix is high rank in this case, the neurons tend to converge to one another, especially when they are close, as if there is a hidden attraction force between them. To visualize our experimental discovery, in Figure B.1 we plot the first two principal components (with the largest singular values) of the feature vectors for each neuron, and observe how it changes while running label noise SGD. We further pick 12 neurons for which the first and second components are bounded in $[0, 0.1]$ and $[0, 0.5]$, respectively. As one can see in Figure B.1, most of the principal components of the neurons except possibly one outlier are converging to the same point. This is indeed the case for the feature vectors of neurons as well, namely, they collapse into clusters, but unfortunately, we only have two dimensions to visualize this, and we have picked the most two effective indicators (the first and second principal components.)

A.6 Conclusion

In this paper, we take an important step toward understanding the implicit bias of label noise SGD with trace of Hessian induced regularizer for a class of two layer networks. We discover an intriguing relationship between this induced regularizer and the low-rank simplicity bias conjecture of neural networks proposed in [247]: we show that by initializing the neurons in the subspace of high-dimensional input data, all of the neurons converge into a single vector. Consequently, for the final model, (i) the rank of the feature matrix is effectively one, and (ii) the functional representation of the final model is *very simple*; specifically, its sub-level sets are half-spaces. To prove this, in spite of the lack of convexity, we uncover a novel structure in the landscape of the loss regularizer: the trace of Hessian regularizer becomes locally geodetically convex at points that are approximately stationary. Furthermore, in the limit of step size going to zero, we prove that label noise SGD or 1-SAM converges *exponentially fast* to a global minimizer.

Furthermore, generalizing the class of activations that enjoy fast convergence or proving the existence of fundamental barriers and handling the case of low-dimensional input data are interesting future directions. Based on the compelling empirical evidence from [247] and our results for two-layer networks, we hypothesize that a low-rank simplicity bias can also be

shown for deeper networks, possibly using the sharpness minimization framework.

Impact Statement

This paper presents work whose goal is to advance the eld of Machine Learning. There are many potential societal consequences of our work, none which we feel must be speci cally highlighted here.

Acknowledgement

This research was supported in part by O ce of Naval Research grant N00014-20-1-2023 (MURI ML-SCOPE), NSF AI Institute TILOS (NSF CCF-2112665), NSF award 2134108, the Alexander von Humboldt Foundation, and the NSF TRIPODS program (award DMS-2022448).

Appendix B

Can Looped Transformers Learn to Implement Multi-step Gradient Descent for In-context Learning?

Abstract

The remarkable capability of Transformers to do reasoning and few-shot learning, without any ne-tuning, is widely conjectured to stem from their ability to implicitly simulate a multi-step algorithms such as gradient descent with their weights in a single forward pass. Recently, there has been progress in understanding this complex phenomenon from an expressivity point of view, by demonstrating that Transformers can express such multi-step algorithms. However, our knowledge about the more fundamental aspect of its learnability, beyond single layer models, is very limited. In particular, can training Transformers enable convergence to algorithmic solutions? In this work we resolve this for in-context linear regression with linear looped Transformers a multi-layer model with weight sharing that is conjectured to have an inductive bias to learn x-point iterative algorithms. More speci cally, for this setting we show that the global minimizer of the population training loss implements multi-step preconditioned gradient descent, with a preconditioner that adapts to the data distribution. Furthermore, we show a fast convergence for gradient ow on the regression loss, despite the non-convexity of the landscape, by proving a novel gradient dominance condition. To our knowledge, this is the rst theoretical analysis for multi-layer Transformer in this setting. We further validate our theoretical ndings through synthetic experiments.

B.1 Introduction

Transformers [271] have completely revolutionized the eld of machine learning and have led to state-of-the-art models for various natural language and vision tasks. Large scale Transformer models have demonstrated remarkable capabilities to solve many di cult problems, including those requiring multi-step reasoning through large language models [272, 273]. One such

particularly appealing property is their few-shot learning ability, where the functionality and predictions of the model adapt to additional context provided in the input, without having to update the model weights. This ability of the model, typically referred to as in-context learning , has been crucial to their success in various applications. Recently, there has been a surge of interest to understand this phenomenon, particularly since [7] empirically showed that Transformers can be trained to solve many in-context learning problems based on linear regression and decision trees. Motivated by this empirical success, [9, 10] theoretically showed the following intriguing expressivity result: multi-layer Transformers with linear self-attention can implement gradient descent for linear regression where each layer of Transformer implements one step of gradient descent. In other words, they hypothesize that the in-context learning ability results from approximating gradient-based few-shot learning within its forward pass. [274], further, extended this result to more general model classes.

While such an approximation is interesting from the point of view of expressivity, it is unclear if the Transformer model can learn to implement such algorithms. To this end, [275, 276] theoretically show, in a Gaussian linear regression setting, that the global minimizers of a one-layer model essentially simulate a single step of preconditioned gradient descent, and that gradient ow converges to this solution. [275] further show for the multi-layer case that a single step of gradient descent can be implemented by some stationary points of the loss. However, a fundamental characterization of all the stationary points for multi-layer Transformer, and the convergence to a stationary point that implements multi-step gradient descent, remains a challenging and important open question.

In this work, we focus our attention on the learnability of such multi-step algorithms by Transformer models. Instead of multi-layer models, we consider a closely related but di erent class of models called looped Transformers, where the same Transformer block is looped multiple times for a given input. Since the expectation from multi-layer models is to simulate an iterative procedure like multi-step gradient descent, looped models are a fairly natural choice to implement this. There is growing interest in looped models with recent results [277] theoretically showing that the iterative nature of the looped Transformer model can be used to simulate a programmable computer, thus allowing looped models to solve problems requiring arbitrarily long computations. Looped Transformer models are also conceptually appealing for learning iterative optimization procedures the sharing of parameters across di erent layers, in principle, can provide a better inductive bias than multi-layer Transformers for learning iterative-optimization procedures. In fact, by employing a regression loss at various levels of looping, [278] empirically nd that looped Transformer models can be trained to solve in-context learning problems, and that looping on an example for longer and longer at test time converges to a desirable xed-point solution, thus leading them to conjecture that looped models can learn to express iterative algorithms[1].

Despite these strong expressivity results for looped models and their empirically observed inductive bias towards simulating iterative algorithms, very little is known about the optimization landscape of looped models, and the theoretical convergence to desirable and interpretable iterative procedures. In fact, a priori it is not clear why training should even succeed given that looped models heavily use weight sharing and thus do not enjoy the optimization bene ts

[1] The algorithm should converge to a desirable xed point.

of overparameterization that has been well studied [279, 280]. In this work, we delve deeper into the problem of optimizing looped Transformers and theoretically study their landscape and convergence for in-context linear regression under the Gaussian data distribution setting used in [275, 276]. In particular, the main contributions of our paper are as follows:

- We obtain a precise characterization of the global minimizer of the population loss for a linear looped Transformer model, and show that it indeed implements multi-step preconditioned gradient descent with pre-conditioner close to the inverse of the population covariance matrix, as intuitvely expected.
- Despite the non-convexity of the loss landscape, we prove the convergence of the gradient ow for in-context linear regression with looped Transformer. To our knowledge, ours is the rst such convergence result for a network beyond one-layer in this setting.
- To show this convergence, we prove that the loss satis es a novel gradient-dominance condition, which guides the ow toward the global optimum. We expect this convergence proof to be generalizable to rst-order iterative algorithms such as SGD with gradient estimate using a single random instance [281].
- We further translate having a small sub-optimality gap, achieved by our convergence analysis, to the proximity of the parameters to the global minimizer of the loss.

B.2 Related Work

In-context learning. Language models, especially at larger scale, have been shown to empirically demonstrate the intriguing ability to in-context learn various tasks on test data [272] More recently, [7] formalized in-context learning ability and empirically observed that Transformers are capable of in-context learning some hypothesis classes such as linear or two layer neural networks, sometimes improving over conventional solvers. There have since been many paper studying this intriguing in-context learning phenomenon [9, 10, 282, 283]

Transformers in modeling iterative optimization algorithms. [284] rst observed that neural networks with residual connections are able to implicitly implement gradient descent. [9, 10] use this line of reasoning for in-context learning by constructing weights for linear self-attention layers that can emulate gradient descent for various in-context learning tasks, including linear regression. Furthermore, [10] empirically investigate various in-context learners that Transformers can learn as a function of depth and width. Also, [285] hypothesize the ability of Transformers to (i) build an internal loss based on the speci c in-context task, and (ii) optimize over that loss via an iterative procedure implemented by the Transformer weights. [274] generalize the results to show that Transformers can implement gradient descent over a smaller Transformer. Recently, [286] empirically observe that Transformers can learn to emulate higher order algorithms such as Newton's method that converge faster than gradient descent.

Transformers in reasoning and computation. Indeed the in-context capabilities of Transformers in doing reasoning at test time and emulating an input-speci c algorithm as a computer bear deep similarities [287 289]. Years before the advent of Transformers, [290]

study the Turing completeness of recurrent neural networks. To show the computational power of Transformer as a programmable device, [260, 291, 292] demonstrate that Transformers can simulate Turing machines. Furthermore, [293] propose using Transformer as programmable units and construct a compiler for the domain speci c programming language called RASP. [291] further nd a more e cient implementation of a programming language that is also Turing complete using looped Transformers, without scaling with the number of lines of code. More recently [277] used looped models to simulate a single-instruction program. [278] show that looped Transformers can in-context learn solvers for linear regression or decision trees as well as normal Transformers but with much fewer parameters.

B.3 Preliminaries

B.3.1 In-context learning (ICL)

One of the surprising emergent abilities of large language models is their ability to adapt to speci c learning tasks without requiring any additional ne tuning. Here we restate the formalism of in-context learning introduced by [7]. Suppose for a class of functions F and input domain X, we sample an in-context learning instance $\tilde{} \sim (x_i, y_i)_{i=1}^n, x_q)$ by sampling $x_i \sim D_X$ and $f \sim D_F$ independently, then calculating $y_i = f(x_i)$. An in-context learner M parameterized by θ is then a mapping from the instance $\tilde{}$ to a prediction for the label of the query point $f(x_q)$. The population loss of M is then de ned as

$$\ell(M_\theta) = \mathbb{E}_{f, D_X}\left[M_\theta(\tilde{}) - f(x_q)\right]^2 \tag{B.1}$$

B.3.2 Linear regression ICL setup

In this work, we consider linear regression in-context learning; namely, we assume sampling a linear regression instance is given by $\tilde{} = (x_i, y_i)_{i=1}^n, x_q)$ where for $w^* \sim N(0, \frac{1}{d} I_d)$, $x_i \sim N(0, \frac{1}{d} I_d)$ we have $y_i = f_{w^*}(x_i) = w^{*\top} x_i$ for all $i \geq n$. The goal is to predict the label of x_q, i.e. $w^{*\top} x_q$. De ne the data matrix $X \in \mathbb{R}^{d \times n}$, whose columns are the data points $(x_i)_{i=1}^n$:

$$X = [x_1, \ldots, x_n]$$

We further assume $n \geq d$, i.e. the number of samples is larger than the dimension. This combined with the fact that I is realizable implies that we can recover w^* from $(x_i, y_i)_{i=1}^n$ by the well-known pseudo-inverse formula:

$$w^* = (XX^\top)^{-1} Xy.$$

While a reasonable option for the in-context learner $M(\theta)$ is to implement $(XX^\top)^{-1} Xy$, matrix inversion is arguably an operation that can be costly for Transformers to implement.

On the other hand, it is known that linear regression can also be solved by first order algorithms that move along the negative gradient direction of the loss

$$\ell(w) = \frac{1}{2n}\|Xw - y\|^2.$$

Using a standard analysis for smooth convex optimization, since the Hessian of the loss $\|Xw - y\|^2$ is $XX^\top$ with condition number κ, gradient descent with step size $\frac{1}{\beta}$ converges in $O(\kappa)$ iterations. This means that we need $O(\kappa)$ many layers in the Transformer to solve linear regression. Particularly, [9] show a simple weighting strategy for the key, query, and value matrices of a linear self-attention model so that it implements gradient descent, which we introduce in the next section.

B.3.3 Linear self-attention layer

Here we define a single attention layer that forms the basis of the linear Transformer model we consider. Define the matrix $Z^{(0)}$, which we use as the input prompt to the Transformer, by combining the data matrix X, their labels y, and the query vector x_q as

$$Z^{(0)} = \begin{bmatrix} X & x_q \\ y^\top & 0 \end{bmatrix}.$$

Following [9, 275, 294], we consider the linear self-attention model $\mathrm{Attn}^{\mathrm{lin}}(Z; W_{k,q,v})$ defined as

$$\mathrm{Attn}^{\mathrm{lin}}(Z; W_{k,q,v}) = W_v Z M (Z^\top W_k^\top W_q Z),$$
$$M = \begin{bmatrix} I_{n\times n} & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)},$$

where W_k, W_q, W_v are the key, query, and value matrices, respectively and the index r below a matrix determines its dimensions. Furthermore, similar to [275], we use mask matrix M in order to avoid the tokens corresponding to (x_i, y_i) to attend the query vector x_q, and combine product of the key and query matrices into $Q = W_k^\top W_q$ to obtain the following parameterization for the attention layer (denoting W_v by P):

$$\mathrm{Attn}^{\mathrm{lin}}(Z; Q, P) = PZM(Z^\top QZ). \tag{B.2}$$

B.3.4 Linear looped Transformer

The linear looped transformer $\mathrm{TF}_L(Z^{(0)}; Q, P)$ can be defined by simply chaining L linear self-attention layers with shared parameters Q and P. In particular, we define

$$Z^{(t)} = Z^{(t-1)} + \frac{1}{n}\mathrm{Attn}^{\mathrm{lin}}(Z; Q, P). \tag{B.3}$$

for all $t > L$. Then, the output of an L layer looped transformer $\mathrm{TF}_L(Z^{(0)}; Q, P)$ just uses the $(d+1, n+1)$ entry of matrix $Z^{(L)}$ i.e.,

$$\mathrm{TF}_L(Z^{(0)}; Q, P) = Z^{(L)}_{d+1, n+1}. \tag{B.4}$$

We note that the minus sign in the final output of the Transformer is only for simplicity of our expositions later on.

Can looped Transformer implement multi-step gradient descent? We first examine the expressivity of looped Transformer. The key idea is to leverage the existing result of one step preconditioned gradient descent from [275] and use the loop structure of looped Transformer to show that it can implement multi-step preconditioned gradient descent. For completeness, we first restate the observation of [275] that linear attention can implement a step of preconditioned gradient descent with arbitrary preconditioner A. For this, it is enough to pick

Proposition 1 (Expressivity; Lemma 1 from [275]) For an appropriate choices of P and Q, the linear looped Transformer from Equation (B.4) implements multiple steps of preconditioned gradient descent on the linear regression instance.

The proof described below is identical to the one from [275]. While this is an expressivity result, our main contribution in later sections is to show convergence to such a solution.

Proof. For Proposition 1, preconditioned gradient descent with preconditioner A can be implemented by setting P and Q to the following:

$$Q = \begin{pmatrix} A_{d\times d} & 0 \\ 0 & 0 \end{pmatrix}; P = \begin{pmatrix} 0_{d\times d} & 0 \\ 0 & 1 \end{pmatrix}. \tag{B.5}$$

Then for the matrix $X = [x_q] \in R^{(d+1)\times(n+1)}$

$$\mathrm{Attn}^{lin}(Z; Q; P) = Z + \frac{1}{n} P Z M (Z^\top Q Z), \tag{B.6}$$

$$= \begin{pmatrix} X & 0_{d\times 1} \\ 0_{1\times n} & 1 \end{pmatrix} + \frac{1}{n} \begin{pmatrix} A\nabla_w \ell(w_0)^\top X & 0_{d\times 1} \\ 0_{1\times n} & 0 \end{pmatrix}, \tag{B.7}$$

$$\mathrm{Attn}^{lin}(Z; Q; P) = \begin{pmatrix} y^\top & 0_{1\times 1} \end{pmatrix}, \tag{B.8}$$

where index $(k:r, \bullet)$ denotes the restriction of the matrix to its rows between k and r, and we used the fact that $\nabla_w \ell(0) = -Xy$. But if we update w with the gradient of $\ell(w)$ preconditioned by A and step size $\frac{1}{n}$ and assuming $w_0 = 0$, then

$$w_1 = w_0 - \frac{1}{n} A \nabla_w \ell(w_0) = 0 - \frac{1}{n} A \nabla_w \ell(0).$$

Plugging this into Equation equation B.7:

$$\mathrm{Attn}^{lin}(Z; Q; P)_{(d+1,\, 1:n)} = w_1^\top X$$

$$\mathrm{Attn}^{lin}(Z; Q; P)_{(d+1,\, n+1)} = w_1^\top x_q.$$

Further, by using Equation equation B.7 in Equation equation B.3 we get

$$Z^{+1}_{(d+1,\, 1:n)} = y^\top - w_1^\top X,$$

$$Z^{+1}_{(d+1,\, n+1)} = -w_1^\top x_q, \quad Z^{+1}_{(1:d,\, \cdot)} = X. \tag{B.9}$$

It is easy to see that Equations equation B.9 hold for all Z^{t} with w_1 substituted by corresponding w_t, thus, allowing implementation of multi-step gradient descent. □

B.3.5 Loss function on the weights

In previous section, while we observed that looped Transformer can implement preconditioned gradient descent, the choice of the preconditioner and its learnability by optimizing a loss function (e.g. squared error loss) still remain unclear. Following [275, 276], we search for the best setting of matrices P, Q, where $Q = \begin{bmatrix} A_{d\times d} & 0 \\ 0 & 0 \end{bmatrix}$, i.e. only the top left $d \times d$ block can be non-zero, and $P = \begin{bmatrix} 0_{d\times d} & 0 \\ u^{\top} & 1 \end{bmatrix}$ for parameter vector $u \in R^d$.

The population squared loss as a function of A and u is

$$\ell = E[(w_{L+1} - X_{TF_L(Z_0;Q,P)})^2].$$

We define a parameter $\gamma = \sqrt{\frac{8Ld}{n}} \cdot (1+2L)$ which governs the accuracy of our estimates, which goes to zero as $n \to \infty$.

Figure B.1: Summary of main theoretical results and key assumptions

B.3.6 Choice of the preconditioner

It is instructive to discuss the choice of the preconditioner A since it determines speed of convergence of w_i to the solution of the regression. Note that the exact solution of an over-determined linear regression instance (X, y) is $w = (X^{\top}X)^{-1}X^{\top}y$. This can be obtained only after one step of preconditioned gradient descent starting from the origin and using inverse of the data covariance matrix preconditioner

$$\frac{1}{n}\sum_{i=1}^{n} x_i x_i^{\top}.$$

In general, it may not be possible to pick the weights of the Transformer to ensure such a preconditioner for all possible regression instances as every instance (X, y) has its own data covariance matrix $\frac{1}{n}XX^\top$. But since x_i's are sampled i.i.d from $N(0, \Sigma)$, it is known that the inverse of the data covariance matrix concentrates around the inverse of the population covariance Σ. Thus, a reasonable choice of A is the inverse of the population covariance matrix Σ^{-1}.

In fact, [275] show that the global minimum of one-layer linear self-attention model under Gaussian data is the inverse of the population covariance matrix plus some small regularization term. However, the characterization of global minimizer(s) of the population loss for the multilayer case is largely missing. Specifically, is there a global optimum for solving regression with Transformers that is close to gradient descent with preconditioner Σ^{-1}? In this work, we solve this open problem for looped Transformers; given that data is sampled iid from $N(0, \Sigma)$, we show that the optimal looped Transformer under constraints stated in Section B.3.5, A^{opt} will be close to Σ^{-1}.

B.3.7 Convergence

In this section, we state our main results. First, we give a tight estimate on the set of global minimizers of the population loss, under the Gaussian assumption, for the looped Transformer model with arbitrary number of loops L.

Theorem 32 (Characterization of the optimal solution). Suppose $(\tilde{A}^{\text{opt}}, u^{\text{opt}})$ are a global minimizer for $L(A, u)$. Then, under condition $\frac{8Ld^2}{n} \le \frac{1}{2^{2L}}$,

1. $L(A^{\text{opt}}, u^{\text{opt}}) \le \frac{8Ld^2 2^{2L}}{n}$
2. $(1-c)\Sigma^{-1} \preceq A^{\text{opt}} \preceq (1+c)\Sigma^{-1}$, $c \le 8d^{1-2L}$ and $u^{\text{opt}} = 0$, where recall $S = \frac{8Ld}{n}$.

Remark 5. From Theorem 32, we first observe that the parameter u has no effect in obtaining a better regression solver and has to be set to zero in the global minimizer. This result was not known in the previous work [275]. A value of $u^{\text{opt}} = 0$ implies that the optimal looped Transformer exactly implements L steps of preconditioned gradient descent, with preconditioner A^{opt}.

Secondly, as discussed in Appendix B.3.6, the choice of preconditioner plays an important role in how fast gradient descent converges to the solution of linear regression. Intuitively the inverse of the population covariance seems like a reasonable choice for a single fixed preconditioner, since it is close to the inverse of the data covariance for all linear regression instances. The above result shows that the global optimum is indeed very close to the inverse of the population covariance.

Precisely how close the optimum is to the population covariance depends on the parameter $\frac{4kd}{n}$, which goes to zero as the number of examples in each prompt goes to infinity. In general, we do not expect the global minimizer to be exactly equal to Σ^{-1}. Indeed for the case of one layer Transformer, which is equivalent to a loop-transformer with looping parameter $L = 1$, the global minimizer found in [275] is not exactly the inverse of the covariance matrix,

but close to it. Even in their case, the distance goes to zero as a. This shows that our estimate in Theorem 32 is essentially the best that one can hope for.

Next, we state our second result, which concerns the convergence of the gradient flow of the loss to the proximity of the global minimizer.

Theorem 33 (Convergence of the gradient flow) Consider the gradient flow with respect to the loss $L(A;0)$ for $\Sigma = I$:

$$\frac{d}{dt}A(t) = -\nabla_A L(A(t);0).$$

Then, for any $C > 4/\alpha^{2L}$, after time $t \geq \frac{1}{\alpha^{L-1}} + \frac{16L}{L-1}\alpha^{-(2L-1)}$ we have

1. $L(A(t)) \leq \varepsilon$,
2. $\|A(t)\| \leq (1+\delta)\|A^{opt}\|$ $\|A\| \geq (1-\delta)\|A^{opt}\|$

Note that the landscape of the loss with respect to A is highly non-convex, hence Theorem 33 does not follow from the typical convex analysis of gradient flows. The key in obtaining this result is that we show a novel gradient dominance condition for the loss with power $2L-1$, which we state next.

Theorem 34 (Gradient dominance) Given $\Sigma = I$, for any A that $L(A;0) \leq \frac{16}{\delta}\left(\frac{d}{n}\right)^L$, we have the following gradient dominance condition:

$$\|\nabla_A L(A;0)\|^2 \geq \frac{1}{16}L(A;0)^{2L-1}.$$

Remark 6. Theorem 34 illustrates that the squared norm of the gradient is at least proportional to the power $2L-1$ of the value of the loss. On the other hand, it is easy to see that the speed of change of the value of the loss on the gradient flow, namely $\frac{d}{dt}L(A(t);0)$, is equal to the squared norm of the gradient. But when the value of the loss is large, then the size of the gradients increase accordingly due to gradient dominance, therefore the convergence is faster when the loss is high. This trend is evident in the rigorous rate that we obtain on the convergence of the gradient flow in Theorem 32.

B.4 Proof Ideas

The proof is structured as follows:

- We obtain closed form formula for the loss function in Lemma 51 in terms of the parameter A and covariance Σ. The loss depends on how close A^{-2} A^{-2} is close to identity for a randomly sampled . Using the estimates in Lemma 53, we obtain an estimate on the loss based on the eigenvalues of the matrix $A^{-2}\Sigma A^{-2}$. Importantly, the result of Lemma 53 is based on estimating the higher moments of the Wishart matrix with arbitrary covariance, shown in Lemma 52 Using our estimate of the loss in Lemma 53, we obtain a precise characterization of the global optimum.

(a) Loss curve (b) Distance to identity

Figure B.2: Measuring the e ect of number of samplesn for looped models trained on inputs of dimensiond 5. Theorem 32 shows that the global optima for looped models A_L is I (since $\Sigma = I$ here) for large enough n. Here we verify that A converges to something very close to I even for smaller values of n and even $n \approx d$.

Figure B.3: Loss trajectory as training proceeds for looped models and baseline multilayer models. Interestingly a 1-layer model looped L times performs similarly to an L layer multilayer model. Furthermore, increasing the number of loops leads to lower loss.

- We further use Lemma 53 to drive an estimate on the magnitude of the gradient based on the same eigenvalues, those of $A^{1/2} \Sigma A^{1/2}$. Comparing this with our estimate for the loss from Lemma 53, we obtain the gradient dominance condition in Theorem 34.
- We use the gradient dominance condition to estimate the speed of convergence of the gradient ow to the proximity of the global minimizer in Theorem 33.

The starting point of the proof is that we can write the loss in a matrix power format based on A when u is set to zero:

Lemma 51. Given $u = 0$, the loss for looped Transformer is as follows:

$$L(A; 0) = E[\operatorname{tr}(I - A^{1/2} \hat{\Sigma} A^{1/2})^{2L}];$$

where $\hat{\Sigma} = \frac{1}{n} \sum_{i=1}^{n} x_i x_i^{\top}$.

To be able to estimate the global minimizers of this loss, rst we need to estimate its value. In particular, we hope to relate the value of the loss to the eigenvalues of A. Note that if the data covariance matrix $\hat{\Sigma}$ was equal to the population covariance matrix Σ, then loss would turn into $\operatorname{tr}(I - A^{1/2} \Sigma A^{1/2})^{2L}$, whose global minimum is $A = \Sigma^{-1}$. However, we can still hope

to approximate the value of $\mathbb{E}[\mathrm{tr}(I - A^{-1/2} \hat{\Sigma} A^{-1/2})^{2\ell}]$ with $\mathrm{tr}(I - A^{-1/2} \Sigma A^{-1/2})^{2\ell}$ given that we have a control on the expectation of the powers of the form $\mathbb{E}[(A^{-1/2} \hat{\Sigma} A^{-1/2} - I)^k]$ for $1 \le k \le 2\ell$. While there are some work on obtaining formulas for the moments of the Wishart matrix (note that A is a Wishart matrix), these formulas [295] are in the form of large summations and do not directly provide closed-form estimates in the general case. In general, the moment of the product of n Gaussian scalar variables can be written as a sum over various allocations of the variables into pairs, then multiplying the covariances of the pairs, due to Isserlis' Theorem. However, this gives a formula in terms of a large summation. Here, we propose a simple combinatorial argument in Lemma 52 which relates the moments of the Wishart matrix to the cycle structure of certain graphs related to the pairings of the Gaussian vectors, while using Isserlis' theorem. In particular, we show the following Lemma which relates the eigenvalues of the moments of the data covariance matrix and the covariance matrix itself:

Lemma 52 (Moment controls). Suppose $\{i > n ; x_i \sim N(0, \Sigma)\}$. Consider the eigen-decomposition $\Sigma = \sum_{i=1}^{d} \lambda_i u_i u_i^T$ with eigenvalues $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_d$. Then, for all $1 \le k \le 2\ell$, $\mathbb{E}[(\frac{1}{n} \sum_{i=1}^{n} x_i x_i^T - I)^k]$ can be written as

$$\mathbb{E}\left[\frac{1}{n} \sum_{i=1}^{n} x_i x_i^T - I\right]^k = \sum_{j=1}^{d} \lambda_j^k \, u_j u_j^T,$$

where for all $1 \le j \le d$:

$$\lambda_j^k = \lambda_j^k + \sum_{1 \le k \le k} \alpha_{n,d,k} \lambda_j^k \le \lambda_j^k + \sum_{1 \le k \le k} \lambda_j^k.$$

Next, we translate this Lemma to a control over the eigenvalues of $A^{-1/2} \mathbb{E}[(I - A^{-1/2} \hat{\Sigma} A^{-1/2})^k]$ with respect to that of $A^{-1/2} \Sigma A^{-1/2}$:

Lemma 53 (Eigenvalue approximation) Given the eigen-decomposition

$$A^{-1/2} \Sigma A^{-1/2} = \sum_{i=1}^{n} \lambda_i u_i u_i^T,$$

then for all $k \le 2\ell$, the matrix $\mathbb{E}[(I - A^{-1/2} \hat{\Sigma} A^{-1/2})^k]$ can be written as

$$\mathbb{E}[(I - A^{-1/2} \hat{\Sigma} A^{-1/2})^k] = \sum_{i=1}^{n} \alpha_i^k \, u_i u_i^T,$$

where

$$\alpha_i^k = (1 - \lambda_i)^k + \sum_{1 \le k \le k} \beta_i^k \le (1 - \lambda_i)^k + \sum_{1 \le k \le k} (1 - \lambda_i)^k.$$

We use Lemma 53 to argue that the matrix $(I - A^{-1/2} \hat{\Sigma} A^{-1/2})^{2\ell}$ for data covariance matrix $\frac{1}{n} \sum_{i=1}^{n} x_i x_i^T$ roughly behaves like $(I - A^{-1/2} \Sigma A^{-1/2})^{2\ell}$, plus some noise on each eigenvalue.

The rest of the proof in a high level goes as follows. Given that the value of the loss has certain amount of sub-optimality gap, we deduce, using Lemma 53, a lower bound on the distance of the eigenvalues of $A^{-1/2} \Sigma A^{-1/2}$ from one in 2ℓ-norm. We then again apply Lemma 53, this

time for power $2L - 1$, which is relevant from the algebraic form of the gradient $\nabla_A L(A; 0)$, to deduce a lower bound for norm of the gradient based on the distance of the eigenvalues of $A^{1/2} A^{1/2}$ to one in the $4L - 2$-norm. Finally by relating these two results using Holder inequality, we obtain the gradient dominance for values of sub-optimality that are not too small.

Note that as in Lemma 53, the magnitude of the noise on all the eigenvalues is controlled by the largest eigenvalue, hence the noise is multiplicative only for the largest eigenvalue. This introduces additional difficulty in arguing about the distance of eigenvalues of A from one given a certain suboptimality gap. Next, using the gradient dominance condition, we estimate the gradient flow ODE and upper bound the value of the loss at a positive time $t_A > 0$ in Theorem 33. To finish the proof of Theorem 33, we need to translate a small suboptimality gap into closeness to global optimum, which we prove the following Theorem:

Theorem 35 (Small loss implies close to optimal) For $A > 4$, if $L(A; 0) \le B^{2L-2}$, then for $c(4 + 16\log(1/\epsilon))^{1/(2L)}$

$$(1 - c\epsilon)A^{opt} \preceq A \preceq (1 + c\epsilon)A^{opt}.$$

B.4.1 Out of distribution generalization

In the result below, we show that a looped Transformer learned on one in-context distribution can generalize to other problem instances with different covariance, owing to the fact that it has learned a good iterative algorithm.

Theorem 36. Let A^{opt}, u^{opt} be the global minimizers of the poplulation loss for looped Transformer with depth L when the in-context input $x_i \sim N(0, \Sigma)$ are sampled from $N(0, \Sigma)$ and w^* is sampled from $N(0, \Sigma^{-1})$. Suppose we are given an arbitrary linear regression instance $I^{out} = \{x_i^{out}, y_i^{out}\}_{i=1}^n, w^{out,*}$ with input matrix $X^{out} = [x_1^{out}, \ldots, x_n^{out}]$, query vector x_q^{out}, and label $y_q^{out} = w^{out,*} x_q^{out}$. Then, if for parameter $0 < \epsilon < 1$, the input covariance matrix $\Sigma^{out} = X^{out} X^{out\top}$ of the out of distribution instance satisfies

$$\|\Sigma^{-1/2}(\Sigma^{out} - \Sigma)\Sigma^{-1/2}\| \le \epsilon, \tag{B.10}$$

we have the following instance-dependent bound on the out of distribution loss:

$$\begin{aligned} &TF_L(Z_0^{out}; Q, P) - y_q^{out})^2 \\ &\le (1 + 16\epsilon(4 + 16\log(1/\epsilon)))^2 - 1 + 16\epsilon(4 + 16\log(1/\epsilon)) + \epsilon)^{2L} \\ &\quad \|x_q^{out}\|_\Sigma^2 \|w^{out,*}\|_\Sigma^2. \end{aligned}$$

B.5 Experiments

In this section we run experiments on in-context learning linear regression to validate the theoretical results and to go beyond them. In particular, we test if looped models can indeed be trained to convergence, as the theory suggests, and whether the learned solution is close

Figure B.4: In-context linear regression loss on in-distribution (left) and out-of-distribution (right) data which is sampled using a di erent covariance x I . For looped models trained with just few loops (2, or 5), evaluating with more loops keeps improving the loss in both cases, suggesting that it learned the correct iterative algorithm.

Figure B.5: The iterate A converges to identity for all number of loops. The converge is slower for large number of loops which is also observed by our rate of convergence in Theorem 33. Interestingly just training with 1 loop does not converge in this setup.

to the predicted global minima. Furthermore, we investigate the e ect of various factors such number of loops, number of in-context samples and depth of the model (in the multi-layer case). We use the codebase and experimental setup from [275] for all our linear regression experiments. In particular we work with d 10 dimensional inputs and train with L attention layer models for multilayer training and 1 layer attention model looped times. Inputs and labels are sampled exactly based on the setup from Appendix B.3.2, using covariance I .

B.5.1 E ect of loops

We rst test whether training with looped model converges to a low loss, and how small the loss can be made with more loops. In Figure B.3, we see that looped models indeed converge to very small loss very quickly, and higher loops leads to lower loss as expected. Interestingly, we nd that a 1-layer model looped L times roughly has very loss to an L -layer non-looped model.

B.5.2 E ect of in-context samples

Theorem 33 shows convergence of the gradient ow for looped models when the number of in-context samples, n, is large compared to the dimension d. In these experiments we test the convergence of loss and iterate for smaller values of n, when n is closer to, or even smaller than d.

In Figure B.2 we observe that the loss converges for all values of A and the iterates also converge to a value very close identity. Theoretically proving this result remains an open question.

B.5.3 Out-of-distribution evaluation

While the looped model was trained with linear regression instances with identity covariance, we evaluate the trained looped model on out-of-distribution (OOD) data with a different covariance $\Sigma \neq I$. Theorem 36 predicts that the model trained on identity covariance should also generalize to other covariances, because it simulates multi-step preconditioned gradient descent that works for all problems instances. In Figure B.4, we find that the learned looped model achieves small loss for OOD data, although the scale of the loss is higher than in-distribution (ID) data. Interestingly, for looped models trained with just 2 (or 5) loops, evaluating them with arbitrarily large number of loops during test time continues to decrease the loss even further for ID and OOD data. This suggests that the trained looped models are indeed learning a good iterative algorithm.

B.6 Conclusion

This work provides the first convergence result showing that attention based models can learn to simulate multi-step gradient descent for in-context learning. The result not only demonstrates that Transformers can learn interpretable multi-step iterative algorithms (gradient descent in this case), but also highlights the importance of looped models in understanding such phenomena. There are several open questions in this space including understanding the landscape of the loss, convergence of training without weight sharing across layers, and handling of non-linearity in the attention layers. It is also interesting to understand the empirical phenomenon that looping the trained models beyond the number of loops used in training can continue to improve the test loss. One way to show this is by obtaining a tighter upper bound on the optimal loss value.

Appendix A

Code listing

This example uses the listings package.

```
function print_rate(kappa,xMin,xMax,npoints,option)
      local c = 1-kappa*kappa
      local croot = (1-kappa*kappa)^(1/2)
      local logx = math.log (xMin)
      local psi = 0

      local xstep = ( math.log (xMax)- math.log (xMin))/(npoints-1)

      arg0 = math.sqrt (xMin/c)
      psi0 = (1/c)* math.exp((kappa*arg0)^2)*(erfc(kappa*arg0)-erfc(
         arg0))

      if option~=[[]] then
        tex.sprint("\\addplot+["..option.."] coordinates{")
        -- addplot+ for color cycle to work
      else
        tex.sprint("\\addplot+ coordinates{")
      end
      tex.sprint("("..xMin..","..psi0..")")

      for i=1, (npoints-1) do
        x = math.exp(logx + xstep)
        arg = math.sqrt (x/c)
        karg = kappa*arg
        if karg<5 then
      -- this break compensates for exp(karg^2), which multiplies the
          error in the erf approximation...
           logpsi = - math.log (croot) + karg^2 + math.log (erfc(karg)-
              erfc( arg))
           psi = math.exp(logpsi)
```

```
28          else
29             psi = (1/(karg) - 1/(2*(karg^3)) + 3/(4*(      arg^5)) )/(1
                  .77245385*croot)
30             -- this is the large x asymptote of the reaction rate
31          end
32          logx =  math.log (x)
33          tex.sprint("("..x..","..psi..")")
34       end
35       tex.sprint("}")
36 end
37 \end{luacode*}
```

Appendix B

One-term coe cients for heat conduction

B.1 A multipage table of numbers

This example uses the longtable package: $A_1 f_1 \exp\hat{}\ {}^2_1 Fo\bullet, \bar{}\ D_1 \exp\hat{}\ {}^2_1 Fo\bullet$.

Table B.1: One-term coe cients for one-dimensional heat conduction with a convective boundary condition. Data follow H. D. Baehr and K. Stephan [296].

Bi	Plate			Cylinder			Sphere		
	$_1$	A_1	D_1	$_1$	A_1	D_1	$_1$	A_1	D_1
0:01	0.09983	1.0017	1.0000	0.14124	1.0025	1.0000	0.17303	1.0030	1.0000
0:02	0.14095	1.0033	1.0000	0.19950	1.0050	1.0000	0.24446	1.0060	1.0000
0:03	0.17234	1.0049	1.0000	0.24403	1.0075	1.0000	0.29910	1.0090	1.0000
0:04	0.19868	1.0066	1.0000	0.28143	1.0099	1.0000	0.34503	1.0120	1.0000
0:05	0.22176	1.0082	0.9999	0.31426	1.0124	0.9999	0.38537	1.0150	1.0000
0:06	0.24253	1.0098	0.9999	0.34383	1.0148	0.9999	0.42173	1.0179	0.9999
0:07	0.26153	1.0114	0.9999	0.37092	1.0173	0.9999	0.45506	1.0209	0.9999
0:08	0.27913	1.0130	0.9999	0.39603	1.0197	0.9999	0.48600	1.0239	0.9999
0:09	0.29557	1.0145	0.9998	0.41954	1.0222	0.9998	0.51497	1.0268	0.9999
0:10	0.31105	1.0161	0.9998	0.44168	1.0246	0.9998	0.54228	1.0298	0.9998
0:15	0.37788	1.0237	0.9995	0.53761	1.0365	0.9995	0.66086	1.0445	0.9996
0:20	0.43284	1.0311	0.9992	0.61697	1.0483	0.9992	0.75931	1.0592	0.9993
0:25	0.48009	1.0382	0.9988	0.68559	1.0598	0.9988	0.84473	1.0737	0.9990
0:30	0.52179	1.0450	0.9983	0.74646	1.0712	0.9983	0.92079	1.0880	0.9985
0:40	0.59324	1.0580	0.9971	0.85158	1.0931	0.9970	1.05279	1.1164	0.9974
0:50	0.65327	1.0701	0.9956	0.94077	1.1143	0.9954	1.16556	1.1441	0.9960
0:60	0.70507	1.0814	0.9940	1.01844	1.1345	0.9936	1.26440	1.1713	0.9944
0:70	0.75056	1.0918	0.9922	1.08725	1.1539	0.9916	1.35252	1.1978	0.9925
0:80	0.79103	1.1016	0.9903	1.14897	1.1724	0.9893	1.43203	1.2236	0.9904
0:90	0.82740	1.1107	0.9882	1.20484	1.1902	0.9869	1.50442	1.2488	0.9880

Table B.1: (continued)

Bi	Plate			Cylinder			Sphere		
	$_1$	A_1	D_1	$_1$	A_1	D_1	$_1$	A_1	D_1
1:00	0.86033	1.1191	0.9861	1.25578	1.2071	0.9843	1.57080	1.2732	0.9855
1:10	0.89035	1.1270	0.9839	1.30251	1.2232	0.9815	1.63199	1.2970	0.9828
1:20	0.91785	1.1344	0.9817	1.34558	1.2387	0.9787	1.68868	1.3201	0.9800
1:30	0.94316	1.1412	0.9794	1.38543	1.2533	0.9757	1.74140	1.3424	0.9770
1:40	0.96655	1.1477	0.9771	1.42246	1.2673	0.9727	1.79058	1.3640	0.9739
1:50	0.98824	1.1537	0.9748	1.45695	1.2807	0.9696	1.83660	1.3850	0.9707
1:60	1.00842	1.1593	0.9726	1.48917	1.2934	0.9665	1.87976	1.4052	0.9674
1:70	1.02725	1.1645	0.9703	1.51936	1.3055	0.9633	1.92035	1.4247	0.9640
1:80	1.04486	1.1695	0.9680	1.54769	1.3170	0.9601	1.95857	1.4436	0.9605
1:90	1.06136	1.1741	0.9658	1.57434	1.3279	0.9569	1.99465	1.4618	0.9570
2:00	1.07687	1.1785	0.9635	1.59945	1.3384	0.9537	2.02876	1.4793	0.9534
2:20	1.10524	1.1864	0.9592	1.64557	1.3578	0.9472	2.09166	1.5125	0.9462
2:40	1.13056	1.1934	0.9549	1.68691	1.3754	0.9408	2.14834	1.5433	0.9389
2:60	1.15330	1.1997	0.9509	1.72418	1.3914	0.9345	2.19967	1.5718	0.9316
2:80	1.17383	1.2052	0.9469	1.75794	1.4059	0.9284	2.24633	1.5982	0.9243
3:00	1.19246	1.2102	0.9431	1.78866	1.4191	0.9224	2.28893	1.6227	0.9171
3:50	1.23227	1.2206	0.9343	1.85449	1.4473	0.9081	2.38064	1.6761	0.8995
4:00	1.26459	1.2287	0.9264	1.90808	1.4698	0.8950	2.45564	1.7202	0.8830
4:50	1.29134	1.2351	0.9193	1.95248	1.4880	0.8830	2.51795	1.7567	0.8675
5:00	1.31384	1.2402	0.9130	1.98981	1.5029	0.8721	2.57043	1.7870	0.8533
6:00	1.34955	1.2479	0.9021	2.04901	1.5253	0.8532	2.65366	1.8338	0.8281
7:00	1.37662	1.2532	0.8932	2.09373	1.5411	0.8375	2.71646	1.8673	0.8069
8:00	1.39782	1.2570	0.8858	2.12864	1.5526	0.8244	2.76536	1.8920	0.7889
9:00	1.41487	1.2598	0.8796	2.15661	1.5611	0.8133	2.80443	1.9106	0.7737
10:00	1.42887	1.2620	0.8743	2.17950	1.5677	0.8039	2.83630	1.9249	0.7607
12:00	1.45050	1.2650	0.8658	2.21468	1.5769	0.7887	2.88509	1.9450	0.7397
14:00	1.46643	1.2669	0.8592	2.24044	1.5828	0.7770	2.92060	1.9581	0.7236
16:00	1.47864	1.2683	0.8541	2.26008	1.5869	0.7678	2.94756	1.9670	0.7109
18:00	1.48830	1.2692	0.8499	2.27556	1.5898	0.7603	2.96871	1.9734	0.7007
20:00	1.49613	1.2699	0.8464	2.28805	1.5919	0.7542	2.98572	1.9781	0.6922
25:00	1.51045	1.2710	0.8400	2.31080	1.5954	0.7427	3.01656	1.9856	0.6766
30:00	1.52017	1.2717	0.8355	2.32614	1.5973	0.7348	3.03724	1.9898	0.6658
35:00	1.52719	1.2721	0.8322	2.33719	1.5985	0.7290	3.05207	1.9924	0.6579
40:00	1.53250	1.2723	0.8296	2.34552	1.5993	0.7246	3.06321	1.9942	0.6519
50:00	1.54001	1.2727	0.8260	2.35724	1.6002	0.7183	3.07884	1.9962	0.6434
60:00	1.54505	1.2728	0.8235	2.36510	1.6007	0.7140	3.08928	1.9974	0.6376
80:00	1.55141	1.2730	0.8204	2.37496	1.6013	0.7085	3.10234	1.9985	0.6303
100:00	1.55525	1.2731	0.8185	2.38090	1.6015	0.7052	3.11019	1.9990	0.6259
200:00	1.56298	1.2732	0.8146	2.39283	1.6019	0.6985	3.12589	1.9998	0.6170
[a]	1.57080	1.2732	0.8106	2.40483	1.6020	0.6917	3.14159	2.0000	0.6079

Appendix C

Computing Optimal Regularizers for Online Linear Optimization

C.1 Additional Related Work

Many modern learning algorithms are actually variants of mirror descent / FTRL [297 302]. Recently, [303] used a variant of mirror descent to solve in nite-horizon MDPs, achieving linear runtime in the number of samples. [304] extended mirror descent to optimize convex functionals on an in nitesimal space, demonstrating that the primal iterations of Sinkhorn's algorithm for entropic optimal transport in a continuous domain are an instance of mirror descent. [305] studied alternating mirror descent for two-player bilinear zero-sum games, proving a regret bound ofO^T$^{1-3}$•. [306] discusses the equivalence of FTPL (Follow-The-Perturbed-Leader) and FTRL. [307] presents scale-free generalizations of FTRL algorithms (with regret bounds depending on the total magnitude of the losses instead of the time horizon).

Mirror descent has also been used in the context of stochastic optimization [308]. Authors in [309] study mirror descent for composite loss functions under both stochastic and online settings. [310] relaxed the subgradient boundedness condition from [309] and extended their analysis to examine the generalization performance of multi-pass SGD in non-parametric settings. [311 313] applied mirror descent to address the problem of online linear optimization with bandit feedback. [314] introduced a novel interpretation of mirror descent as optimizing a dual-based lower bound for the objective. Building on this perspective, they proposed a coupling between mirror descent and gradient descent that achieves an accelerated convergence rate. [315, 316] applied mirror descent in distributed settings. [317] utilized mirror descent for a constrained online revenue maximization problem with unknown parameters. Authors in [318 320] employ mirror descent for online resource allocation problems. Mirror descent has also been instrumental in primal-dual methods for solving structured saddle-point problems [321 326].

C.2 The ideal regularizer and proving stronger martingale type inequalities for $p \geq 2$

Here, we state the existence of an ideal regularizer such that running FTRL with this regularizer achieves the optimal rate up to a constant. This result is adapted from [17], except that they prove the same regularizer results in a regret bound which is $o(\cdot)$ by a logarithmic factor of $\log(T)$; this log factor is indeed not desirable for our purpose as we are interested in long time horizon regimes when T can potentially be exponentially large in dimension. Our contribution here is that we improve the result of [17] for $p \geq 2$ case and remove this log factor. We further demonstrate a continuity condition for this ideal regularizer that we use for our smoothing arguments in Section C.3.

First, we state the result of [25] and [327] that we build upon; it is known [25, 327] that the optimal rate for adversarial online linear optimization translates into a property on the growth of the norm $\|\cdot\|_{X^c}$ of an arbitrary Rademacher martingale sequence. We state this property rigorously in Theorem 37, which is stated as Theorem 4 in [17].

Theorem 37 (Restatement of Theorem 4 in [17]) Given the optimal rate for online linear optimization with action and loss sets $X, L \subset \mathbb{R}^d$ is $O(C\sqrt{T})$, then for a Rademacher random vector $\epsilon = (\epsilon_1, \ldots, \epsilon_n)$ and any sequence of functions $x_i(\epsilon_1, \ldots, \epsilon_{i-1}) : \{\pm 1\}^{i-1} \to \mathbb{R}^d$, where x_i is a function of the first $i-1$ coordinates in ϵ, we have

$$\mathbb{E}\left[\sum_i \epsilon_i x_i(\epsilon)\right] \leq 2C \sup_{0 \leq i \leq n} \sup \|x_i(\epsilon)\|_L. \tag{C.1}$$

The main contribution of authors in [17] is that they translate equation C.1 to the existence of a suitable barrier for mirror descent. In particular, they prove the following key Lemmas 54, 57. We start with Lemma 54 which translates property equation C.1 to a more refined argument about the growth of martingale norms that are defined based on the action and loss sets.

Lemma 54 (Restatement of Lemma 12 in [17] for $r = 2$). For $1 < r \leq 2$, if there exists a constant $C > 0$ such that for any natural number n and any sequence of mappings $\{x_i(\cdot)\}_{i=1}^n$, $x_i : \{\pm 1\}^{i-1} \to \mathbb{R}^d$ and Rademacher random vector $\epsilon = (\epsilon_1, \ldots, \epsilon_n)$ satisfy

$$\mathbb{E}\left[\sum_{i=1}^n \epsilon_i x_i(\epsilon)\right] \leq 2C n^{1-\frac{1}{r}} \sup_{0 \leq i \leq n} \sup \|x_i(\epsilon)\|_L,$$

then for $p > r$ and $p \geq \frac{20C}{r-1}$, for any sequence $\{x_i(\cdot)\}_{i=1}^n$ as described above, we have the following inequality:

$$\mathbb{E}\left[\sum_{i=1}^n \epsilon_i x_i(\epsilon)\right] \leq 8\sqrt{p} \sup\left(\mathbb{E}\left[\sum_i \|x_i(\epsilon)\|_L^p\right]\right)^{\frac{1}{p}}. \tag{C.2}$$

The next Lemma states how authors in [17] translate the property in Equation equation C.2 to the existence of the ideal regularizer:

Lemma 55 (Restatement of Lemma 11 in [17]) For constant $\tilde{C}$, the following statements are equivalent:

1. For all n and sequence of mappings $\{x_i(\cdot)\}_{i=0}^n$ where $x_i : \{-1,1\}^i \to \mathbb{R}^d$:

$$\mathbb{E}\left[\sup_{w\in W}\left\{\sum_{i=1}^n \langle w, x_i(\epsilon)\rangle - \frac{1}{p}\|w\|^p\right\}\right] \le \sup_{w\in W}\left\{\langle w, \sum_{i=1}^n \mathbb{E}[x_n(\epsilon)]\rangle - \frac{1}{p}\|w\|^p\right\},$$

2. There exists a 2-homogeneous non-negative convex function f_0 on $\mathbb{R}^d$ which is 1-strongly convex w.r.t. $\|\cdot\|_W$ and $\forall x, \frac{1}{q}\|x\|_W^q \le f_0(x) \le \frac{C^q}{q}\|x\|_X^q$, where $\frac{1}{p} + \frac{1}{q} = 1$.

The existence of such regularizer from Lemma 55 then implies a $CT^{1-\frac{1}{p}}$ regret bound for FTRL. Nonetheless, the reason they end up with a $\log(T)$ factor in the regret is that they need to use Lemma 54 with a power of p slightly less than two, as the constant p reciprocally depends on $2-p$, so p has to be $1-\log(T)^{-1}$ less than 2. We improve Lemma 54 in Lemma 56 below, for the case of $p = 2$, and shave off the p factor which is causing the additional $\log(T)$. This enables us to show a tighter upper bound for the regularizer on domain X in Theorem 38.

khasha: change w's below to x maybe not

Lemma 56 (Improving the Martingale Type for $p = 2$). Suppose for the norm $\|\cdot\|_X$ we have

$$\mathbb{E}\left[\|x_0 + \sum_{i=1}^n \epsilon_i x_i(\epsilon)\|_X^2\right] \le D^2 n^{1-2} \sup_{0<i\le n} \sup_{\epsilon} \|x_i(\epsilon)\|_W^2, \tag{C.3}$$

for arbitrary vector valued functions $x_n : \{-1,1\}^{n-1} \to \mathbb{R}^d$ and Rademacher sequence $\{\epsilon_i\}_{i=1}^n$, $\epsilon_i = \pm 1$. Then, we have

$$\mathbb{E}\left[\|x_0 + \sum_{i=1}^n \epsilon_i x_i(\epsilon)\|_X^{1-2}\right] \le D^2 C^2 \left(\sum_{i=1}^n \|x_i(\epsilon)\|_W^2\right)^{1-2}.$$

Proof. First, note that if we average equation C.3 over x_0 and $-x_0$ and extend the functions $x_i(\cdot)$ to also depend on a Rademacher variable ϵ_0 at time zero, then we get

$$\mathbb{E}\left[\|\sum_{i=0}^n \epsilon_i x_i(\epsilon)\|_X^2\right] \le D^2 n^{1-2} \sup_{0<i\le n} \sup_{\epsilon} \|x_i(\epsilon)\|_W^2. \tag{C.4}$$

Now let $c_i = \|x_i\|_W$. Take a fresh rademacher sequence $\{\epsilon_j\}_{j=1}^a$. We will define the sequence $\{\epsilon_i\}_{i=1}^n$ based on the randomness of ϵ_j's: define $\epsilon_i = 1$ if $\sum_{j=t_{i-1}+1}^{t_i-1} \epsilon_j \le C\frac{\|x_i\|}{D}$ and $\epsilon_i = -1$ if $\sum_{j=t_{i-1}+1}^{t_i-1} \epsilon_j \ge -C\frac{\|x_i\|}{D}$. From symmetry, it is easy to check that ϵ_i's are indeed i.i.d distributed uniformly on $\{-1,1\}$. Next, for a given positive $A > 0$, define the sequence of indices $\{t_i\}_{i=1}^n$ and the alternative sequence $\{x_i\}_{i=0}^m$ such that for all i, $x_{t_i} = x_{t_{i-1}} = \cdots = x_{t_{i-1}+1} = \frac{x_i}{\|x_i\|_W}$, and t_i is the first index such that $\left|\sum_{j=t_{i-1}+1}^{t_i-1} \epsilon_j\right| \ge C\frac{\|x_i\|}{D}$. Now from this definition. we have that x_i's satisfy

$$\left\|x_i + \sum_{j=t_{i-1}+1}^{t_i-1} x_j\right\|_X = \left\|\frac{x_i}{\|x_i\|_W}\right\|_X. \tag{C.5}$$

But for $t_{sum} = \sum_{i=0}^{n} t_i$ equation C.5 implies:

$$\left| \sum_{i=0}^{n} x_i \cdot \sum_{j=0}^{t_{sum}} \gamma_j x_j \right| \leq \left| n \cdot \max_{i=0}^{n} \left| \frac{x_i}{\|x_i\|_L} \right| \right|_{X^c} .$$

The key observation is for all $i > n$, the distribution of t_i is sub-exponential and the sum concentrates around its expectation. In particular,

$$P\left(t_i \geq \left\| x_i \right\|_L^2 \right) \leq B e^{-\cdot} . \tag{C.6}$$

It is su cient for us to show that the sum $\sum_{i=1}^{n} t_i$ is at most $O\left(\sum_{i=1}^{n} \left\| x_i \right\|_L^2 \right)$ with at least constant probability p. Call this event E. First, we use Chebyshev inequaility to show $P(E) \geq 1/2$. Note that Equation equation C.6 imlies

$$E t_i^2 \leq O\left(\left\| x_i \right\|_L^4 \right) ;$$

which implies

$$Var\left(\sum_i t_i \right) \leq O\left(\sum_i \left\| x_i \right\|_L^4 \right) .$$

Therefore, from Chebyshev inequality

$$P\left(\sum_{i=1}^{n} t_i \geq \sum_{i=1}^{n} \left\| x_i \right\|_L^2 - \sqrt{2} \sqrt{\sum_{i=1}^{n} \left\| x_i \right\|_L^4} \right) \geq B \frac{1}{2} ;$$

which implies

$$P\left(\sum_{i=1}^{n} t_i \geq \frac{1}{2} \sum_{i=1}^{n} \left\| x_i \right\|_L^2 \right) \geq B \frac{1}{2} ;$$

hence we showed that E happens with at least constant probability. Furthermore, It is easy to check that conditioned on E, ε_i's are still Rademacher variables. On the other hand, using equation C.4 for sequence $\{x_i\}$ and $m = \sum_{i=0}^{n} \left\| x_i \right\|_L^2$:

$$E\left| \sum_{j=1}^{m} \gamma_j x_j \right|_{X^c} \leq B D m^{1/2} \sup_{0 \leq i \leq n} \sup \left\| x_i \right\|_L . \tag{C.7}$$

but from positivity of norm khasha: explain the thir equality below is because there we don't care about the size of $_i$

$$E\Big[\sum_{j=1}^{m} \sim_j x_j \hat{} \bullet\Big] \quad C E\Big\{\sum_{j=1}^{m} \sim_j x_j \hat{} \bullet\Big] \quad U E\Big\} P \hat{} E\bullet$$

$$C E\Big[\sum_{i=1}^{n} {}_i x_i \hat{} \bullet\Big] \quad U E \quad P \hat{} E\bullet \quad \hat{} n \quad 1\bullet \quad \max_{i\ 0}\Big[\frac{x_i}{Y x_i Y_L}\Big]$$

$$E\Big[\sum_{i=1}^{n} {}_i x_i \hat{} \bullet\Big] \quad P \hat{} E\bullet \quad \hat{} n \quad 1\bullet \quad \max_{i\ 0}\Big[\frac{x_i}{Y x_i Y_L}\Big]$$

$$C\frac{1}{2} E\Big[\sum_{i=1}^{n} {}_i x_i \hat{} \bullet\Big] \quad \hat{} n \quad 1\bullet \quad \max_{i\ 0}\Big[\frac{x_i}{Y x_i Y_L}\Big] \quad :$$

Note that the equality in the third line above is because the size of $_i$'s is independent of 's. Plugging this back into equation C.7

$$E\Big[\sum_{i=1}^{n} {}_i x_i \hat{} \bullet\Big] \quad B \quad \Big\{ D \sum_{i=0}^{n} Y x_i Y_L^2 \bullet \quad \hat{} n \quad 1\bullet \quad \max_{i\ 0}\Big[\frac{x_i}{Y x_i Y_L}\Big] \quad :$$

Sending $\quad 0$ nishes the proof. □

Next we state and prove Lemma 57. This Lemma in similar to Lemma 57 for the case 2, i.e. it translates the margtingale property to the existence of an ideal regularizer, except that we show an additional useful Lipschitz property for the regularizer which we use for smoothing the regularizer in Section C.3. The proof of Theorem 38 directly follows from combining Lemmas 57 and 56.

Lemma 57 (Martingale type ideal regularizer) For constant C, the following statements are equivalent:

1. For all n and sequence of mappings $x_i \bullet_{i\ 0}^{n}$ where $x_i \sim \bullet_{i\ 1} R^d$:

$$E\Big[x_0 \sum_{i=1}^{n} {}_i x_i \hat{} \bullet\Big]^2 \quad B C^2 \Big(Y x_0 Y_L^2 \sum_{i=1}^{n} E Y x_n \hat{} \bullet Y_L^2 \Big)$$

2. There exists a 2-homogeneous non-negative convex function on R^d which is -strongly convex w.r.t $Y:Y_{L^c}$ and $\forall x; \frac{1}{2} Y x Y_{L^c}^2 \quad B \quad f_0 \hat{} x\bullet \quad B \quad \frac{C^2}{2} Y x Y_X^2$. Furthermore, f is Lipschitz continuous as

$$\$ f_0 \hat{} x_1\bullet \quad f_0 \hat{} x_2\bullet S B C^2 Y x_1 \quad x_2 Y_X \hat{} Y x_1 Y_X \quad Y Y x_2 Y Y_X \bullet :$$

Proof. This is Lemma 11 in [17], except that we are claiming an additional Lipschitz continuity here for f_0, which we need to show regularity properties for the gaussian smoothed function later on. To show the Lipschitz continuity, we note that from the proof of Lemma 11

in [17], f_0 is defined as the Fenchel dual of a barrier $f_0^\ddagger$, i.e. $f_0(x) = \sup_z \langle x, z\rangle - f_0^\ddagger(z)$, where $\frac{1}{C^2}\|x\|_{X^c} \le f_0^\ddagger(x) \le \|x\|_{L^c}^2$. Therefore, defining $z(x) = \arg\max_z \langle x, z\rangle - f_0^\ddagger(z)$, we have

$$0 \le f_0(z(x)) \le \|x\|_X \|z(x)\|_{X^c} - \frac{1}{C^2}\|z(x)\|_{X^c}^2,$$

which implies

$$C^2 \|x\|_X \ge \|z(x)\|_{X^c}.$$

Therefore, for $x_1, x_2 \in X$ we have

$$\begin{aligned} f_0(x_1) - \langle x_1, z(x_2)\rangle &\le f_0^\ddagger(z(x_2)) - \langle x_2, z(x_2)\rangle \le f_0^\ddagger(z(x_2)) - \|x_1 - x_2\|_X \|z(x_2)\|_{X^c} \\ f_0(x_2) &\le C^2 \|x_1 - x_2\|_X + \|x_1\|_X \|x_2\|_X. \end{aligned}$$

Noting the reverse symmetric inequality $f_0(x_2) \le f_0(x_1) + C^2\|x_1 - x_2\|_X + \|x_1\|_X\|x_2\|_X$ completes the proof. □

C.3 Smoothing the Regularizer

The goal of this section is to show the existence of a regularizer which enables FTRL to achieve the optimal regret for arbitrary pair (X, L) of action and loss sets which also has smooth derivatives. We achieve this by using Gaussian smoothing of the regularizer from [17]. First, we state Theorem <empty citation> in which we prove that FTRL with this regularizer indeed achieves the optimal rate $O(\mathrm{Rate}(X, L)\sqrt{T})$; note that this is a $\log(T)$ improvement over the result of [17], and in addition the regularizer satisfies a desirable Lipschitz property. We then proceed to smooth this regularizer by adding Gaussian noise and showing the smoothness properties we want.

Theorem 38 (Existence of an ideal regularizer for mirror descent) There exists a 2-homogeneous continuous regularizer $f_0 : \mathbb{R}^d \to \mathbb{R}$ which satisfies

1. $\max_{x \in X} f_0(x) \le \mathrm{Rate}(X, L)^2$
2. f_0 is 1-strongly convex w.r.t. $\|\cdot\|_{L^c}$ on X, where $\|\cdot\|_{L^c}$ is the dual norm of $\|\cdot\|_L$.
3. f_0 satisfies the following Lipschitz continuity condition: $\forall x_1, x_2$:

$$f_0(x_1) - f_0(x_2) \le \mathrm{Rate}(X, L)^2 \|x_1 - x_2\|_X \left(\|x_1\|_X + \|x_2\|_X + \|x_1 - x_2\|_X\right).$$

Proof. Directly from the relation between optimal rate of online optimization and Equation equation C.1, which we state in Theorem 37, with Lemmas 57 and 56. □

For the regularizer f_0 given by Theorem 38, we define the Gaussian smoothed function $f : \mathbb{R}^d \to \mathbb{R}$:

$$f(x) = E_{y \sim N(x, \sigma^2 I)} f_0(y). \tag{C.8}$$

We start by showing that strong convexity property with respect to arbitrary norms is inherited for f_0 to f.

Lemma 58 (Strong convexity of the smoothed function) If f_0 is strongly convex w.r.t $\|\cdot\|_{L_c}$, the f is also strong convex w.r.t $\|\cdot\|_{L_c}$.

Proof. From strong convexity of f, for $0 \leq \lambda \leq 1$ we have

$$f_0(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f_0(x_1) + (1-\lambda) f_0(x_2) - \frac{\lambda(1-\lambda)}{2}\|x_1 - x_2\|^2.$$

Now consider the gaussian random variable $N(0, \sigma^2 I)$ and write $f(x_1) = E f(x_1 + \cdot)$, $f(x_2) = E f(x_2 + \cdot)$. Then

$$\begin{aligned} f(\lambda x_1 + (1-\lambda)x_2) &= E f_0(\lambda x_1 + (1-\lambda)x_2 + \cdot) \\ &= E f_0(\lambda(x_1 + \cdot) + (1-\lambda)(x_2 + \cdot)) \\ &\leq \lambda E f_0(x_1 + \cdot) + (1-\lambda) E f_0(x_2 + \cdot) - \frac{\lambda(1-\lambda)}{2}\|x_1 - x_2\|^2_{L_c} \\ &= \lambda f_0(x_1) + (1-\lambda) f_0(x_2) - \frac{\lambda(1-\lambda)}{2}\|x_1 - x_2\|^2_{L_c}. \end{aligned}$$

□

Lemma 59 (Strong convexity $\Leftrightarrow$ Hessian lower bound) If f is twice continuously differentiable and strongly convex with respect to $\|\cdot\|_{L_c}$, then for its hessian at arbitrary point x and arbitrary direction v we have

$$v^T \nabla^2 f(x) v \geq c\|v\|^2_{L_c}. \tag{C.9}$$

Proof. From Taylor series around x_1 at points x_2 and $\lambda x_1 + (1-\lambda)x_2$:

$$f(x_2) = f(x_1) + \nabla f(x_1)(x_2 - x_1) + \frac{1}{2}(x_2 - x_1)^T \nabla^2 f(x_1)(x_2 - x_1) + o(\|x_2 - x_1\|^2),$$

$$f(\lambda x_1 + (1-\lambda)x_2) = f(x_1) + \nabla f(x_1)((1-\lambda)(x_2 - x_1)) + \frac{1}{2}(1-\lambda)^2 (x_2 - x_1)^T \nabla^2 f(x_1)(x_2 - x_1) + o(\|x_2 - x_1\|^2).$$

Therefore

$$f(x_2) - f(x_1) \geq f(\lambda x_1 + (1-\lambda)x_2) - \frac{1}{2}(1-\lambda)^2 (x_2 - x_1)^T \nabla^2 f(x_1)(x_2 - x_1) - o(\|x_2 - x_1\|^2).$$

Therefore, strong convexity is equivalent to equation C.9 for all directions v. □

Lemma 60 (Norm and norm squared Gaussian integral) Given a two-homogeneous function f_0 satisfying 1 and $\max_{\|x\|=1} f_0(x) = B \leq C^2$, then for f defined in equation C.8

$$f(x) \leq B\frac{C^2}{r^2}\sigma^2 d + C^2\|x\|^2_X,$$

$$E_{y \sim N(x, \sigma^2 I)} \|f_0(y)\|^2 \leq B^2 C^4 (\|x\|^4_X + \frac{4}{r^4}\sigma^4 d^4).$$

Proof. Note that from the property (1) in Theorem 38 and the 2-homogeneity of f_0, we have for all $y \in R^d$, $f_0(y) \le C^2 \|y\|_X^2$. Now using triangle inequality and Lemma 61, we can write

$$\begin{aligned}
\|f(x)\| &= \|E_{y \sim N(x, I)} f_0(y)\| \le E\|f_0(y)\| \\
&\le E C^2 \|y\|_X^2 \\
&\le E C^2 \|y - x\|_X^2 + C^2 \|x\|_X^2 \\
&\le E C^2 \frac{1}{r^2} \|y - x\|^2 + C^2 \|x\|_X^2 \\
&\le \frac{C^2}{r^2} d + C^2 \|x\|_X^2 .
\end{aligned}$$

Furthermore

$$\begin{aligned}
E_y \|f_0(y)\|^2 &\le E_y C^4 \|y\|_X^4 \le 8C^4 E \|x\|_X^4 + \|y - x\|_X^4 \\
&\le 8C^4 \|x\|_X^4 + \frac{1}{r^4} E \|y - x\|^4 \\
&\le 8C^4 \|x\|_X^4 + \frac{4}{r^4} d^4 .
\end{aligned}$$

□

Lemma 61 (Norm comparison) The $\|\cdot\|_X$ can be upper bounded by the Euclidean norm $\|\cdot\|$ as

$$\forall y \in R^d, \frac{1}{R} \|y\| \le \|y\|_X \le \frac{1}{r} \|y\|.$$

Proof. Note that for any $y \in R^d$, for $\|y\| \le r$ we have $\tilde{y} \in X$. Therefore, from the definition of $\|\cdot\|_X$:

$$\|y\|_X \le \inf \{\lambda > 0, \frac{y}{\lambda} \in X\} \le \frac{\|y\|}{r}.$$

Furthermore, for $\lambda \ge \frac{\|y\|}{R}$, then $\frac{y}{\lambda} \in A$ R, which means $y \in X$ (since X is contained in a ball of radius R). Therefore, $\|y\|_X \ge \frac{\|y\|}{R}$. □

Lemma 62 (Gaussian smoothing) For arbitrary unit direction v, given the smooth regularizer defined in equation C.8 we have

$$\begin{aligned}
\|\nabla f(x)\| &\le \frac{1}{\sigma} \sqrt{E f_0(y)^2} \\
\|\nabla^2 f(x, v)\| &\le \frac{4}{\sigma^2} \sqrt{E f_0(y)^2}, \\
\|\nabla^3 f(x, v, w, u)\| &\le \frac{5}{\sigma^3} \sqrt{E f_0(y)^2}.
\end{aligned}$$

Proof. Consider the function $f_0(y) e^{-\frac{\|y - x\|^2}{2\sigma^2}}$; it is continuous in both y, x due to continuity of f_0 by Lemma 57, and its partial derivative with respect to x in direction v is $f_0(y) \cdot \frac{\langle y - x, v \rangle}{\sigma^2}$, we which

is again continuous wrtx and y. Therefore, from the Leibnitz rule, for arbitrary direction v, $Df(x) \cdot v$ exists and is equal to

$$Df(x) \cdot v = E_y\left(\frac{y-x}{2}, v\right) f'(y).$$

Therefore, from Cauchy Schwarz

$$\|Df(x) \cdot v\| \le \frac{1}{2}\sqrt{E(y-x, v)^2}\sqrt{Ef'(y)^2} \le \frac{1}{2}\sqrt{Ef'(y)^2}.$$

For the second derivative

$$D^2 f(x) \cdot (v, w) = E_y\left(\left(\frac{y-x}{2}, v\right)\left(\frac{y-x}{2}, w\right) f'(y) + \frac{1}{2}(v, w) f'(y)\right)$$

which gives

$$\|D^2 f(x) \cdot (v, w)\| \le \frac{1}{2}\sqrt{E(N(0,1))^4}\left(\frac{1}{2}\right)\sqrt{Ef'(y)^2} + \frac{4}{2}\sqrt{Ef'(y)^2}.$$

where N is normal gaussian with variance one. Similarly for the third derivative

$$D^3 f(x) \cdot (v, w, u) = E_y\left(\left(\frac{y-x}{2}, v\right)\left(\frac{y-x}{2}, w\right)\left(\frac{y-x}{2}, u\right) f'(y) + \frac{1}{2}\sum_{u,v,w}(v, w)\left(\frac{y-x}{2}, u\right) f'(y)\right).$$

Therefore,

$$\|D^3 f(x) \cdot (v, w, u)\| \le \frac{1}{3}\sqrt{E(N(0,1))^6}\sqrt{Ef'(y)^2} + \frac{1}{3}\sqrt{E(N(0,1))^4}\sqrt{Ef'(y)^2} + \frac{1}{3}(3)(1)\sqrt{Ef'(y)^2} \le \frac{5}{3}\sqrt{Ef'(y)^2}.$$

□

Corollary 10 (Final smoothed derivatives) For the smoothed barrier defined in Equation equation C.8 and $x > X$, we have

$$\|f(x)\| \le C^2\left(2 + \frac{2}{r^2}\right)(1)$$

$$\|Df(x) \cdot v\| \le C^2\sqrt{1 + 4\left(\frac{1}{r^4}\right)}(4)$$

$$\|D^2 f(v, v)\| \le \frac{4C^2}{2}\sqrt{1 + 4\left(\frac{1}{r^4}\right)}(4);$$

$$\|D^3 f(x) \cdot (v, w, u)\| \le \frac{5C^2}{3}\sqrt{1 + 4\left(\frac{1}{r^4}\right)}(4).$$

Proof. Directly by combining Lemmas 60 and 62. □

Theorem 39 (Existence of a smooth regularizer) Given that there exists a 2-homogeneous regularizer $f_0 : \mathbb{R}^d \to \mathbb{R}$ that is α-strongly convex w.r.t. $\|\cdot\|_{L_c}$ and that $\max_{x \in X} f_0(x) \le B$, C^2, then there also exists a smooth regularizer f which is $\frac{\alpha}{2}$-strongly convex w.r.t. $\|\cdot\|_{L_c}$ and

$$f(x) \le B + C^2,$$

$$\|\nabla f(x)\|_* \le B + C^2 \frac{d^{1-4}}{r},$$

$$\nabla^2 f(v, v) \le B + C^2 \frac{d^{1-2}}{r^2},$$

$$\nabla^3 f(x)(v, w, u) \le B + C^2 \frac{d^{3-4}}{r^3}.$$

Proof. It is enough to set $\frac{r}{d^{1-4}}$ in Corollary 10. □

We can now prove Theorem 3.

Proof of Theorem 3. The proof follows from combining Theorems 39 and 38 with Fact 1. □

C.4 Calculating the Regularizer

In this section, building upon the properties that we showed for feasible points of the program 2.13, we show how to compute a suitable regularizer $g(\cdot)$ on X. To do so, we build a separation oracle for P_I. We start by defining the notions of separation oracle, as well as membership and linear optimization oracle. Before defining these oracle, we need to state the definition of set neighborhoods.

Definition 26 (Membership Oracle) For convex set $D \subset \mathbb{R}^d$, a membership oracle receives a vector $y \in \mathbb{R}^d$ and real number $\delta \ge 0$ and with probability 1 asserts $y \in B(D, \delta)$, or it asserts $y \notin B(D, \delta)$. We denote the computational cost of a query to our membership oracle by $\mathrm{MEM}_X(\delta)$.

Definition 27 (Set neighborhoods) For a subset $D \subset \mathbb{R}^d$, let $B(D, \delta)$ be the set of points that are within distance δ of D, and $B^-(D, \delta)$ be the set of points that where a ball of radius δ around them is completely included in D.

Definition 28 (Separation Oracle) For a convex set $L \subset \mathbb{R}^d$, a separation oracle receives a vector $y \in \mathbb{R}^d$ and real number $\delta \ge 0$ and either asserts $y \in B(L, \delta)$, or it returns a unit vector $c \in \mathbb{R}^d$ such that $c^T y \le c^T x + \delta$ for all $x \in B(L, \delta)$. We denote the computation time of separation oracle by $\mathrm{SEP}_L(\delta)$.

Definition 29 (Linear Optimization Oracle). For a convex set $L \subset \mathbb{R}^d$, a linear optimization oracle receives a unit vector $c \in \mathbb{R}^d$ and real number γ_{lin} and returns a point $y \in C$ such that $\forall x \in C,\ c^T y \le c^T x + \gamma_{\mathrm{lin}}$. We denote the computational cost of calling the linear optimization oracle by $\mathrm{LinO}_L(\gamma_{\mathrm{lin}})$.

Separation, membership, and linear optimization oracles are known to be equivalent and can be used to implement convex optimization over convex sets [328]. Next, we state a simplified version of Theorem 42 in [329] (or Theorem 15 in [330]) on how to build a linear optimization oracle from a separation oracle for a convex set, which we use in the proof of Theorem 41.

Theorem 40 (Theorem 15 in [329] or Theorem 42 in [330]) Let K be a convex set satisfying $B_2(0,r) \subseteq K \subseteq B_2(0,1)$ and let $\delta = \frac{1}{r}$. For $0 < \varepsilon < 1$, with probability $1 - \delta$, we can compute $x \in B(K, \varepsilon)$ such that

$$c^T x < \min_{x \in K} c^T x + \varepsilon \|c\|_2$$

with an expected running time of $O(n \cdot \mathrm{SEP}(K, \varepsilon) \cdot \log(n/\varepsilon) + n^3 \log^{O(1)}(n/\varepsilon))$, where $\mathrm{SEP}(K, \varepsilon)$.

We now state how we solve the optimization problem in Theorem equation 2.13 based on a separation oracle that we build for its feasibility set P_I in Section C.6.

Theorem 41 (Computing the Regularizer - abstract) In the context of Lemma 5 Then, given arbitrary accuracy parameter $0 < \varepsilon_1 < 1$, there is a cutting-plane method that approximately solves the program in khasha: x the program equation parts equation 2.13 and obtains an almost feasible instance I^o, in the sense that

1. $\max_{x \in X} \{g(I^o, x)\} \leq (1 + c_2^2) \cdot \varepsilon_1 \cdot \varepsilon_1$ khasha: x this
2. $g(I^o, x)$ is $\frac{1}{4}$ strongly convex with respect to $\|\cdot\|_{L^c}$,

and runs in time (assuming $N \geq d$)

$$O\left(\frac{N \cdot (c_0^2 + c_1^2 + c_2^2) \cdot \log(c_2 \cdot \|L\| \cdot R)}{\varepsilon_1 \cdot r}\right) \cdot \mathrm{LinO}_L\left(\frac{r}{R^2}, \frac{1}{N \cdot (c_0^2 + c_0^2 + c_2^2) \cdot \varepsilon_1}\right) \cdot \frac{\log(1/\varepsilon_1)}{\varepsilon_1}.$$

Proof. The program equation 2.13 is a linear optimization problem over the convex set P_I, for which we can exploit the separation oracle that we constructed in Lemma 63. In particular, the result directly follows from a simplified version of Theorem 42 in [330] (or Theorem 15 in [329]), a classical result on how to build a linear optimization oracle from the separation oracle for a convex set. For convenience of the reader, we have restated this result in Theorem 40. According to this theorem, for any $0 < \varepsilon_1 < 1$, with probability $1 - \delta_1$ we can compute an instance I^o such that its corresponding barrier $g(I^o, \cdot)$ satisfies

1. $\max_{x \in X} \{g(I^o, x)\} \leq \max_{x \in X} \{g(I^\ddagger, x)\} + \varepsilon_1$, where $I^\ddagger$ is the optimal solution to the LP.
2. I^o is ε_1 close to a feasible instance $\hat{I}$ in Euclidean distance.

Now applying Lemma 5 we conclude the first argument, namely $\max_{x \in X} \{g(I^o, x)\} \leq (1 + c_2^2) \cdot \varepsilon_1 \cdot \varepsilon_1$. Now we need to show that $g(I^o, \cdot)$ roughly remains $\hat{\cdot}$ strongly convex w.r.t $\|\cdot\|_{L^c}$. For this, note that given $x \in X$, if $\|x_j - x\| \leq \left(\frac{\bar{c}_0}{L_0}\right)^{1/3}$ and $\|x_i - x\| \leq \varepsilon$, then from Lemma 4 and the feasibility of $\hat{I}$ we have $r_{x_i} \geq g(\hat{I}, x) \geq \bar{c}_0$ where r_{x_i} is the variable of the valid instance I^r. But picking $\varepsilon_1 \leq \frac{\bar{c}_0}{2R^2}$ we get that $g(I^o, x) \geq g(I^o, x)$. Therefore, again the maximum at x is achieved by one of the functions $g(I^o, x)$ where x_j is not farther than $\left(\frac{\bar{c}_0}{L}\right)^{1/3}$ of x. But then similar to Equation equation C.46 in Lemma 64, for all $\hat{I} \in I$ and arbitrary direction v:

$$v^T \nabla^2 g(\hat{I}, x) v \geq \frac{1}{2} \|v\|_{L^c}^2 :$$

On the other hand, $\|y\| \le \|y\|_2 \le \|y\|_{L_c}$ implies $\|\nabla^2 g(x)\| \le \|\nabla^2 g(x)\|_2 \le \|\nabla^2 g(x)\|_F \le 1$. Therefore, using $\|y\| \le \frac{1}{4r^2}$ we conclude

$$\|\nabla^2 g(x)\| \le \|\nabla^2 g(x)\| + \frac{1}{4}\|y\|_{L_c},$$

which is the desired property. Finally, using the third argument in Lemma 5, we have the following runtime based on Theorem 40:

$$O\left(N \cdot \mathrm{SEP}_{P_1}(\cdot) \cdot \log\left(\frac{1}{\varepsilon}\right) + N^3 \log^{O(1)}\left(\frac{1}{\varepsilon}\right)\right),$$

for

$$\left\langle \left(\frac{\frac{1}{N}\cdot\frac{L^3}{N(1+C_0^2)+d(c_0^2+c_2^2)}}{}\right)^{O(1)}, \left(\frac{\frac{1}{N}\cdot L}{N(C_0^2+c_0^2+c_2^2)}\right)^{O(1)}\right\rangle.$$

Note that from Lemma 63, for this choice of we have

$$\mathrm{SEP}_{P_1}(\cdot) = O\left(\frac{N(C_0^2+c_1^2+c_2^2)+c_2 R}{\frac{1}{N}Lr}\cdot \log\left(\frac{LR}{r}\right)\right)^{O(d)} \cdot \mathrm{SEP}_{L\left(\frac{r}{R^2}\right)}\left(\frac{r}{R}\right) \cdot \left(\frac{\frac{1}{N}\cdot L}{N(C_0^2+c_0^2+c_2^2)}\right)^{O(1)} \cdot \log\left(\frac{d^2}{\varepsilon}\right)$$
$$= O(N^2 d^2);$$

which completes the proof. □

Next, we appropriately instantiate the constants of the convex program equation 2.13 based on Theorem 39 and Lemma 5 in Theorem 1 below. We find the running time of our cutting-plane method to solve this program based on Theorem 41.

Theorem 42 (Restatement of Theorem 1) Assuming $R \ge 1, r \le 1$ for simplicity, given that the best achievable rate for online linear optimization with action and constraint sets (X, L) is $\mathrm{Rate}(X, L) \le T$, there exists an algorithm that runs in time

$$\left(\frac{dR}{r}\right)^{O(d^2)} \cdot \mathrm{SEP}_{L\left(\frac{r}{dR}\right)}\left(\frac{r}{dR}\right)^{O(d)} \cdot \log\left(\frac{d}{\varepsilon}\right);$$

and calculates a regularizer $g(\cdot)$ given by the representation (α, v, r) as described in Section equation 2.8, which satisfies

1. $\sup_{x \in X} g(x) \le 2\mathrm{Rate}(X, L)^2$
2. $g(\cdot)$ is 1-strongly convex w.r.t. $\|\cdot\|_{L_c}$.

Proof. Let $C = \mathrm{Rate}(X, L)$. From Theorem 39 there exists a 2-homogeneous barrier which is $c_1 = O(C^2 d^{1-4})$ Lipschitz, $c_2 = O(C^2 d^{1-2})$ Gradient Lipschitz, $L = O(C^2 d^{3-4})$ Hessian Lipschitz, and 1-strongly convex w.r.t. $\|\cdot\|_{L_c}$. Therefore, to enjoy the properties of Lemma 5, assuming that we guarantee,

$$\varepsilon \le \min\left(\frac{c_1}{L}, \frac{c_2}{L}, \frac{C^2}{L}\right) \tag{C.10}$$

then we get that $c_0; c_2; C_0$ are of the same order as $c_1; c_2; C^2$, respectively (this follows from the de nition of $c_0; c_2; C_0$ which involves the term L^{-3}). Now following the condition of Lemma 5, we consider a cover of accuracy such that

$$B \min\left\{ \frac{1}{r^2}; \frac{r^6}{C^6 d^2}; \frac{r}{d^{1-4}}; C\frac{d^{3-8}}{r^{1-2}}; r d^{1-4}; \frac{r^7}{R^6 C^6 d^{11-8}} \right\}:$$

where we set $L = {}_5 C^2 \frac{d^{3-4}}{r^3}$ for small enough constant ${}_5$. For simplicity if either R or C were smaller than one, we upper bound them by one, so we can assume $R; C \geq 1$ without loss of generality. Similarly if $r > 1$, we can take $r = 1$, so without loss of generality we assume $r \leq 1$. Then, the above bound simpli es to

$$B \frac{r^6}{R^6 C^6 d^2}: \tag{C.11}$$

Furthermore we consider the discretization set X to be points each entry is of the form k for an integer k. Then, to guarantee equation C.10 we should have

$$^3 B \frac{r^2}{d^{1-2}}: \tag{C.12}$$

On the other hand, rounding every point x to its closest multiple of in each coordinate implies that the cover has accuracy as small as $\bar{d}$. Hence, to satisfy condition equation G.122 we set

$$< \frac{{}_4 r^6}{R^6 C^6 d^2 \bar{d}};$$
$$< \frac{{}_4 r^6}{R^6 C^6 d^2};$$

for small enough constant ${}_4$. Then, it is easy to check that condition equation C.12 is automatically satis ed. Furthermore, with this choice of we see that ${}_2 d c_1 \leq B \frac{C^2}{2}$ for small enough constant ${}_4$ (${}_2$ is de ned in Lemma 41); hence, from the guarantee of Lemma 41

$$\max_{x \in X} \left\| \hat{\nabla}^{-1}(x) - \nabla^{-1}(x) \right\| \leq B C^2 \quad {}_2 d c_1 \quad {}_1 B \frac{3}{2} C^2 \quad {}_1;$$

where recall ${}_1$ is the accuracy parameter for our solver in Lemma 41. Setting

$${}_1 \quad \frac{C^2}{2};$$

we conclude

$$\max_{x \in X} \left\| \hat{\nabla}^{-1}(x) - \nabla^{-1}(x) \right\| \leq B 2C^2:$$

Note that the attained constant two behind C^2 does not matter since the parameter C of the smoothed barrier in Theorem 39 can be o by a universal constant from Rate $\hat{X}; L$. Now since the regularizer f is 1 strongly convex, Lemma 41 also guarantees that the

regularizer that we find, $g^{I^o}(x)$, is $\frac{1}{4}$ strongly-convex with respect to $\|\cdot\|_{L_c}$. Finally from the runtime guarantee of Lemma 41, finding such regularizer has runtime

$$O\left(\frac{NR}{r}\right)^{O(d)} \mathrm{poly}\left(L, \log\frac{r}{NR}\right)^{(1)};$$

where we used the fact that $C_0^2 + C_1^2 + C_2^2 = O(C^4R^4d^2)$ and $d \le N$, and that we can upper bound C by R (Note that we dropped the d in the term $\frac{NRd}{r}$ since N is already exponentially large in d). Furthermore, the cover that we considered has size at most $\left(\frac{TR}{r}\right)^d \le \left(\frac{dR}{r}\right)^{O(d)}$. Therefore, the overall runtime is

$$\left(\frac{dR}{r}\right)^{O(d^2)} \mathrm{poly}\left(L, \log\frac{r}{dR}\right).$$

□

C.5 Online Linear Optimization

Here we show how to run FTRL with regularizer g^{I^o} that is based on the instance I^o which we computed in Section C.4 for a general instance of the online linear optimization problem as we defined in Section 2.3.1; as we mentioned, our approach results in the optimal information theoretic rate up to universal constants.

Theorem 43 (Optimal online optimization). Consider the problem of online linear optimization with action and loss sets (X, L) as described in Section 2.3.1. Given access to the regularizer g^{I^o} for the instance I^o of the program 2.13 that we can compute as described in Theorem 1 and a membership oracle for X, there is a cutting-plane algorithm to run FTRL with regularizer g^{I^o}, with running time

$$O\left(Td^2 \ln^{O(1)}(dRT)\right) \mathrm{MEM}_X(\cdot) \cdot \mathrm{poly}(1);$$

which guarantees regret $O(\mathrm{Rate}(X, L^o, T))$.

Proof. We run FTRL with the regularizer $g^{I^o}(\cdot)$; namely, to calculate each step $1 \le t \le T$, we solve the following convex optimization using separation oracle for X:

$$x_t \in \arg\min_{x \in X} G_t(x) \tag{C.13}$$

$$G_t(x) = \langle x, \sum_{s=1}^{t-1} g_s \rangle + g^{I^o}(x); \tag{C.14}$$

up to accuracy $O(\frac{r}{R^2T})$, namely for x_t being the output of the algorithm we have

$$G_t(x_t) - G_t(x_t^*) \le O\left(\frac{r}{R^2T}\right) \cdot \left(\sup_{x \in X} G_t(x) - \inf_{x \in X} G_t(x)\right) \le O\left(\frac{r}{R^2T}\right) \mathrm{Rate}(X, L^o)^{(2)}. \tag{C.15}$$

Note that we used the property that for the regularizer g° that we calculate in Theorem 41 we have $\sup_{x \in X} g^\circ(x) \leq \frac{1}{2}\mathrm{Rate}(X, L)^2$. Then, from Theorem 1 in [329], there is a cutting-plane method whose number of queries to a membership oracle for X is

$$O\left(n^2 \ln\left(1 + \frac{D}{\varepsilon}\right)\right)$$

in addition to $O\left(n^2 \ln\left(1 + \frac{D}{\varepsilon}\right)\right)$ arithmetic operations.

But since x_t is the global minimizer of G_t we have $\nabla G_t(x_t) = 0$, and further from $\frac{r}{4}$-strong convexity of G_t w.r.t. $\|\cdot\|_{L^c}$:

$$G_t(x_t') \geq G_t(x_t) + \frac{r}{4}\|x_t' - x_t\|_{L^c}^2 \geq G_t(x_t) + \frac{r}{4}\|x_t' - x_t\|^2,$$

which combined with equation F.61 implies

$$\|x_t' - x_t\| \leq \frac{\mathrm{Rate}(X, L)}{R\sqrt{T}}.$$

Then, from the mirror descent guarantee we have the following regret bound for the sequence x_t

$$\mathbb{E}\left[\max_{x^\ddagger \in X} \sum_{t=1}^{T} \langle x_t, g_t \rangle - \langle x^\ddagger, g_t \rangle\right] \leq \mathrm{Rate}(X, L)\sqrt{T}. \tag{C.16}$$

On the other hand, using the fact that $\|g_t\| \leq R$ and that $L \geq \frac{1}{R}$,

$$\begin{aligned}
&\mathbb{E}\left[\sum_{t=1}^{T} \langle x_t, g_t \rangle - \langle x_t', g_t \rangle\right] \\
&\mathbb{E}\left[\sum_{t=1}^{T} \|x_t - x_t'\| \|g_t\|\right] \\
&R\,\mathrm{Rate}(X, L)\sqrt{T}.
\end{aligned} \tag{C.17}$$

Combining equation C.16 and equation C.17 completes the proof for the regret guarantee. □

C.6 Separation Oracle

Here we show a separation oracle for the feasible polytope P of program 2.13.

Lemma 63 (Linear optimization oracle for L – Separation Oracle) The polytope P_l for $l = (r, v, \ldots)$ defined in equation 2.13 has a separation oracle with computational cost

$$\mathrm{SEP}_K(\cdot) \leq O\left(\frac{2c_2 R^3}{r^3}\left(n \cdot \mathrm{LinO}_L(1, r, 8R^2) + d^2\right) + O(T^2 d^2)\right),$$

where $\mathrm{LinO}_L(1, r, 8R^2)$ is the cost of a linear optimization oracle for L with parameter $(1, r, 8R^2)$.

Proof. We can readily check if conditions (1) and (2) hold for the instance , and if not, that condition de nes the direction c for which `I ; ce C `I ; ce for all I > P_I . To check condition (3) we can do singular value decomposition in O(d^3). Condition (4) is a bit trickier since it might be hard to directly maximize v^T x_i v over L. Therefore, we work with the discretization set S_d of the unit d-dimensional sphere; in particular, for every unit direction v > S_d, we consider condition (5) with a margin _m, namely

$$v^T x_i v \geq \|v\|_{L_c}^2 + 1 - \gamma_m. \tag{C.18}$$

This margin allows us to easily obtain a feasible solution in P_I which satis es v^T x_i v for all v > L, using condition in equation C.18 which is only for the discretization points; moreover, we check equation C.18 with our linear optimization oracle which has error _lin in calculating ||v||_{L_c}; namely, suppose equation C.18 holds for all > S_d given that we substitute ||v||_{L_c} in equation C.18 with the output of LinO(L, _lin). Then, we are guaranteed that for every v > S^d:

$$v^T x_i v \geq (\mathrm{LinO}(L, \gamma_{lin}) - \gamma_{lin})^2 + 1 - \gamma_m. \tag{C.19}$$

Now from the fact that ||v||_{L_c} r and LinO(L, _lin) ||v||_{L_c} _lin, picking _lin , we get that

$$v^T x_i v \geq (1 - \gamma_{lin}/2)\|v\|_{L_c}^2 + 1 - \gamma_m, \tag{C.20}$$

which using the fact that we picked _lin _m/4 implies

$$v^T x_i v \geq \|v\|_{L_c}^2 (1 - \gamma_{lin}/2) + 1 - \gamma_m \geq 1 - \gamma_m/2. \tag{C.21}$$

Now for arbitrary direction v > S^d on the unit sphere, we bound the value of the quadratic form the closest point in the discretization set: namely for v > S^d where ||v|| = 1:

$$\begin{aligned} |v^T x_i v - \|v\|_{L_c}^2 \; v^T x_i v - \|v\|_{L_c}^2| \\ |v^T x_i v - \|v\|_{L_c}^2 \; v^T x_i v - \|v\|_{L_c}^2| + |v^T x_i v - \|v\|_{L_c}^2 \; v^T x_i v - \|v\|_{L_c}^2|. \end{aligned} \tag{C.22}$$

but for the first term, using ||x_i|| c_2:

$$|v^T x_i v \; v^T x_i v| = |(v + v)^T x_i (v - v)| \leq \|v + v\| \|x_i\| \|v - v\| \leq 2c_2 \|v - v\|$$

and ||v||_{L_c} r. Hence, from 1

$$|v^T x_i v - \|v\|_{L_c}^2 \; v^T x_i v - \|v\|_{L_c}^2| \leq 2c_2 \frac{1}{r^2}. \tag{C.23}$$

For the second term, using the fact that r ||v - v||_{L_c}, ||v||_{L_c} R and ||v - v||_{L_c} R ||v - v||:

$$\begin{aligned} |v^T x_i v - \|v\|_{L_c}^2 \; v^T x_i v - \|v\|_{L_c}^2| &\leq 2c_2 \frac{\|v - v\|_{L_c}^2 \; \|v\|_{L_c}^2}{\|v - v\|_{L_c}^2 \; \|v\|_{L_c}^2} \\ &\quad 2c_2 \frac{\|v - v\|_{L_c} (\|v\|_{L_c} - \|v - v\|_{L_c}) \; \|v - v\|_{L_c}}{\|v - v\|_{L_c}^2 \; \|v\|_{L_c}^2} \\ &\quad c_2 \frac{\|v - v\|_{L_c}}{\|v - v\|_{L_c} \; \|v\|_{L_c}} \quad c_2 \frac{\|v - v\|_{L_c}}{\|v - v\|_{L_c}^2 \; \|v\|_{L_c}} \\ &\quad 2c_2 \frac{1}{r^3}. \end{aligned} \tag{C.24}$$

Combining Equations equation G.48 and equation C.24 (from $R \leq r$ C1) and plugging into equation C.22

$$\nabla_{x_i} v \leq \nabla Y_{L_c}^2 \quad \nabla_{x_i} \nabla \leq \nabla Y_{L_c}^2 + \frac{4 c_2 R}{r^3};$$

which combined with equation C.21 and triangle inequality

$$\nabla_{x_i} v \leq (\nabla Y_{L_c} \cdot)^2 + (1 + m) \cdot \frac{4 c_2 R}{r^3}.$$

Using $\leq \frac{r^3 m}{c_2 R}$, we get

$$\nabla_{x_i} v \leq (\nabla Y_{L_c} \cdot)^2 + (1 + m) \cdot.$$

Recall that v was arbitrary in S^d. Therefore, in the case when all inequalities in equation C.19 are satisfied, we showed that I indeed satisfies condition (4) in equation 2.13. Finally if any of the inequalities equation C.20 are violated, i.e. if $\nabla_{x_i} \nabla \leq (\text{LinO}_L + _{\text{lin}}) \cdot \nabla \cdot)^2 + (1 + m) \cdot$, then similar to equation C.20 we get

$$\nabla_{x_i} \nabla \leq (1 + _{\text{lin}}) \cdot \nabla Y_{L_c} \cdot)^2 + (1 + m) \cdot + \nabla_{x_i} \nabla \leq (\text{LinO}_L + _{\text{lin}}) \cdot \nabla \cdot)^2 + (1 + m) \cdot;$$

which implies (from $_{\text{lin}} + _m = 4$)

$$\nabla_{x_i} \nabla \leq (\nabla Y_{L_c} \cdot)^2 + (1 + m) \cdot (1 + _{\text{lin}}) \cdot^2 (1 + m) \cdot 2 _m \cdot.$$

Therefore, we find that the unit direction $\nabla \nabla$ which satisfies

$$\begin{aligned} \nabla \nabla, _{x_i} e &+ \nabla Y_{L_c}^2 \quad 2 _m \nabla Y_{L_c}^2 \\ &+ \nabla Y_{L_c}^2 \quad 2 _m R^2; \end{aligned}$$

while for a valid $I > P_I$, we should have $\nabla v, _{x_i} e \quad \nabla Y_{L_c}^2$ for all unit directions v. Hence, we constructed a separation oracle with $2 _m R^2$, which uses S^d queries to the linear optimization oracle, and its overal computational cost is $O(S^d) \cdot \text{LinO}_L + _{\text{lin}} \cdot d^2 \cdot + d^2 \cdot$. Finally to have a -separation oracle, we need to guarantee $2 _m R^2 + , _{\text{lin}} + \frac{m}{4}, \frac{r m}{2}, \leq \frac{r^3 m}{c_2 R}$, hence we pick

$$\begin{aligned} _m &< \frac{}{2 R^2}; \\ _{\text{lin}} &< \frac{_m (1 + r \cdot)}{4} \quad \frac{(1 + r \cdot)}{8 R^2}; \\ \leq &< \frac{r^3}{2 c_2 R^3}. \end{aligned}$$

Hence, the overall computational cost is

$$O(1 \leq \cdot^d \log \text{LinO}_L (1 + r \cdot) \cdot 8 R^2 \cdot) \quad d^2) \quad O(S^d) \cdot d^2 \cdot$$

$$O\left(\frac{2 c_2 R^3}{r^3} \cdot^d \log \text{LinO}_L (1 + r \cdot) \cdot 8 R^2 \cdot) \quad d^2) \quad O(S^d) \cdot d^2 \cdot\right).$$

□

C.7 Proofs for Sections 2.5 and 2.8

C.7.1 Proof of Lemma 1

For the lower bound, we use the inequality $\nabla^2 f(x_1) \succeq \nabla^2 f(x_0) - L\|x_1 - x_0\|I$:

$$\begin{aligned} f(x) &\geq f(x_0) + \nabla f(x_0)^T(x - x_0) + \int_0^1 \int_0^t (x - x_0)^T \nabla^2 f(x_0 + s(x - x_0))(x - x_0)\, ds\, dt \\ &\geq f(x_0) + \nabla f(x_0)^T(x - x_0) + \int_0^1 \int_0^t (x - x_0)^T (\nabla^2 f(x_0) - sL\|x - x_0\| I)(x - x_0)\, ds\, dt \\ &= f(x_0) + \nabla f(x_0)^T(x - x_0) + \frac{1}{2}(x - x_0)^T \nabla^2 f(x_0)(x - x_0) - \frac{L}{6}\|x - x_0\|^3 \\ &\geq f_{x_0}(x) - \frac{L}{6}\|x - x_0\|^3. \end{aligned}$$

For upper bound, we use the inequality $\nabla^2 f(x_1) \preceq \nabla^2 f(x_0) + L\|x_1 - x_0\|I$:

$$\begin{aligned} f(x) &\leq f(x_0) + \nabla f(x_0)^T(x - x_0) + \int_0^1 \int_0^t (x - x_0)^T \nabla^2 f(x_0 + s(x - x_0))(x - x_0)\, ds\, dt \\ &\leq f(x_0) + \nabla f(x_0)^T(x - x_0) + \frac{1}{2}(x - x_0)^T \nabla^2 f(x_0)(x - x_0) + \frac{L}{6}\|x - x_0\|^3 \\ &\leq f_{x_0}(x) + \frac{L}{2}\|x - x_0\|^3. \end{aligned}$$

C.7.2 Proof of Lemma 4

We denote $g_{x_i}(x)$ in short by $g_{x_i}(x)$, and without loss of generality let $x_i = x_0$ and $x_j = x_1$. First, note that we can translate the convex program conditions on the norm of v_{x_i} to

$$\|v_{x_i}\| \leq c_1,$$

for $c_1 \in [c_0, \bar{c}_0]$. From the program constraint khasha: is this correct? we have

$$g_{x_1}(x_0) \leq \frac{15L}{96}\|x_1 - x_0\|^3 + r_{x_0}. \tag{C.25}$$

On the other hand, from $\|x_0 - x\| \leq \delta$ and the norm bounds on gradient and Hessian

$$\begin{aligned} g_{x_1}(x_0) - g_{x_1}(x) &\geq -v_{x_1}^T(x_0 - x) + (x_0 - x)^T \int_0^1 \nabla^2 f_{x_0}(x + s(x_0 - x))(x_0 - x)\, ds - \frac{L}{3}\|x_1 - x_0\|^3 - \|x_1 - x\|^3 \qquad \text{(C.26)} \\ &\geq -c_1\|x_0 - x\| - c_2\|x_0 - x\|^2 - \|x_1 - x_0\|^2\|x_0 - x\| - \|x_1 - x\|^2\|x_0 - x\| \qquad \text{(C.27)} \\ &\geq -\frac{L}{3}\|x_0 - x\|^3 - \|x_1 - x_0\|^2\|x_0 - x\| - \|x_1 - x\|^2\|x_0 - x\|; \qquad \text{(C.28)} \\ &\geq -c_1\|x_0 - x\| - c_2\|x_0 - x\|^2 - 2\|x_0 - x\|^2 \qquad \text{(C.29)} \\ &\geq -\frac{L}{3}\|x_0 - x\|^3 - 4\|x_1 - x_0\|^2\|x_0 - x\| - 2\|x_0 - x\|^2; \qquad \text{(C.30)} \end{aligned}$$

where in the last line we used

$$(x_1 - x_0)^2 \le (x_1 - x)^2 + (x_1 - x_0)(x_1 - x) \le (x_1 - x_0)^2 + 2(x_1 - x_0)(x_1 - x) + (x_1 - x)^2 \tag{C.31}$$

$$\le 4(x_1 - x_0)^2 + 2(x_0 - x)^2. \tag{C.32}$$

Note that picking $\delta \le 3$, from the triangle inequality, $|x - x_1| \le 3\delta \le \frac{c_1}{L}|x - x_1|^{1-\gamma}$, and the condition that $\frac{\bar{d}c_0}{L} \le 1$,

$$|x_0 - x_1| \le |x_1 - x| + |x - x_0| \le 2\left(\frac{c_1}{L}\right)^{\frac{1}{\gamma}}. \tag{C.33}$$

Now based on equation C.33, for the first term in equation C.30, we can write

$$c_1|x_0 - x|\left(c_1 + \frac{L}{48}|x_1 - x_0|^2\right), \tag{C.34}$$

Similarly, also because $\delta \le \frac{L}{2000c_1c_2^3}$, for the second term we have

$$2c_2|x_0 - x||x - x_0|\left(c_1 + \frac{L}{24}|x_1 - x_0|^2\right), \tag{C.35}$$

and because $\delta \le \frac{8L}{c_2}$,

$$2c_2|x_0 - x|^2\left(\frac{L}{24}|x_1 - x_0|^2\right). \tag{C.36}$$

Finally for the last term, because $\delta \le \sqrt{\frac{c_1}{4096}}$,

$$\frac{4L}{3}|x_0 - x||x - x_0|\left(c_1 + \frac{L}{48}|x_1 - x_0|^2\right) \tag{C.37}$$

and

$$\frac{4L}{3}|x_0 - x|^3\left(\frac{L}{48}|x_1 - x_0|^2\right). \tag{C.38}$$

Therefore, defining

$$\begin{aligned}\Delta_{x_0,x}(x_0 - x_1) \le\ & c_1|x_0 - x| + c_2|x_0 - x|^2 + 2|x_0 - x_1||x_0 - x| \\ & \frac{L}{3}|x_0 - x|\left(4|x_1 - x_0|^2 + 2|x_0 - x|^2\right),\end{aligned}$$

we showed in equation C.30 that for arbitrary x_1,

$$g(x_1) - g(x_0) \ge -\Delta_{x_0,x}(x_0 - x_1), \tag{C.39}$$

and for x_1 such that $|x - x_1| \le 3\left(\frac{\bar{d}c_0}{L}\right)^{\frac{1}{\gamma}}$, or $|x_1 - x_0| \le 2\left(\frac{c_1}{L}\right)^{\frac{1}{\gamma}}$, Combining equation C.34, equation J.16, equation G.119, equation C.37, equation C.38 with equation C.30:

$$\Delta_{x_0,x}(x_0 - x_1) \le \frac{3L}{48}|x_0 - x|\left(c_1 + \frac{L}{48}|x_1 - x_0|^2\right). \tag{C.40}$$

Therefore, for $x_1 \in [x_0, x_0 + 4(\bar{dc}_0/L)^{1/3}]$,

$$g_{x_1}(x_0) \geq g_{x_1}(x) + \frac{7L}{48}(x_0 - x_1)^3;$$

which combined with Equation equation C.25

$$g_{x_1}(x) \leq \frac{L}{96}(x_1 - x_0)^3 + r_{x_0}. \tag{C.41}$$

On the other hand, note that

$$g_{x_0}(x) \leq r_{x_0} + v_{x_0}(x)(x - x_0) + \frac{1}{2}(x - x_0)^2 \leq \frac{c_2}{2}\|x - x_0\|^2 + 2c_1;$$

where in the last line we used $r \leq \frac{c_1}{c_2}$. But now picking the constant large enough we can guarantee that

$$\frac{L}{96}(x_0 - x_1)^3 \leq 3c_1:$$

Combining equation C.42 with equation C.41, we conclude the first argument

$$g_{x_1}(x) \leq c_1 + g_{x_0}(x):$$

On the other hand, note that $\|x_0; x\|(x_1)$ is increasing in $(x_1 - x_0)$. Therefore, combining equation C.39 and equation C.40, for any x_1 such that $(x_1 - x) \leq \left(\frac{\bar{dc}_0}{L}\right)^{1/3}$

$$g_{x_1}(x_0) \geq g_{x_1}(x) + \|x_0; x\|(x_1 - x) + \|x_0; x\|\left(\frac{\bar{dc}_0}{L}\right)^{1/3}\left(\frac{3L}{48}\left(\frac{\bar{dc}_0}{L}\right)^{1/3}\right)^3 \tag{C.42}$$

$$\geq \frac{\bar{dc}_0}{2}. \tag{C.43}$$

C.7.3 Proof of Lemma 5

Note that the Hessian of $(x - x_0)^3$ is strong convexity of f means for v with $(v, v)_L = 1$ we have $v \nabla^2 f(x_0) v \geq$. But from Assumption equation 1 we get $(v, v)_B \leq R$. Therefore,

$$\begin{aligned} v \nabla^2 f_{x_0}(x) v &\geq v \nabla^2 f(x_0) v - L\|x - x_0\|(x - x_0) v \\ &\geq v \nabla^2 f(x_0) v - L\|x - x_0\|(x - x_0) v \\ &\geq v \nabla^2 f(x_0) v - L(x - x_0)\left(\frac{L}{(x - x_0)}(x - x_0)(x - x_0)^T\right) v \\ &\geq -2R^2 L(x - x_0) \\ &\geq -\frac{}{2}: \end{aligned}$$

C.7.4 Proof of Theorem 5

Here we prove Theorem 5. Before diving into the proof, we need to state and prove Lemma 64 so that we can obtain an $\tilde{\Omega}(\sqrt{\cdot})$ strong convexity property for the approximate regularizer in Theorem 5. In particular, Lemma 64 combines Lemmas 4 and 5 and concludes that the feasibility of I for the program implies strong convexity of g with respect to $\|\cdot\|_{L^c}$.

Lemma 64 (Program feasibility $\Rightarrow$ strong convexity). Suppose $(\hat{r}, v, \cdot)$ is a feasible solution to LP equation 2.13 with respect to an ϵ-cover $\{x_i\}_{i=1}^N$ in X for the Euclidean norm, i.e. $\forall x \in X, \exists x_i$ s.t. $\|x - x_i\| \le \beta$, where β satisfies

$$\beta \le \sqrt[3]{\frac{\bar{d}c_0}{512R^6L^2c_0}}.$$

Then, for any point $x \in X$, g is second order continuously right and left differentiable with

$$D^{2,l}g(x; v, v), D^{2,r}g(x; v, v) \ge -\frac{\epsilon}{2}\|v\|_{L^c}^2,$$

where $D^{2,l}g(x; v, v)$ and $D^{2,r}g(x; v, v)$ denote the left and right second order directional derivative of f at x in direction v.

Proof. For $x \in X$ let $I(x) = \arg\max_{i \in [N]} g_{x_i}(x)$ be the set of indices for which $g_{x_i}(x)$ achieves its maximum at x. First, note that for the one-dimensional function $h(t) = g^{I(x)}(x + tv)$, the subgradient of h zero is exactly

$$\min_{i \in I(x)} Dg_{x_i}(x; v), \max_{i \in I(x)} Dg_{x_i}(x; v),$$

due to the convexity of g_{x_i}'s. In fact, $h'(0) = \min_{i \in I(x)} Dg_{x_i}(x; v)$ and $h'(0) = \max_{i \in I(x)} Dg_{x_i}(x; v)$. Now let

$$I^{r,v} = \arg\max_{i \in I(x)} Dg_{x_i}(x; v)$$
$$I^{l,v} = \arg\min_{i \in I(x)} Dg_{x_i}(x; v).$$

Then the second left and right directional derivatives at point x are given by

$$D^{2,l}g(x; v, v) = h''(0) = \max_{i \in I^l(x)} D^2 g_{x_i}(x; v, v), \tag{C.44}$$
$$D^{2,l}g(x; v, v) = h''(0) = \max_{i \in I^r(x)} D^2 g_{x_i}(v; v). \tag{C.45}$$

Furthermore, note that from Lemma 4, for every x_i such that $\|x_i - x\| \le \left(\frac{\bar{d}c_0}{L}\right)^{1/3}$, we have $g_{x_i}^{I(\cdot)}(x) = g_{x_0}^{I(\cdot)}(x)$, therefore $i \in I$. Hence, we should have $\|x - x_{\hat{I}(x)}\| \le \beta \le \left(\frac{\bar{d}c_0}{L}\right)^{1/3}$ for all $\hat{I} \in I$. But using the upper bound given on β we get

$$\|x - x_{\hat{I}(x)}\| \le \beta \le \left(\frac{\bar{d}c_0}{L}\right)^{1/3} \le \frac{\bar{d}}{2R^2L}.$$

Hence, From Lemma 5, we have that $g_{x_{\hat{i}}}(x)$ is $\frac{\sigma}{2}$ strongly convex at x, for all $\hat{i} > I$:

$$\nabla^2 g_{x_{\hat{i}}}(x) \succeq \frac{\sigma}{2} I_{L_c} \tag{C.46}$$

Finally combining this with equation C.45 we conclude

$$D^{2,l} g(x)(v,v) + D^{2,r} g(x)(v,v) \geq \frac{\sigma}{2} \|v\|^2_{L_c}.$$

□

Next, we state the proof of Theorem 5.

Proof of Theorem 5. Consider the solution $\tilde{\Gamma} = (\tilde{w}, \tilde{v}, \tilde{\alpha})$ where $\forall i > N$

$$\begin{aligned}
\tilde{\alpha}_{x_i} &= \nabla^2 f(x_i), \\
\tilde{v}_{x_i} &= \nabla f(x_i), \\
\tilde{r}_{x_i} &= f(x_i).
\end{aligned}$$

First note that from Lemma 1 we get $f_{x_0}(x) \leq \frac{1}{6}\|x - x_0\|^3 + f(x)$, which implies $g^{\hat{\Gamma}}_{x_i}(x_j) \leq \frac{15L}{96}\|x_j - x_i\|^3 + r_{x_j}$ for the above choice for $\tilde{\Gamma}$. Moreover, $r_{x_0} \leq f(x_0) + C^2 \leq C_0$, and from c_1 Lipschitz and c_2 gradient Lipschitz conditions on f, we get $\forall i, \forall x_i \in B(c_1, \forall i)$, $\tilde{\alpha}_{x_i} \preceq c_2 I$, and the $\|\cdot\|_{L_c}$ strong convexity of f shows that $\tilde{\Gamma}$ satisfies the condition $\tilde{v}_{x_i} \geq \tilde{\alpha}_{x_i} + \forall v > 0, \forall i$. Hence, $\tilde{\Gamma}$ is feasible for the LP. In particular, note that we do not need the additional L^3 terms in the definition of c_0, c_2, C_0 to show the feasibility of $\tilde{\Gamma}$ for the LP; these extra terms are only required for the third argument of Lemma 5 to show that not only $\tilde{\Gamma}$ is feasible, but a ball around it is also feasible. We will prove that shortly. Next, from Lemma 4, we see that the maximum $\max_{i>N} g^{\tilde{\Gamma}}_{x_i}(x)$ at point $x > X$ is never achieved by far x_j's from x, farther than $\|x_j - x\| > C + \frac{\bar{dc}_0}{L}$, since the value of $g_{x_j}(x)$ is smaller than $g_{x_i}(x)$ for the element of the cover x_i that is close to x. On the other hand, again from Lemma 4 for x_i such that $\|x_i - x\| \leq B$ and any x_j such that $\|x_j - x\| \leq B + \frac{\bar{dc}_0}{L}$, we have

$$\left| g^{\hat{\Gamma}}_{x_j}(x_i) - g^{\hat{\Gamma}}_{x_j}(x) \right| \leq \frac{\bar{dc}_0}{2},$$

and from LP feasibility

$$g^{\hat{\Gamma}}_{x_j}(x_i) \leq r_{x_i}.$$

Therefore,

$$\begin{aligned}
\max_{i>N} \left| g^{\hat{\Gamma}}_{x_i}(x) \right| &\leq B + \max_{i>N} |r_i| + \frac{\bar{dc}_0}{2} \\
&\leq \max_{i>N} |f(x_i)| + \frac{\bar{dc}_0}{2} \\
&\leq B + C^2 + \frac{\bar{dc}_0}{2}.
\end{aligned}$$

Therefore, the optimal solution $I^{\ddagger}$ should satisfy $\max_{i>N} \{g_{x_i}(x) - f(x)\} \geq C^2 \cdot \frac{\varepsilon}{2} - c_0$ which proves the first argument equation 1. Finally, combining Lemmas 64 and 65 we get the second shows strong convexity of $g_{x_i}(\cdot)$ with respect to $Y - Y_{L_c}$ for argument equation 2.

Next we show the third argument; note that f satisfies a slightly stronger inequality compared to the first condition of the LP equation 2.13, namely

$$f(x_i) - f(x_j) - (x_i - x_j)^T \nabla f(x_j) \geq \frac{1}{2}\|x_i - x_j\|^2 \nabla^2 f(x_j)(x_i - x_j) + \frac{L}{3}\|x_i - x_j\|^3 \tag{C.47}$$

$$\geq \frac{L}{6} - \frac{L}{96}\|x_i - x_j\|^3 - \frac{L}{96}\|x_i - x_j\|^3 \geq f(x_j) - f(x_i), \tag{C.48}$$

or, since we constructed instance $\tilde{f}$ from f,

$$g_{x_i}(x_j) - f(x_j) \geq \frac{15L}{96}\|x_i - x_j\|^3 - \frac{L}{96}\|x_i - x_j\|^3 \geq f(x_j) - f(x_i). \tag{C.49}$$

But if $\|\nabla^2 f(x_i)\| \leq \frac{L}{144}$ then $\|\nabla f(x_i) - \nabla f(x_j)\| \leq \frac{L}{144}\|x_i - x_j\|$, then

$$\begin{aligned}\frac{1}{2}\|x_j - x_i\|\|\nabla^2 f(x_i)(x_j - x_i)\| &\leq \|x_j - x_i\|\|\nabla f(x_j) - \nabla f(x_i)\| \leq \|x_j - x_i\|\left\|\int_0^1 \nabla^2 f(x_i + t(x_j - x_i))(x_j - x_i)\,dt\right\| \\ &\leq \|x_j - x_i\|\int_0^1 \|\nabla^2 f(x_i + t(x_j - x_i))\|\|x_j - x_i\|\,dt \\ &\leq \frac{L}{288}\|x_j - x_i\|^3.\end{aligned}$$

Given $\|\nabla f(x_i) - v\| \leq \frac{L^2}{288} + \frac{L}{288}\|x_j - x_i\|^2$, we get

$$\|\nabla f(x_i) - v\|\|x_j - x_i\| \leq \|v\|\|x_j - x_i\| + \|\nabla f(x_i) - v\|\|x_j - x_i\| \leq \|v\|\|x_j - x_i\| + \frac{L}{288}\|x_j - x_i\|^3.$$

Finally under $\|\nabla f(x_i)\| \leq \frac{L}{288}^3 + \frac{L}{288}\|x_j - x_i\|^2$. Hence, if we assume $\|v\| \leq \frac{L}{288}^3$, then combining the above Equations we get

$$g_{x_i}(x_j) - g_{x_i}(x_j) \leq g_{x_i}(x_j) - f(x_j) + f(x_j) - f(x_i) + \frac{L}{96}\|x_i - x_j\|^3.$$

But plugging this into equation C.49

$$g_{x_i}(x_j) - f(x_j) \geq \frac{15L}{96}\|x_i - x_j\|^3 \geq f(x_j) - f(x_i), \tag{C.50}$$

Finally note that $\|v\| \leq \frac{L}{288}^3$ also implies $\forall i > N$:

$$\begin{aligned}&\|x_i - x_j\| \geq \|x_i - x_j\| - \|x_i - x_j\| \geq \|x_i - x_j\| - c_2 \|x_i - x_j\|^3, \\ &\|\nabla f(x_i) - v\| \geq \|\nabla f(x_i)\| - \|v\| \geq \|\nabla f(x_i)\| - c_1 \|x_i - x_j\|^3, \\ &\|x_i - x_j\| \leq \|x_i - x_j\| + \|x_i - x_j\| \leq \|x_i - x_j\| + c_2 \|x_i - x_j\|^3.\end{aligned}$$

Therefore, $\tilde{r}$ is still feasible for the program equation 2.13 with our choice of parameters $c_0; c_2; C_0$ here. Hence, we conclude

$$B_L \leq 288\hat{r}\bullet\, b P_l \;\; b B_2 \sqrt{\hat{N}\;\; 1\bullet C_0{}^2\;\; Nd\%_0^2\;\; c_2^2}\hat{r}\bullet:$$

Finally note that for arbitrary $I > P_l$ which satis es the conditions in LP equation 2.13, we have

$$\|I\|^2 Br^2 \;\; \sum_i S_{x_i}^2 S \;\; \|v_{x_i}\|^2 \;\; \|x_i\|^2$$
$$B\hat{N}\;\; 1\bullet C_0{}^2\;\; Ndc_0^2\;\; Ndc_2^2;$$

which implies

$$P_l \;\; b B_2 \sqrt{\hat{N}\;\; 1\bullet C_0{}^2\;\; Nd\%_0^2\;\; c_2^2}\hat{r}\bullet:$$

□

C.7.5 Proof of Theorem 25

Consider the random distribution in Theorem 1.2 of [28]. Then from property (3), there exists a unit direction v with $\|v\|_L \;\; B \frac{1}{d^1}$. Then we claim that $\|v\|_{L^c} \;\; B \frac{1}{d^1}$. This is because $\|v\|_L \;\; \sup_{\|w\|_L^c \;\; 1} \langle v; w\rangle \leq \|v\|_L^c \frac{v}{\|v\|_L}$ e $\frac{1}{\|v\|_L^c}$. Hence, $\|v\|_L^c \;\; C d^1$. Hence, for $v \;\; \frac{v}{\|v\|_{L^c}}$ we have $\|v\|_{L^c} \;\; 1$ and $\|v\| \;\; \|v\|_{L^c} \;\; B \frac{1}{d^1}$.

C.8 Strong convexity

Here we show that a lower bound on the second derivative implies strong convexity with respect to arbitrary norms.

Lemma 65 (Lower bound on second derivative $\Rightarrow$ strong convexity). Suppose for convex function $g : X \;\; R$ which is second order continuously di erentiable except in a nite number of points in which it is only left or right second order di erentiable. Suppose the second left or right derivatives in arbitrary direction v, which we denote by $D^{2;l} g\hat{x}\bullet\, v; v\rangle$; $D^{2;r} g\hat{x}\bullet\, v; v\rangle$ respectively, are at least $\|v\|_{L^c}^2$. Then, g is strongly convex with respect to $\|:\|_L^c$, namely for any $x; y > X$ and any subgradient v_x of f at point x:

$$f\hat{y}\bullet \;\; f\hat{x}\bullet\;\; \langle\, f\hat{x}\bullet; y \;\; x\rangle \;\; \|y \;\; x\|_{L^c}^2:$$

Proof. Without loss of generality assume $\|y \;\; x\|_{L^c} \;\; 1$ and de ne the one variable function $h\hat{t}\bullet : 0;1 \;\; R: h\hat{t}\bullet \;\; g\hat{x} \;\; t\hat{y} \;\; x\bullet\bullet$, and let $0 B t_1 B t_2 B \;\; B t_k B 1$ are the non-di erentiable points of $h\hat{t}\bullet$ on $0;1$, which we know are nite from our assumption. But from di erentiability of h between these points, we can write (de ne $t_0 \;\; 0; t_{k\;\; 1} \;\; 1$)

$$f\hat{y}\bullet \;\; g\hat{1}\bullet \;\; \sum_{i\;\; 1}^{k} \int_{t_i}^{t_{i\;\; 1}} g\text{œ}t\bullet dt; \qquad (C.51)$$

where for the integral in $t_i; t_{i\ 1}$ by $h^{\oe}t_i\bullet$ and $h^{\oe}t_{i\ 1}\bullet$ we mean the right derivative$h^{\oe\hat{}}t_i\bullet$ and left derivative $h^{\der\hat{}}t_{i\ 1}\bullet$, respectively. Now we show that for all$t > 0; 1$

$$h^{\der\hat{}}t\bullet; h^{\oe\hat{}}t\bullet \ C h^{\oe\hat{}}0\bullet \quad t: \tag{C.52}$$

We show this inductively for$t > \hat{} t_i; t_{i\ 1}\bullet$ for $i \quad 0; : : : ; k$. Particularly, the induction argument for step i is that for $t > \hat{} t_i; t_{i\ 1}\bullet$, $h^{\oe}t\bullet \ C \ t \quad h^{\oe\hat{}}0\bullet$, and $h^{\der\hat{}}t_{i\ 1}\bullet; h^{\oe\hat{}}t_{i\ 1}\bullet \ C h^{\oe\hat{}}0\bullet \quad t_{i\ 1}$. The base trivial since$h^{\oe\hat{}}0\bullet \ C h^{\oe\hat{}}0\bullet$. For the step of induction from$i \quad 1$ to i, we know

$$g^{\oe\hat{}}t_i\bullet \ C \ t_{\ i}: \tag{C.53}$$

Now for any$t > \hat{} t_i; t_{i\ 1}\bullet$ we can write

$$h^{\oe}t\bullet \quad S_{t_i}^{\ t} h^{\oe\oe}s\bullet ds \ C \ \hat{} t \quad t_i\bullet; \tag{C.54}$$

and particularly for $t_{i\ 1}$:

$$h^{\der\hat{}}t_{i\ 1}\bullet \quad S_{t_i}^{\ t_{i\ 1}} h^{\oe\oe}s\bullet ds \ C \ \hat{} t_{i\ 1} \quad t_i\bullet: \tag{C.55}$$

On the other hand, from the convexity of$g$,

$$h^{\der\hat{}}t_{i\ 1}\bullet \ B h^{\oe\hat{}}t_{i\ 1}\bullet: \tag{C.56}$$

Combining equation C.54 equation C.55 equation C.56 with equation C.53 completes the setp of induction.

Finally combining equation C.52 with equation C.51 and noting the fact that for any subgradient$v$ at point x, $\grave{} v; y \quad x; B e h^{\oe\hat{}}0\bullet$,

$$f \hat{} y\bullet \ C h^{\oe\hat{}}0\bullet \quad S_0^{\ 1} tdt \ \ C h^{\oe\hat{}}0\bullet \quad \frac{}{2};$$

which completes the proof. □

Appendix D

Projection-Free Online Convex Optimization via E cient Newton Iterations

D.1 Self-concordance properties

Throughout, for a twice-di erentiable function f int K R, we let ^x; f • Y©f ^x•Y© ²f ^x• denote the Newton decrement of f at x >int K.

Lemma 66. Let f int K R be a self-concordant function with constant M f C1. Further, let x >int K and x f >arg min x>K f ^x•. Then, I) whenever ^x; f • @1~M f , we have

$$Yx \quad x_f Y_{©^2 f ^x_f} . YYX \quad x_f Y_{©^2 f ^x} . B \ ^x; f •~^1 \quad M_f \ ^x; f ••;$$

and II) for any M CM f , the Newton step x x © ²f ^x•©f ^x• satis es x >int K and ^x ; f • BM ^x; f •²~^1 M ^x; f ••²:

Lemma 67. Let f int K R be a self-concordant function with constant M f and x >int K. Then, for any w such that r Yw xY©²f ^x• @1~M f , we have

$$^1 \quad M_f r•^2©^2 f ^w• j \ ©^2 f ^x• j \ ^1 \quad M_f r• \ ^2©^2 f ^x•:$$

The following result from [62, Theorem 5.1.5] will be useful to show that the iterates of algorithms are always in the feasible set.

Lemma 68. Let f int K R be a self-concordant function with constant M f C1 and x >int K. Then, E x ~w >R^d Yw xY x @1~M f • b int K. Furthermore, for all w >E x, we have

$$Yw \quad xY_w B \frac{Yw \quad xY_x}{1 \quad M_f Yw \quad xY_x}:$$

Finally, we will also make use of the following result due to [54]:

Lemma 69. Let $f : \mathrm{int}\, K \to \mathbb{R}$ be a self-concordant function with constant $M_f > 0$. Then, for any $x, w \in \mathrm{int}\, K$ such that $r := \|x - w\|_{\nabla^2 f(x)} < 1/M_f$, we have

$$\|\nabla f(x) + \nabla f(w)\|^2_{\nabla^2 f(x)^{-1}} \le \frac{1}{(1 - M_f r)^2} \|w - x\|^2_{\nabla^2 f(x)}.$$

D.2 Technical Lemmas

Our analysis relies on the crucial fact that the Newton decrement can be sufficiently decreased by taking a Newton step using only approximate gradients and Hessians. We state this fact next; the proof is in §D.4.1.

Lemma 70 (Decrease in the Newton decrement) Let f be a self-concordant function over $\mathrm{int}\, K$ with constant $M > 0$, and let $y \in \mathbb{R}^d$ be such that $\lambda(y) \le 1/40M$. Further, let $H \in \mathbb{R}^{d \times d}$ and $\hat{g}_y \in \mathbb{R}^d$ be such that

$$\|\hat{g}_y - \nabla f(y)\|_{\nabla^2 f(y)^{-1}} \le \varepsilon := \frac{1}{40M}, \tag{D.1}$$

$$(1 - \varepsilon)^2 \nabla^2 f(y) \preceq H \preceq (1 - \varepsilon)^{-2} \nabla^2 f(y); \tag{D.2}$$

for $\varepsilon \le 1/5$. Then, for $y^+ := y - H^{-1} \nabla f(y)$ and $y^+ := y - H^{-1} \hat{g}_y$, we have

$$\lambda(y^+) \le 9M \lambda(y)^2 + 2.5 \lambda(y),$$
$$\lambda(y) \le 20/19 \cdot \|\hat{g}_y\|_{y^+} + 20/19 \cdot \varepsilon \cdot \lambda(y^+).$$

Next, we show that as long as the Newton decrement is small enough at the current iterate w_{t-1}, the intermediate Newton iterates w_t^m remain close to the landmark point u_{t-1}; this will be important for the proof of Theorem 6. The proof is in §D.4.2.

Lemma 71 (Invariance under Newton iterations) Let f be a self-concordant function over $\mathrm{int}\, K$ with constant $M > 0$. Let $b > 0$, $m_{\mathrm{Newton}} := \log \frac{1}{\varepsilon M}$, $\alpha \le 1/1000M$, $\varepsilon \le 1/20000M$, $\eta := 0.001$, and $\beta \le 1/1000M$. Further, let $(w_t, w_t^m, u_t, g_t, H_t)$ be as in Algorithm 1 with input $(\alpha, \varepsilon, \eta, m_{\mathrm{Newton}})$. Suppose that at round $t-1$ of Algorithm 1, we have

$$\lambda(w_{t-1}, \nabla f(u_{t-1})) \le \beta \quad \text{and} \quad \|u_{t-1} - w_{t-1}\|_{\nabla^2 f(u_{t-1})} \le \frac{1}{40M}. \tag{D.3}$$

For $t > 1$, if the sub-gradient g_{t-1} at round $t-1$ satisfies $\|g_{t-1}\|_{\nabla^2 f(w_{t-1})^{-1}} \le b$, then

$$\lambda(w_t^m, \nabla f(w_t^m)) \le \left(\frac{15}{16}\right)^{m-1} \lambda(w_t^1, \nabla f(w_t^1)) + 500\eta \le \frac{1}{40M}. \tag{D.4}$$

Furthermore, we have for all $m \ge m_{\mathrm{Newton}}$:

$$\frac{1}{2}\nabla^2 f(w_t^m) \preceq \nabla^2 f(w_t^\dagger) \preceq 2\nabla^2 f(w_t^m); \tag{D.5}$$

$$\|w_t^m - u_{t-1}\|_{H_{t-1}} \le \frac{1}{10M}; \tag{D.6}$$

$$\|w_t^m - w_t^\dagger\|_{\nabla^2 f(w_t^\dagger)} \le \frac{1}{49M} \cdot \left(\frac{15}{16}\right)^{m-1} + 240\eta; \tag{D.7}$$

$$\|\nabla f(w_t^m) - u_{t-1}\|_{H_{t-1}} \le \|\nabla f(w_{t-1}) - u_{t-1}\|_{H_{t-1}} + 2\|g_{t-1}\|_{\nabla^2 f(w_{t-1})} \cdot \frac{1}{40M} \cdot \left(\frac{15}{16}\right)^{m-1} + 500\eta + 2\alpha; \tag{D.8}$$

where $w_t^\dagger > \arg\min_{w \in K} \ell_t(w)$ is the optimum solution of ℓ_t.

We note that we have not made an attempt to optimize over the constants in Lemma 71.

Lemma 72. Let ϕ be a (M, ν)-self-concordant barrier for K, and let $w^\dagger > \arg\min_{w \in K} \sum_{t=1}^{T} \ell_t(w)$. Further, suppose that the losses satisfy:

$$\sup_{w \in \text{int } K, t \in [T]} |\nabla \ell_t(w)^\top (w - w^\dagger)| \le \frac{1}{\sqrt{T}}. \tag{D.9}$$

Then, for any $w \in K$, there exists $\tilde{w} \in K_{1-\frac{1}{T}}$ (where K_c is as in equation 3.4) such that

$$\sum_{t=1}^{T} \left(\ell_t(w_t) - \ell_t(w) \right) \le \sum_{t=1}^{T} \left(\ell_t(w_t) - \ell_t(\tilde{w}) \right) + O(\sqrt{T}). \tag{D.10}$$

Proof. Fix $w \in K$ and define $\tilde{w} = (1 - \frac{1}{T}) w^\dagger + \frac{1}{T} w \in K_{1-\frac{1}{T}}$. Then, by equation D.9, we have, for all $t \in [T]$,

$$\nabla \ell_t(w_t)^\top (w_t - w^\dagger) - \nabla \ell_t(w)^\top (\tilde{w} - w^\dagger) \le O(T^{-1/2}). \tag{D.11}$$

Summing this over $t = 1, \ldots, T$ leads to the desired result. □

D.3 Proofs of the Main Results

Next, we present the proof of Theorem 6.

D.3.1 Proof of Theorem 6

Proof. The proof consists of three parts: I) First, we show that BARONS keeps the Newton decrements $\|w_t, \ell_t\|_t^*$ small this is the main invariant of BARONS; II) Then, we bound the regret of BARONS using this invariant and the results of Lemma 71; III) Finally, we bound the runtime of BARONS.

Bounding the Newton decrements. We will show that the Newton decrements satisfy

$$\|w_s, \ell_s\|_s^* \le \min\left\{ \frac{1}{1000M}, 1000 \right\}, \tag{D.12}$$

for all $s \in [1]$. We will show equation D.12 by induction over $t \in [1]$.

Base case. The base case follows by the facts that $w_1 = \arg\min_{w \in K} \phi(w)$, $\nabla_1 = 0$ and that the Newton decrement is zero at the minimizer.

Induction step. Suppose that equation D.12 holds with $s = t - 1$ for some $t \in [1]$. We will show that it holds for $s = t$. First, note that by the update rule of landmark (see Lines 12 and 17 of Alg. 1), we have that

$$\|w_{t-1} - u_{t-1}\|_{H_{t-1}} \le \frac{1}{41M},$$

where $H_{t-1} = \nabla^2 h(u_{t-1})$. Thus, by the choice of η in the theorem's statement, we have

$$\|u_{t-1} - w_{t-1}\|_{\nabla^2 h(u_{t-1})} \le \frac{1}{40M}. \tag{D.13}$$

This, combined with the fact that equation D.12 holds with $s = t-1$ (the induction hypothesis) implies that the conditions of Lemma 71 are satisfied. This in turn implies

$$h(w_t, \eta_t) \overset{(a)}{\le} h(w_t^{m_{\text{Newton}}}, \eta_t) \le \left(\frac{15}{16}\right)^{m_{\text{Newton}}} h(w_t^1, \eta_t) \le 500 \cdot \frac{50}{M} \left(\frac{15}{16}\right)^{m_{\text{Newton}}} \overset{(b)}{\le} \beta,$$

where (a) follows by the fact that $w_t = w_t^{m_{\text{Newton}}}$ (by definition; see Algorithm 1) and (b) follows by the choice of m_{Newton} in the theorem's statement. This shows that equation D.12 holds for $s = t$ and concludes the induction.

Bounding the regret. To bound the regret of BARON, we make use of the FTRL iterates $\tilde{w}_t^\dagger$, which are given by $w_t^\dagger = \arg\min_{w \in \mathcal{K}} \eta_t h(w)$. By Lemma 71, we have that for all $t > T_0$,

$$\|w_t - w_t^\dagger\|_{\nabla^2 h(w_t^\dagger)} \cdot \|w_t - w_t^{m_{\text{Newton}}}\|_{\nabla^2 h(w_t^\dagger)} \le \frac{1}{49M} \left(\frac{15}{16}\right)^{m_{\text{Newton}}} \le 240 \cdot \varepsilon_{\text{opt}}, \tag{D.14}$$

where the last inequality follows by the choice of $m_{\text{Newton}} \ge \log(1/\varepsilon_{\text{opt}}M)$ in the theorem's statement. Using this and Hölder's inequality, we now bound the sum of linearized losses of the algorithm in terms of the sum of linearized losses with respect to $\tilde{w}_t^\dagger$:

$$\sum_{t=1}^{T} \langle w_t, g_t \rangle - \sum_{t=1}^{T} \langle w_t^\dagger, g_t \rangle \le \sum_{t=1}^{T} \|w_t - w_t^\dagger\|_{\nabla^2 h(w_t^\dagger)} \cdot \|g_t\|_{\nabla^2 h(w_t^\dagger)^{-1}}. \tag{D.15}$$

Now, by equation D.5 in Lemma 71 (which holds due to equation D.13 and equation D.12 with $s = t-1$ as we showed in the prequel), we have $\frac{1}{2}\|\cdot\|^2_{\nabla^2 h(w_t)} \le \|\cdot\|^2_{\nabla^2 h(w_t^\dagger)}$, which implies that $\|g_t\|_{\nabla^2 h(w_t^\dagger)} \le 2\|g_t\|_{\nabla^2 h(w_t)}$. Combining this with equation D.14 and equation D.15, we get that

$$\sum_{t=1}^{T} \langle w_t, g_t \rangle - \sum_{t=1}^{T} \langle w_t^\dagger, g_t \rangle \le \varepsilon_{\text{opt}} \sum_{t=1}^{T} \|g_t\|_{\nabla^2 h(w_t)}. \tag{D.16}$$

Now $x, w \in \mathcal{K}$. Subtracting $\sum_{t=1}^{T} \langle w, g_t \rangle$ from both sides of equation D.16 implies the following bound the regret of BARON:

$$\begin{aligned} \text{BARON}_T(w) &\le \sum_{t=1}^{T} \langle w_t^\dagger, g_t \rangle - \sum_{t=1}^{T} \langle w, g_t \rangle + \varepsilon_{\text{opt}} \sum_{t=1}^{T} \|g_t\|_{\nabla^2 h(w_t)}, \\ &\le \frac{1}{\eta} h(w) + \sum_{t=1}^{T} \|g_t\|^2_{\nabla^2 h(w_t)^{-1}} + \varepsilon_{\text{opt}} \sum_{t=1}^{T} \|g_t\|_{\nabla^2 h(w_t)}, \end{aligned}$$

where the last inequality follows by the regret bound of FTRL (see e.g. [55]).

Bounding the run-time. Note that BARONUSpdates the landmark points on the rounds where Yu$_t$ w$_t$Y$_{H_t}$ A 1~^41M •. Now, by equation D.8 in Lemma 71 (which holds due to equation D.13 and equation D.12 with$s$ t 1 as we showed in the prequel), we have

$$\begin{aligned} \text{UuY}_t \quad w_t Y_{H_t} \quad Yu_t \quad w_{t\ 1} Y_{H_t} \text{U} \quad \text{UuY}_t \quad w_t^{m_{\text{Newton}}} Y_{H_t} \quad Yu_t \quad w_{t\ 1} Y_{H_t} \text{U}, \\ \text{B2 Yg}_{t\ 1} Y_{\copyright\ 2\ \hat{}w_{t\ 1}\bullet} \quad \frac{1}{40M} < \frac{15}{16}\bullet^{m_{\text{Newton}}} \quad 500' \quad 2 \\ \text{B2 Yg}_{t\ 1} Y_{\copyright\ 2\ \hat{}w_{t\ 1}\bullet} \quad O\hat{}"\bullet; \end{aligned}$$

where the last inequality follows by the choice ofm_{Newton} in the theorem's statement. Hence, the quantity Yu$_t$ w$_t$Y$_{H_t}$ increases each time by at mos2 Yg$_{t\ 1}$Y$_{\copyright\ 2\ \hat{}w_{t\ 1}\bullet}$ O^"•. Therefore, the number of times that the landmarku$_t$ changes is bounded by

$$\text{OCE}\frac{\text{P}_{t\ 1}^{T}\hat{}2\ \text{Yg}_t Y_{\copyright\ 2\ \hat{}w_t\bullet} \quad O\hat{}"\bullet\bullet}{1\sim\hat{}41M\ \bullet}\text{`} \quad \text{OCEM T"} \quad M \sum_{t\ 1}^{T} \text{Yg}_t Y_{\copyright\ 2\ \hat{}w_t\bullet}\text{`}:$$

Thus, the overall computational cost of recalculating the Hessians and their inverses at the landmark iterates is bounded by

$$C^{\text{hess}}\ \text{CEM T"} \quad M \sum_{t\ 1}^{T} \text{Yg}_t Y_{\copyright\ 2\ \hat{}w_t\bullet}\text{`}:$$

where the multiplicative cost C$^{\text{hess}}$ re ects the fact that the instance of BARONUSn the theorem's statement needs1 accurate approximations of the Hessians and their inverses (in the sense of De nition 9) at the landmark iterates. MoreoverBARONUSeeds to compute an "-approximate gradient of at every point w$_t^m$ for all t > T and m > m$_{\text{Newton}}$. Thus, the cost of computing the gradients isC^{grad} T log $\frac{1}{"M}$. Finally, the matrix-vector product H$_t$ $^{1}\copyright$ $_t$^w• in BARONUSosts O^d^2• work, and so overall the computational cost is

$$\text{OCEC}^{\text{grad}} \quad d^2\bullet \quad T \log\frac{1}{"M} \quad C^{\text{hess}}\ \text{CEM T"} \quad M \sum_{t\ 1}^{T} \text{Yg}_t Y_{\copyright\ 2\ \hat{}w_t\bullet}\text{`}:$$

□

D.3.2 Proof of Theorem 7

Proof. Note that without having any e ect on the algorithm, we can add an arbitrary constant to the barrier . Thus, without loss of generality, we assume ^w†• 0, which implies ^w• C0, for all w >K. We de ne the restricted comparator class

$$K_c \quad \tilde{}w > K \quad \hat{}w\bullet \text{B} \quad \hat{}w^{\dagger}\bullet \quad \log c\bullet:$$

By [62, Corollary 5.3.3] and the fact that is an ^M ; •-self-concordant barrier forK, we have that

$$K_c \text{ b } K_{\text{C}}; \tag{D.17}$$

and so it su ces to bound the regret against comparators in $\mathcal{C}$. Fix $w^* \in \mathcal{C}$. Under the assumptions of the theorem, the preconditions of Theorem 6 are satis ed and so we have,

$$\sum_{t=1}^{T} \ell_t(w_t) - \ell_t(w^*) \le \frac{1}{\eta}\left(\psi(w^*) - \min_{w} \psi(w)\right) + \eta \sum_{t=1}^{T} \|g_t\|_{(t)}^2 + \epsilon \sum_{t=1}^{T} \|g_t\|_{(t)};$$
$$\le \frac{1}{\eta} \log c + b^2 T\eta + \epsilon T b; \quad (\text{since } w^* \in \mathcal{C} \text{ and } \|g_t\|_{(t)} \le b)$$
$$\le 2b\sqrt{T \log c} + b\epsilon T;$$

where in the last step we used the choices of η and ϵ in equation 3.5. Combining this with equation D.17 implies the desired regret bound. The bound on the computational complexity follows immediately from Theorem 6, the fact that $\|g_t\|_{(t)} \le b$, and the choices of η and ϵ in equation 3.5. □

D.3.3 Proof of Theorem 8

Proof. Similar to the proof of Theorem 7, and without loss of generality, we assume that ψ is zero at its minimum, i.e. $\psi(w^\dagger) = 0$. We de ne the restricted comparator class

$$\mathcal{C}_K = \{w \in K \mid \psi(w) \le \psi(w^\dagger) + \log T\}. \tag{D.18}$$

By [62, Corollary 5.3.3] and the fact that ψ is an (M, ν)-self-concordant barrier for K, we have that

$$K_{1/T} \subseteq \mathcal{C}_K. \tag{D.19}$$

On the other hand, by Lemma 72 we have that

$$\sup_{w \in \operatorname{int} K} \sum_{t=1}^{T} \ell_t(w_t) - \ell_t(w) \le \sup_{w \in \operatorname{int} K} \sum_{t=1}^{T} \ell_t(w_t) - \ell_t(w) + O\left(\frac{1}{T}\right).$$

Combining this with equation D.19 implies that it su ces to bound the regret against comparators in $\mathcal{C}_K$. Fix $w^* \in \mathcal{C}_K$. Note that since ψ is equal to ϕ plus a quadratic, ψ is also a self-concordant function with constant $M \le M$ [62]. Thus, under the assumptions of the theorem the preconditions of Theorem 6 are satis ed and so we have,

$$\sum_{t=1}^{T} \ell_t(w_t) - \ell_t(w^*) \le \frac{1}{\eta}\left(\psi(w^*) - \min_{w} \psi(w)\right) + \eta \sum_{t=1}^{T} \|g_t\|_{(t)}^2 + \epsilon \sum_{t=1}^{T} \|g_t\|_{(t)}. \tag{D.20}$$

Now, by the choice of ϵ, we have that

$$\|g_t\|_{(t)} \le R\|g_t\| \le \frac{1}{\eta T} \le RG\frac{1}{\eta T}.$$

Moreover, from the condition that $K \subseteq B_{\|\cdot\|}(R)$, equation D.18 and the fact that $\psi(w^\dagger) \ge 0$, we have for all $w \in \mathcal{C}_K$:

$$\psi(w) \le \log T \le \frac{1}{2\eta}.$$

Plugging this into equation D.20 and using that $w_t > K$, we get

$$\sum_{t=1}^{T} \left(\ell_t(w_t) - \ell_t(w^*) \right) \leq \frac{1}{\eta} \log T + \frac{1}{2} \left(R^2 G^2 \eta T + \varepsilon \frac{T}{\eta} R G \right) ;$$
$$2RG \sqrt{T \log T} + \frac{5}{2} RG \varepsilon T ;$$

where in the last step we used the choices of η and ε in equation 3.8. Combining this with equation D.19 implies the desired regret bound. The bound on the computational complexity follows from the computational complexity in Theorem 6 and the fact a gradient Oracle $O'^{\mathrm{grad}}(\cdot)$ [resp. Hessian Oracle $O'^{\mathrm{grad}}(\cdot)$] for $(\cdot) + \frac{1}{2R^2} \|Y - Y\|^2$ can be implemented with one call to $O'^{\mathrm{grad}}(\cdot)$ [resp. $O^{\mathrm{hess}}(\cdot)$] plus d arithmetic operations. □

D.4 Proofs of the Technical Lemmas

D.4.1 Proof of Lemma 70

Proof. Throughout, we let h is the Newton step based on the exact gradient $\nabla \phi(y)$:

$$h = H^{-1} \nabla \phi(y) .$$

Recall that $\tilde{y}$ and y from the lemma's statement satisfy

$$\tilde{y} = y - h \quad \text{and} \quad \|y - y\|_y \leq \frac{1}{2} .$$

Bounding the Newton decrement at $\tilde{y}$. First, we bound the Newton decrement at $\tilde{y}$. By definition, the square of the Newton decrement at $\tilde{y} = y - h$ is

$$\|\tilde{y} - \cdot\|_{\tilde{y}}^2 = \nabla \phi(y - h) - \nabla \phi(y) \|^2 \|y - h - y\| = \nabla \phi(y - h) - \nabla \phi(y) .$$

Now for the vector z defined below, we define the function F as

$$z < \nabla \phi(y - h) - \nabla \phi(y) \quad \text{and} \quad F(y) = \nabla \phi(y) - z . \tag{D.21}$$

The partial derivative of F in direction h is given by

$$\begin{aligned} DF(y)\, h &= h - \nabla^2 \phi(y) z ; \\ &= \nabla \phi(y) - H^{-1} \nabla^2 \phi(y) z ; \\ &= \nabla \phi(y) - H^{-1/2} H^{-1/2} \nabla^2 \phi(y) H^{-1/2} H^{-1/2} z ; \\ &= \nabla \phi(y) - H^{-1/2} \left(H^{-1/2} \nabla^2 \phi(y) H^{-1/2} \right) H^{1/2} z = \nabla \phi(y) - z : \end{aligned} \tag{D.22}$$

Now, by equation D.2, we have

$$\left\| H^{-1/2} \nabla^2 \phi(y) H^{-1/2} - I \right\| \leq \frac{1}{1} :$$

Thus, the first term on the right-hand side of equation D.22 can be bounded as

$$\begin{aligned}
&\left\| \hat{y}_{\bullet} - H^{1/2}\hat{H}^{1/2} \right\|^2 \left\| \hat{y}_{\bullet} H^{1/2} \right\| \left\| H^{1/2} z \right\| \\
&\leq \frac{1}{n} \left\| \hat{y}_{\bullet} - H^{1/2}\hat{H}^{1/2} \right\| \left\| \hat{y}_{\bullet} - H^{1/2} \right\| \left\| H^{1/2} z \right\| \\
&\quad + \frac{1}{n} \left\| \hat{y}_{\bullet} - H^{1/2} \right\| \left\| H^{1/2} z \right\| ; \\
&\leq \frac{\hat{\lambda}_1 + \bullet}{\hat{\lambda}_1 - \bullet^2} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\| \left\| z - y_{\bullet} \right\| \left\| z \right\| ; \\
&\quad + \frac{\hat{\lambda}_1 + \bullet}{\hat{\lambda}_1 - \bullet^2} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\| \left\| z \right\| .
\end{aligned}$$

Plugging this into equation D.22 and using the definitionz in equation D.21, we obtain

$$\left\| \hat{D} F(\hat{y}_{\bullet}, h) - F(\hat{y}_{\bullet}, \bullet) \right\| \leq \frac{\hat{\lambda}_1 + \bullet}{\hat{\lambda}_1 - \bullet^2} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\| \left\| z - y_{\bullet} \right\| . \tag{D.23}$$

Now, let $y(s, \bullet) = y_{s\bullet}$ and $F(y(s, \bullet), \bullet) = F(y_{s\bullet}, \bullet)$. With this, we have

$$\left\| F(y_{\bullet s}, \bullet) - F(y_{\bullet 0}, \bullet) - h\hat{D}^2 F(\hat{y}_{s\bullet}, \bullet) \right\| \leq \frac{1}{2} \left\| \hat{y}_{s\bullet} - y_{0\bullet} \right\| . \tag{D.24}$$

On the other hand, by Lemma 73 and our assumption on $\hat{y}(s, \bullet)$, we have

$$\left\| y_{s\bullet} - y_{\bullet} \right\| \leq \left\| s h y_{\bullet} \right\| + \left\| \frac{s}{n} \hat{D} \hat{y}_{\bullet} \right\| \leq \frac{1}{n} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\| + \frac{1}{n} , \tag{D.25}$$

$$\leq \frac{1}{30M} . \tag{D.26}$$

Thus, by Lemma 66, we have

$$\left\| \hat{y}_{s\bullet} - y_{\bullet} \right\|^2 \leq \frac{1}{\hat{\lambda}_1 - M \left\| y_{s\bullet} - y_{\bullet} \right\|^2} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\|^2 \left\| \hat{y}_{s\bullet} - y_{\bullet} \right\|^2 \leq \frac{1}{\hat{\lambda}_1 - M \left\| y_{s\bullet} - y_{\bullet} \right\|^2} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\|^2 .$$

This, together with equation J.28 also implies that

$$\left\| \hat{y}_{s\bullet} - y_{\bullet} \right\|^2 \leq \frac{1}{\hat{\lambda}_1 - 3M \left\| y_{s\bullet} - y_{\bullet} \right\|^2} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\|^2 \left\| \hat{y}_{s\bullet} - y_{\bullet} \right\|^2 \leq \frac{1}{\hat{\lambda}_1 - 3M \left\| y_{s\bullet} - y_{\bullet} \right\|^2} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\|^2 .$$

After rearranging, this becomes

$$\left\| \hat{y}_{s\bullet} - y_{\bullet} \right\|^2 \leq \frac{1}{3M \left\| y_{s\bullet} - y_{\bullet} \right\|^2} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\|^2 \left\| \hat{y}_{s\bullet} - y_{\bullet} \right\|^2 \leq \frac{1}{3M \left\| y_{s\bullet} - y_{\bullet} \right\|^2} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\|^2 .$$

Combining this with equation D.25 and the fact that $c \geq \frac{1}{4}$ gives

$$\left\| \hat{y}_{s\bullet} - y_{\bullet} \right\|^2 \leq \frac{4}{M} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\|^2 \left\| \hat{y}_{s\bullet} - y_{\bullet} \right\|^2 \leq \frac{4}{M} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\|^2 . \tag{D.27}$$

Finally, by Lemma 74 and equation F.78, we obtain the following bound on the right-hand side of equation J.20:

$$\left\| F(y_{\bullet s}, \bullet) - F(y_{\bullet 0}, \bullet) \right\| \leq \frac{6}{M} \left\| \hat{y}_{\bullet} - y_{\bullet} \right\| \left\| z - y_{\bullet} \right\| .$$

Integrating this over s gives

$$\begin{aligned}
\nabla f(x_t + h\Delta x_{nt}) - \nabla f(x_t) &= \int_0^1 \nabla^2 f(x_t + s h \Delta x_{nt}) h \Delta x_{nt} \, ds \\
\|\nabla f(x_t + h\Delta x_{nt}) - \nabla f(x_t)\|_2 &\leq \|\nabla^2 f(x_t) h \Delta x_{nt}\|_2 + \int_0^1 \|(\nabla^2 f(x_t + s h \Delta x_{nt}) - \nabla^2 f(x_t)) h \Delta x_{nt}\|_2 \, ds \\
&\leq \|\nabla^2 f(x_t) h \Delta x_{nt}\|_2 + \frac{6M}{1} \|h \Delta x_{nt}\|_2 \int_0^1 s \, ds \\
&\leq \|\nabla^2 f(x_t) h \Delta x_{nt}\|_2 + \frac{6M}{1} \|h \Delta x_{nt}\|_2^2 \qquad \text{(D.28)}
\end{aligned}$$

where the last inequality follows by equation D.25. Now, note that from equation F.78 (with s 1) and the assumption that $\|w_t\|_2 \leq 1/(40M)$, we have

$$\|\nabla^2 f(x_t)\|_2 \leq \frac{10}{9} \|\nabla^2 f(x_t + h\Delta x_{nt})\|_2 .$$

This implies

$$\|\nabla^2 f(x_t + h\Delta x_{nt})\|_2 \geq \frac{10}{9} \|\nabla^2 f(x_t)\|_2 \quad \Rightarrow \quad \frac{10}{9} \|\nabla^2 f(x_t)\|_2 \leq \|\nabla^2 f(x_t + h\Delta x_{nt})\|_2 .$$

Plugging this into equation G.1 and using equation D.23, we get

$$\begin{aligned}
\|\nabla f(x_t + h\Delta x_{nt})\|_2 &\leq \|\nabla f(x_t) + \nabla^2 f(x_t) h \Delta x_{nt}\|_2 + \frac{20M}{3(1 - \beta)} \|\Delta x_{nt}\|_2^2 \\
&\leq \frac{1 - \beta}{1 - \beta^2} \|\Delta x_{nt}\|_2 + \|\nabla^2 f(x_t) h \Delta x_{nt}\|_2 + \frac{20M}{3(1 - \beta)} \|\Delta x_{nt}\|_2^2 \\
&\leq \frac{10}{9} \frac{1 - \beta}{1 - \beta^2} \|\Delta x_{nt}\|_2 + \frac{20M}{3(1 - \beta)} \|\Delta x_{nt}\|_2^2 \\
&\leq \|\Delta x_{nt}\|_2 + \frac{20M}{3(1 - \beta)} \|\Delta x_{nt}\|_2^2 .
\end{aligned}$$

Now, from the de nition of z, we have

$$\nabla f(x_t + h\Delta x_{nt}) - z \leq \|\Delta x_{nt}\|_2^2 .$$

Thus, since $\|w_t\|_2 \leq 1/(40M)$, we nally get

$$\|\Delta x_{nt}\|_2 \leq 9M \|\Delta x_{nt}\|_2^2 + 2.5 \|\Delta x_{nt}\|_2 .$$

This proves the rst part of the claim, i.e. equation J.20.

Bounding the Newton decrement at $x_t + \Delta x_{nt}$. We now bound the Newton decrement at $x_t + \Delta x_{nt}$ in terms of that of $x_t + \Delta x_{nt}$. Note that

$$\Delta x_{nt} + \Delta x_{nt} = -(\nabla^2 f(x_t))^{-1} (\nabla f(x_t) + \Delta x_{nt}) .$$

On the other hand, from equation D.45 and equation J.28, we have

$$\|\Delta x_{nt} + \Delta x_{nt}\|_2 \leq \frac{3M}{1} \|\Delta x_{nt}\|_2^2 + \frac{7}{4} \|\nabla f(x_t) + \Delta x_{nt}\|_2 \leq \frac{7}{4} \|\Delta x_{nt}\|_2 , \qquad \text{(D.29)}$$

which implies that

$$\|\nabla f(x_t) - \nabla f(x^*)\|_2 \le \|\nabla^2 f(x^*)^{1/2}\| \cdot \|\nabla^2 f(x^*)^{-1/2}(\nabla f(x_t) - \nabla f(x^*))\|_2 \le \frac{7}{4}\epsilon_{\text{stat}}.$$

Therefore,

$$\begin{aligned}
&\|\nabla f(x_t) - \nabla f(x^*)\|_2^2 \\
&\quad = \|\nabla f(x_t) - \nabla f(x^*)\|_{\nabla^2 f(x^*)^{-1}} \cdot \|\nabla f(x_t) - \nabla f(x^*)\|_2 \\
&\quad = \|\nabla f(x_t) - \nabla f(x^*)\|_{\nabla^2 f(x^*)^{-1}} \cdot \|\nabla^2 f(x^*)^{1/2}(x_t - x^*)\|_2 \\
&\quad \le \|\nabla f(x_t) - \nabla f(x^*)\|_{\nabla^2 f(x^*)^{-1}} \cdot \|\nabla^2 f(x^*)^{1/2}\| \cdot \|x_t - x^*\|_2 \\
&\quad \le \frac{49}{16}\epsilon^2 \|\nabla f(x_t) - \nabla f(x^*)\|_{\nabla^2 f(x^*)^{-1}} \cdot \|x_t - x^*\|_2 \\
&\quad \le \frac{49}{16}\epsilon^2 \|\nabla f(x_t) - \nabla f(x^*)\|_2 \cdot \|\nabla^2 f(x^*)^{-1}\|.
\end{aligned}$$

Combining this with equation D.29 and our assumption on $\hat{H}_t$ from equation D.1 implies

$$\|\nabla f(x_t) - \nabla f(x^*)\|_2 \cdot \|\nabla^2 f(x^*)^{-1}\| \le \frac{7}{2}\epsilon \|\nabla f(x_t) - \nabla f(x^*)\|_2 \cdot \|\nabla^2 f(x^*)^{-1}\| \le \frac{1}{2}\epsilon. \tag{D.30}$$

Thus, by Lemma 66, we have

$$\|x_t - x^*\|_2 \le \|\nabla^2 f(x^*)^{-1}\| \cdot \|\nabla f(x_t) - \nabla f(x^*)\|_2 \le \frac{1}{(1-\epsilon)^2}\|\nabla^2 f(x^*)^{-1}\| \cdot \|\nabla f(x_t)\|_2.$$

Since $\epsilon \le 1/40$, we get

$$2(1-\epsilon)^2 \|x_t - x^*\|_2 \cdot \|\nabla^2 f(x^*)\| \le \|\nabla f(x_t)\|_2 \le 2(1+\epsilon)^2 \|x_t - x^*\|_2 \cdot \|\nabla^2 f(x^*)\|. \tag{D.31}$$

Now, by Lemma 69 instantiated with $x = x_t$ and $w = x^*$, we have

$$\begin{aligned}
&\sqrt{(\nabla f(x_t) - \nabla f(x^*))^T \nabla^2 f(x^*)^{-1} (\nabla f(x_t) - \nabla f(x^*))} \ge \frac{\|x_t - x^*\|_{\nabla^2 f(x^*)} \cdot \|\nabla f(x_t) - \nabla f(x^*)\|_2}{(1+\epsilon)\|x_t - x^*\|_{\nabla^2 f(x^*)} \cdot \|\nabla f(x_t) - \nabla f(x^*)\|_2} \\
&\qquad \ge (1-3\epsilon)\|x_t - x^*\|_2;
\end{aligned} \tag{D.32}$$

where in the last inequality we used equation D.30 and the fact that $\epsilon \le 1/40$.

Using the triangle inequality, we can bound the Newton decrement at x_t as

$$\begin{aligned}
\lambda(x_t) &= \sqrt{(\nabla f(x_t) - \nabla f(x^*))^T \nabla^2 f(x^*)^{-1} (\nabla f(x_t) - \nabla f(x^*))} \\
&\quad + \sqrt{\nabla f(x^*)^T \nabla^2 f(x^*)^{-1} \nabla f(x^*)}; \\
&\le 2\sqrt{(\nabla f(x_t) - \nabla f(x^*))^T \nabla^2 f(x^*)^{-1} (\nabla f(x_t) - \nabla f(x^*))} \\
&\quad + \sqrt{\nabla f(x^*)^T \nabla^2 f(x^*)^{-1} \nabla f(x^*)}; \quad \text{(by equation D.31 and } \epsilon \le 1/40) \\
&\le 2(1-3\epsilon)^{-1}\|x_t - x^*\|_{\nabla^2 f(x^*)} + \|\nabla f(x^*)\|_{\nabla^2 f(x^*)^{-1}};
\end{aligned}$$

where the last inequality follows by equation D.31 and equation D.32. This completes the proof. □

D.4.2 Proof of Lemma 71

Proof. By definition of $\hat{w}_t^m(\cdot)$ in Algorithm 1, we have $w_t^1 = w_{t-1}$ and $w_t = w_t^{m_{\text{Newton}}}$. We show properties equation D.4, equation D.6, equation D.7, and D.8 using induction over $m = 1, \ldots, m_{\text{Newton}}$.

Base case. We start with the base case; $m = 1$. Note that from the assumption in equation D.3 and definition of w_t^1, we have

$$\|w_t^1 - u_{t-1}\|_{\Sigma_{\hat{w}_{t-1}}} \le B(1-\hat{\alpha})^{40M}. \tag{D.33}$$

Now, by definition of the Oracle O^{hess} and the fact that $H_{t-1} = O^{\text{hess}}(u_{t-1})$ (see Algorithm 1) with $\alpha = 0.001$, we have

$$(1-0.001)\nabla^2 \hat{w}_{t-1}(u_{t-1}) \preceq H_{t-1} \preceq (1+0.001)\nabla^2 \hat{w}_{t-1}(u_{t-1}). \tag{D.34}$$

Combining this with equation D.33 implies property equation D.6 for the base case. Furthermore, since $w_t^1 = w_{t-1}$ (by definition), equation D.8 follows trivially for the base case.

Now, using that $\phi_t(w) = \phi_{t-1}(w) + g_{t-1}w$, we have

$$\begin{aligned}
&\hat{w}_t^1(\cdot) + \tfrac{\alpha}{2}\|\cdot - x_t\|_2^2 \\
&\quad = \hat{w}_{t-1}(\cdot) + g_{t-1}(\cdot) + \tfrac{\alpha}{2}\|\cdot - x_t\|_2^2 \\
&\quad = \hat{w}_{t-1}(\cdot) + \tfrac{\alpha}{2}\|\cdot - x_{t-1}\|_2^2 + g_{t-1}(\cdot) + \tfrac{\alpha}{2}\|\cdot - x_{t-1}\|_2^2 - \tfrac{\alpha}{2}\|\cdot - x_{t-1}\|_2^2 + \tfrac{\alpha}{2}\|\cdot - x_t\|_2^2 \\
&\quad = \hat{w}_{t-1}(\cdot) + \tfrac{\alpha}{2}\|\cdot - x_{t-1}\|_2^2 + g_{t-1}(\cdot) + \tfrac{\alpha}{2}\|\cdot - x_{t-1}\|_2^2 + \alpha\langle x_{t-1} - x_t, \cdot - x_{t-1}\rangle + \tfrac{\alpha}{2}\|x_{t-1} - x_t\|_2^2 \\
&\quad \ge \hat{w}_{t-1}(\cdot) + \tfrac{\alpha}{2}\|\cdot - x_{t-1}\|_2^2 + g_{t-1}(\cdot) + \tfrac{\alpha}{2}\|\cdot - x_{t-1}\|_2^2 \qquad \text{(D.35)} \\
&\quad \ge \tfrac{\alpha}{2}\|\cdot - x_{t-1}\|_2^2 + \tfrac{\alpha}{2}b^2(1-\hat{\alpha}/2)^{2M}; \qquad \text{(D.36)}
\end{aligned}$$

where the last inequality follows by equation D.3 and the fact that $\|g_{t-1}\|_{\Sigma_{\hat{w}_{t-1}}} \le b$. This shows property equation D.4 for the base case. Thus, by Lemma 66, we have, for $w_t^\dagger = \arg\min_{w \in K} \phi_t(w)$,

$$\|w_t^1 - w_t^\dagger\|_{\Sigma_{\hat{w}_t^1}} \cdot \|w_t^1 - w_t^\dagger\|_{\nabla^2 \hat{w}_t^1} \le B\,\hat{w}_t^1(\phi_t) - \hat{w}_t^1(\phi_t^\dagger) \le B(1-\hat{\alpha})^M\,\hat{w}_t^1(\phi_t) - \hat{w}_t^1(\phi_t^\dagger) \le B\frac{1}{49M}. \tag{D.37}$$

Now, combining equation D.34 with the fact that $\|u_{t-1} - w_t^1\|_{\Sigma_{\hat{w}_{t-1}}} \cdot \|u_{t-1} - w_{t-1}\|_{\Sigma_{\hat{w}_{t-1}}} \le B(1-\hat{\alpha})^{40M}$ (see equation D.3) and Lemma 67, we obtain

$$\left(\frac{31}{32}\right)^2 H_{t-1} \preceq \nabla^2 \hat{w}_t^1(\cdot) \preceq \left(\frac{32}{31}\right)^2 H_{t-1}; \tag{D.38}$$

Plugging equation D.38 into equation D.37, we get

$$\|w_t^1 - w_t^\dagger\|_{H_{t-1}} \le B(1-\hat{\alpha})^{40M}. \tag{D.39}$$

Now, by the triangle inequality

$$\begin{aligned}
\|u_{t-1} - w_t^\dagger\|_{H_{t-1}} &\le B\|u_{t-1} - w_t^1\|_{H_{t-1}} + \|w_t^1 - w_t^\dagger\|_{H_{t-1}} \\
&\le B(1-\hat{\alpha}/2)^{20M}; \qquad \text{(D.40)}
\end{aligned}$$

where in the last inequality we used equation D.3 and equation D.39. Combining equation D.40 with equation D.34 and Lemma 66, we get

$$\hat{}4\text{-}5\bullet\copyright^2\ \hat{}w_t^{\dagger}\bullet\ \mathrm{l}\ H_{t\ 1}\ \mathrm{l}\ \hat{}5\text{-}4\bullet\copyright^2\ \hat{}w_t^{\dagger}\bullet: \tag{D.41}$$

Combining Equations equation D.41 and equation D.38 implies property equation D.5 for the base of Induction. Furthermore, note that from Lemma 66:

$$Yw_t^1\quad w_t^{\dagger}Y_{\copyright^2\ \hat{}w_t^{\dagger}}.\ B\frac{50}{49}\ \hat{}w_t^1;\ _t\bullet\ B\frac{1}{49M};$$

which shows property equation D.7 for the base case.

Induction step. Now, assume that properties equation D.4, equation D.6, equation D.7, and equation D.8 hold form C1. We will show that these properties holds form 1. From the hypothesis of induction, we have

$$Yw_t^m\quad u_{t\ 1}Y_{H_{t\ 1}}\ B\frac{1}{12M};$$

which combined with equation D.34 and Lemma 66 implies

$$0{:}84\copyright^2\ \hat{}w_t^m\bullet\ \mathrm{l}\ H_{t\ 1}\ \mathrm{l}\ 1{:}2\copyright^2\ \hat{}w_t^m\bullet: \tag{D.42}$$

Thus, by Lemma 70 (instantiated with c 1-5) and the fact that $\hat{}w_t^m;\ _t\bullet\ B\,1\text{-}\hat{}40M\ \bullet$ (by the induction hypothesis), we get that for$w_t^{m\ 1}\quad w_t^m\quad H_{t\ 1}^{\ 1}\copyright\ _t\hat{}w_t^m\bullet$:

$$\hat{}w_t^{m\ 1};\ _t\bullet\ B\,9M\quad\hat{}w_t^m;\ _t\bullet^2\quad 2{:}5c\ \hat{}w_t^m;\ _t\bullet\ B\hat{}\,7\text{-}8\bullet\ \hat{}w_t^m;\ _t\bullet:$$

Again, by Lemma 70 with c 1-5, we have

$$\begin{aligned}\hat{}w_t^{m\ 1};\ _t\bullet\ B\,20\hat{}\,1\quad c\bullet"\quad\hat{}1\quad 20\hat{}\,1\quad c\bullet"\bullet\ \hat{}w_t^{m\ 1};\ \bullet\\ B\,25'\quad\lessdot\frac{15}{16}\bullet\ \hat{}w_t^m;\ _t\bullet\ B\,1\text{-}\hat{}50M\ \bullet:\end{aligned} \tag{D.43}$$

By the induction hypothesis, we also have that $\hat{}w_t^m;\ _t\bullet\ B\,\%\frac{15}{16}\breve{Z}^{m\ 1}\ \hat{}w_t^1;\ _t\bullet\ 500'$. Combining this with equation D.43, we get

$$\hat{}w_t^{m\ 1};\ _t\bullet\ B\lessdot\frac{15}{16}\bullet^{\ m}\ \hat{}w_t^1;\ _t\bullet\ 500': \tag{D.44}$$

This shows that equation D.4 holds withm replaced bym 1.

Next, we show that equation D.6 holds withm replaced bym 1. Combining equation D.43 with Lemma 66 implies

$$Yw_t^{m\ 1}\quad w_t^{\dagger}Y_{\copyright^2\ \hat{}w_t^{\dagger}}.\ B\ \hat{}w_t^{m\ 1};\ _t\bullet\text{-}\hat{}1\quad M\quad\hat{}w_t^{m\ 1};\ _t\bullet\bullet\ B\,1\text{-}\hat{}49M\ \bullet:$$

This, together with equation D.41 gives

$$Yw_t^{m\ 1}\quad w_t^{\dagger}Y_{H_{t\ 1}}\ B\,1\text{-}\hat{}32M\ \bullet: \tag{D.45}$$

Combining equation D.40 with equation D.45 gives:

$$\|w_t^{m-1} - u_{t-1}\|_{H_{t-1}} \leq \frac{1}{12M},$$

which proves that equation D.6 holds with m replaced by $m-1$.

Next, we show that equation D.7 holds with m replaced by $m-1$. By equation D.44 and Lemma 66, we have

$$\begin{aligned}\|w_t^{m-1} - w_t^\dagger\|_{\hat{\Sigma}_t} &\leq \frac{50}{49}\left(\frac{15}{16}\right)^m \|\hat{w}_t^1 - w_t^\dagger\|_{\hat{\Sigma}_t} + 240\epsilon, \\ &\leq \frac{1}{49M}\left(\frac{15}{16}\right)^m 240\epsilon \leq \frac{1}{20M}, \end{aligned} \tag{D.46}$$

where in the last inequality we used equation D.36 and the bound on ϵ in the lemma's statement. This shows that equation D.7 holds with m replaced by $m-1$.

Next, we show that equation D.8 holds with m replaced by $m-1$. By plugging equation D.41 into equation D.46, we get

$$\|w_t^{m-1} - w_t^\dagger\|_{H_{t-1}} \leq \frac{1}{40M}\left(\frac{15}{16}\right)^m \leq 300\epsilon. \tag{D.47}$$

On the other hand, by equation D.35, we have

$$\|\hat{w}_t^1 - w_t^\dagger\|_{\hat{\Sigma}_t} \leq \frac{\sqrt{\eta_{t-1}\|\hat{w}_{t-1} - \hat{w}_{t-2}\|_{\hat{\Sigma}_{t-1}}^2 + \|\hat{w}_{t-1} - w_{t-1}^\dagger\|_{\hat{\Sigma}_{t-1}}^2}}{\sqrt{2g_{t-1} - \|\hat{w}_{t-1} - w_{t-1}^\dagger\|_{\hat{\Sigma}_{t-1}}^2}} \cdot \|\hat{w}_{t-1} - w_{t-1}^\dagger\|_{\hat{\Sigma}_{t-1}}.$$

Plugging this into equation D.37 and using equation D.36, we get

$$\begin{aligned}\|w_t^1 - w_t^\dagger\|_{\hat{\Sigma}_t} &\leq \frac{50}{49}\|\hat{w}_{t-1} - w_{t-1}^\dagger\|_{\hat{\Sigma}_{t-1}} + \|g_{t-1}\|_{\hat{\Sigma}_{t-1}^{-1}}, \\ &\leq \frac{50}{49}\|g_{t-1}\|_{\hat{\Sigma}_{t-1}^{-1}}, \end{aligned} \tag{D.48}$$

where the last inequality follows by equation D.3. Combining equation D.48 with equation D.42 (instantiated with $m-1$),

$$\|w_t^1 - w_t^\dagger\|_{H_{t-1}} \leq 2\|g_{t-1}\|_{\hat{\Sigma}_{t-1}^{-1}}. \tag{D.49}$$

Now, by equation D.47, equation D.49, and the triangle inequality, we have

$$\|w_t^1 - w_t^{m-1}\|_{H_{t-1}} \leq 2\|g_{t-1}\|_{\hat{\Sigma}_{t-1}^{-1}} + \frac{1}{40M}\left(\frac{15}{16}\right)^m \leq 500\epsilon \leq 2.$$

Via another triangle inequality, we get

$$\begin{aligned}\|u_t - w_t^{m-1}\|_{H_{t-1}} &\leq \|u_{t-1} - w_{t-1}\|_{H_{t-1}} + \|w_{t-1} - w_t^{m-1}\|_{H_{t-1}} \\ &\leq 2\|g_{t-1}\|_{\hat{\Sigma}_{t-1}^{-1}} + \frac{1}{40M}\left(\frac{15}{16}\right)^m \\ &\leq 500\epsilon \leq 2. \end{aligned}$$

This shows that equation D.8 holds with m replaced by $m-1$. Finally, combining equation D.41 with equation D.42 implies

$$\frac{1}{2}\|\hat{w}_t^m - w_t^\dagger\|_{\hat{\Sigma}_t}^2 \leq \|\hat{w}_t^\dagger - w_t^\dagger\|_{\hat{\Sigma}_t}^2 \leq 2\|\hat{w}_t^m - w_t^\dagger\|_{\hat{\Sigma}_t}^2,$$

which completes the proof. □

D.5 Helper Lemmas

Lemma 73 (Bounding norm of the Newton step) Let $y > K$ and $H > R^{d\ d}$ be such that $\hat{}1\ c\bullet\copyright^2\ \hat{}y\bullet\ l\ H\ l\ \hat{}1\ c\bullet\copyright^2\ \hat{}y\bullet$. Then, for $h\quad H\ ^1\copyright\ \hat{}y\bullet$, we have

$$Yh Y_{\copyright^2\ \hat{}y\bullet} B \frac{1}{1\ c} Y\copyright\ \hat{}y\bullet Y_{\copyright\ ^2\ \hat{}y\bullet}:$$

Proof. We can write

$$Yh Y_{\copyright^2\ \hat{}y\bullet} \quad YH\ ^1\copyright\ \hat{}y\bullet Y_{\copyright^2\ \hat{}y\bullet} \\ \text{»} \frac{}{\copyright\ \hat{}y\bullet \text{—} H\ ^1\copyright^2\ \hat{}y\bullet H\ ^1\copyright\ \hat{}y\bullet} \\ ¾ \frac{}{\copyright\ \hat{}y\bullet \text{—}\copyright^2\ \hat{}y\bullet\ ^{1\sim2} \check{S}\copyright^2\ \hat{}y\bullet^{1\sim2} H\ ^1\copyright^2\ \hat{}y\bullet^{1\sim2}\bullet^{\ 2}\copyright^2\ \hat{}y\bullet\ ^{1\sim2}\copyright\ \hat{}y\bullet:} \quad \text{(D.50)}$$

For the middle matrix $\copyright^2\ \hat{}y\bullet^{1\sim2} H\ ^1\copyright^2\ \hat{}y\bullet^{1\sim2}$ we have that

$$\frac{1}{1\ c} l\ l\ \copyright^2\ \hat{}y\bullet^{1\sim2} H\ ^1\copyright^2\ \hat{}y\bullet^{1\sim2} l\ \frac{1}{1\ c} l;$$

since $\hat{}1\ c\bullet\copyright^2\ \hat{}y\bullet\ l\ H\ l\ \hat{}1\ c\bullet\copyright^2\ \hat{}y\bullet$ by assumption. Plugging this back into equation D.50, we get

$$Yh Y_{\copyright^2\ \hat{}y\bullet} B \frac{1}{1\ c} Y\copyright\ \hat{}y\bullet Y_{\copyright^2\ \hat{}y\bullet\ ^1}:$$

□

Lemma 74 (Cauchy-Schwarz) If $B\ l\ A\ l\ B$ are symmetric matrices and B is PSD, then

$$x \text{—} Ay\ B\ Yx Y_B\ Yy Y_B:$$

Appendix E

Online Learning with Temporal Feedback Graphs

E.1 Transitive feedback graphs

In this appendix, we turn our attention to the case of transitive feedback graphs S. Recall that these are acyclic feedback graphs S where if $u > S_t$ and $v > S_u$, then $v > S_t$ (that is, if you can see loss ℓ_u at round t, you can also see all losses you could see at round u).

E.1.1 An efficient algorithm

We will begin by showing that we can efficiently implement Algorithm 2 for any transitive feedback graph S. In particular, it suffices to show that we can efficiently find an optimal basis B and associated primal solution $\alpha^\ddagger$ in time polynomial in T.

Note that for a transitive feedback graph S, any directed path $t_1 \quad t_2 \quad \cdots \quad t_w$ in S forms an order. This allows us to construct an efficient separation oracle (and hence efficiently solve) the upper bound dual program equation 4.7.

Lemma 75. If S is a transitive feedback graph, then we can find an optimal solution $\alpha^\ddagger$ to the upper bound dual program (to within additive error) in time $\mathrm{poly}(T, 1/\delta)$. Moreover, there exists an efficiently computable subgraph S^{core} of S and two sets of rounds $\mathrm{src}(S^{\text{core}})$ and $\mathrm{dest}(S^{\text{core}})$ such that $C(\alpha^\ddagger)$ is equal to the set of all directed paths contained within S^{core} that start at a node in $\mathrm{src}(S^{\text{core}})$ and end at a node in $\mathrm{dest}(S^{\text{core}})$.

Proof. We will first provide an efficient separation oracle which, given an $\alpha \geq 0$, will either describe which of the order constraints $\alpha_c > N$ violates, or report that α is a valid dual solution. With such an oracle, we can use the ellipsoid method to solve the convex program to within δ additive error in time $\mathrm{poly}(T, 1/\delta)$.

Since every directed path in S corresponds to some order C_c, it suffices to be able to find the directed path P in S which maximizes $\sum_{t \in P} \frac{2}{t}$ (if this maximum is larger than 1, we have a violating constraint given by the order P corresponds to). But this is equivalent to

computing the longest weighted path in a directed acyclic graph (where vertex t has weight $\frac{2}{t}$), which can be solved efficiently in $O(T^2)$ time via dynamic programming.

Now, note that $C(\ddagger)$ consists of all directed paths P where $\sum_{t \in P} \frac{2}{t} \geq 1$. We can use the same dynamic program to show that $C(\ddagger)$ simply contains all paths in a subgraph $S^{(\ddagger)}$ of S that start in a source set $src(S^{(\ddagger)})$ and end in a target set $dest(S^{(\ddagger)})$. For each node $t > T$, let V_t be the maximum value of $\sum_{s \in P} \frac{2}{s}$ over any directed path P ending at t. We can use the following procedure to construct $S^{(\ddagger)}$:

- Start by adding the set of vertices t where $V_t \geq 1$ to $S^{(\ddagger)}$.
- For every round t in $S^{(\ddagger)}$ that hasn't been processed, find all ancestors $s > S_t$ with the property that $V_t - V_s \geq \frac{2}{t}$. For each such ancestor s, add the edge $s \to t$ to $S^{(\ddagger)}$ (i.e., add s to $S_t^{(\ddagger)}$), and process s if it has not already been processed.
- Finally, let $src(S^{(\ddagger)})$ equal the set of rounds $t > S^{(\ddagger)}$ where $V_t \geq \frac{2}{t}$, and let $dest(S^{(\ddagger)})$ equal the set of rounds t in $S^{(\ddagger)}$ where $V_t \geq 1$ (equivalently, these are the sources and sinks of the DAG $S^{(\ddagger)}$).

To see why this procedure works, note that any path $P = (t_1, t_2, \ldots, t_k)$ from $src(S^{(\ddagger)})$ to $dest(S^{(\ddagger)})$ along edges of $S^{(\ddagger)}$ will satisfy $\sum_{t \in P} \frac{2}{t} \geq \sum_{i=1}^{k-1} (V_i - V_{i+1}) = V_k - 1$. Moreover, any edge this algorithm does not select cannot possibly be included in a path in $C(\ddagger)$ (the sum of $\frac{2}{t}$ over a path from t_i to t_j containing this edge must be strictly less than $V_j - V_i + 1$). It follows that $C(\ddagger)$ simply contains all source-destination paths in $S^{(\ddagger)}$. □

We can now use the dual solution (and the characterization of $C(\ddagger)$) provided by Lemma 75 to find an efficient, sparse solution to the primal.

Theorem 44. If S is a transitive feedback graph, then we can find an optimal solution $\ddagger$ to the upper bound program (to within additive error) supported on a basis B of size $|B| \leq |S| + T$ in time $\text{poly}(T, 1/\epsilon)$.

Proof. By Lemma 75, we can efficiently construct a dual solution $\ddagger$ and corresponding set $C(\ddagger)$. Our approach will be to use this to find a sparse solution to the linear program equation 4.10 in the proof of Lemma 9 in the c_t random variables (recall that such a solution characterizes an optimal primal solution via $\ddagger_{c,t} = c_t \ddagger_t$ for each $c > C(\ddagger)$, $t > C_c$).

The linear program in equation 4.10 has $|C(\ddagger)|$ random variables, and therefore it would be inefficient to solve directly. Instead, we will show that we can use the structure of $C(\ddagger)$ (as the set of all source-destination paths in the subgraph $S^{(\ddagger)}$) to rewrite it as a flow problem. Indeed, consider the following linear program in the variables f_e (for each edge $e = (s,t)$ in the edge set $E(S^{(\ddagger)})$ of $S^{(\ddagger)}$):

$$\text{Minimize} \quad \sum_{t>\text{dest}(s);(s,t)\in E(S)} f_{s,t} \tag{E.1}$$

$$\text{Subject to} \quad \sum_{s:(s,t)\in E(S)} f_{s,t} \geq 1-\epsilon_t \quad \text{for all } t > T$$

$$\sum_{s:(s,t)\in E(S)} f_{s,t} \geq \sum_{s':(t,s')\in E(S)} f_{t,s'} \quad \text{for all } t > \text{src}(S) \text{ and } t < \text{dest}(S)$$

$$f_{s,t} \geq 0 \quad \text{for all } (s,t) \in E(S).$$

The linear program equation E.1 is a polynomial-sized LP, so we can solve it efficiently. At the same time, it is equivalent to the linear program equation 4.10 in the following sense: first, given any solution α_c to equation 4.10, if we let $f_e = \sum_{c:e\in c} \alpha_c$ equal the sum of α_c over all paths C_c that contain e, then f_e satisfies equation E.1 (with the same objective value). Conversely, given any flow f_e solving equation E.1, we can decompose it into a positive combination of source-destination paths. If we let α_c be the weight of the path C_c in this decomposition, we can likewise check that α_c satisfies equation 4.10 (also with the same objective value).

There are well-known efficient procedures for flow-decomposition (see e.g. [331]), which take a flow f_e and return a positive combination of at most $O(|E(S)|)$ paths. By doing this, we can obtain an optimal solution α_c to equation 4.10 supported on a basis of size at most $O(T)$. We can then shrink this to an optimal basis B of size at most T by removing all other variables from the LP equation 4.10 and using any method for finding a basic feasible solution to the resulting program (e.g. optimizing a random linear functional over the face containing the optimum). □

Theorem 44 implies that given any transitive feedback graph S, we can efficiently construct the algorithm of Corollary 3 that runs in time $O(KT)$ per iteration and incurs regret at most $O(UB(S)\sqrt{\log K})$.

E.1.2 An asymptotically tight lower bound

Finally, we prove a lower bound of $\Omega(UB(S))$ on the regret of any algorithm when the feedback graph S is transitive. This combined with our upper bound in Corollary 3 settles the optimal learning rate for the case of transitive graphs.

Theorem 45 (Tight lower bound for transitive S). *For any transitive graph S, every S-learning algorithm A must incur worst-case regret $\text{Reg}(A) \geq UB(S)/50$.*

Proof. To show this lower bound, we use the fact that the value of the upper bound program $UB(S)$ is equal to its dual equation 4.7. Then, for every instance $(\epsilon_t)_{t=1}^T$ of equation 4.7, we show $\Omega(\sum_{t=1}^T \epsilon_t)$ as a lower bound for the value of the regret. To show this lower bound, we take a similar strategy as in the proof of Theorem 11; the adversary flips a coin $B \sim \{-1, 1\}$ and then consider loss vectors $\ell_t = (X_t, \frac{1}{2})$ where X_t is a $\frac{1}{2}-\epsilon_t$-biased Bernoulli variable, where the bias is to one if $B = 1$ and to zero if $B = -1$. Here again the adversary attempts to use shared randomness between ℓ_t's to block the chances of the player to learn B. At the same

time, the adversary has to be careful not to reveal any information about the $\grave{e}_t$ given the loss vectors $\grave{}_s$'s, $s > S_t$ which it can observe at time t. Here, the adversary exploits the transitive nature of S to cook up these random variables and considers a linearly shifted Brownian motion with rate :

$$L_t \quad Bt \quad B_t:$$

The adversary de nes each X_t as a positivity indicator variable of a chunk of the process L. Namely, for times p_t $C q_t$,

$$X_t \quad 1\tilde{}L_{p_t} \quad L_{q_t} C0\bullet$$

Speci cally, for each time $t > T$, the adversary de nes

$$q_t \quad \max_{s>S_t}\tilde{} p_s\bullet;$$
$$p_t \quad q_t \quad {}^2_t:$$

With this de nition, rst we show by induction that p_t B 1 for all t. We strengthen the hypothesis of induction and concurrently show the argument that for any t, there is an ordered clique p in S ending at t and with $P_{s>p} {}^2_s \quad p_t$. The hypothesis of induction is trivial since $q_1 \quad 0$ and $p_1 \quad {}^2_1$. Now for arbitrary t B T, let $s > S_t$ be the index with maximum p_s in S_t. From the hypothesis of induction we know there is an ordered clique $p^{\text{œ}}$ ending at s with $P_{s^{\text{œ}}>p} {}^2_{s^{\text{œ}}} \quad p_s$. Now from de nition we have $p_t \quad p_s \quad {}^2_t$. Therefore, $p_t \quad P_{s>p^{\text{œ}}} {}^2_s$ where $p^{\text{œ}}$ is the ordered clique of p concatenated with t. This shows the step of induction. Finally, note that from the argument that we showed, that there is an ordered clique p ending at t with $P_{s>p} {}^2_s \quad p_t$, we conclude $p_t \quad P_{s>p} {}^2_s$ B 1 because of the constraint in the dual of the upper bound program.

The second observation is that with this de nition, X_t becomes independent of X_s for $s > S_t$. This follows from the independence of disjoint increments of Brownian motion. Next, we show that with small enough choice of constant , the adversary cannot distinguish between B 1 cases with constant probability. For this, similar to the proof of Theorem 11 it is enough to bound the total variation distance between $Q\ {}^1\hat{}8_{s>S_t}\hat{}L_{p_s} \quad L_{q_s}\bullet\bullet$ and $Q\ {}^1\hat{}8_{s>S_t}\hat{}L_{p_s} \quad L_{q_s}\bullet\bullet$, where here we use the notation $Q\ \hat{}X\bullet$ and $Q\ \hat{}X\bullet$ to refer to the distribution of the random variable (or more generally random process) X given B 1 and B 1, respectively. But again from the data processing inequality, we have

$$TV\check{S}Q\ \hat{}8_{s>S_t}\hat{}L_{p_s} \quad L_{q_s}\bullet\bullet; Q\ \hat{}8_{s>S_t}\hat{}L_{p_s} \quad L_{q_s}\bullet\bullet\bullet \tag{E.2}$$
$$\text{½}\ \_\_\_\_\_$$
$$\text{B} \quad D\check{S}Q\ \hat{}8_{s>S_t}\hat{}L_{p_s} \quad L_{q_s}\bullet\bullet S Q\ \hat{}8_{s>S_t}\hat{}L_{p_s} \quad L_{q_s}\bullet\bullet\bullet \tag{E.3}$$
$$\text{½}\ \_\_\_\_\_$$
$$\text{B} \quad D\check{S}Q\ \%\text{ı}\text{ı}\ 0\ 1\ \check{Z}SQ\ \%\text{ı}\text{ı}\ 0\ 1\ \check{Z}\bullet; \tag{E.4}$$

where in the last Equation, $Q\ \hat{}L\ 0\ 1\bullet$ refers to the distribution corresponding to the whole sample path of the process L in the interval $0;1$ and we used the fact that for all times s, $0 \text{B} q_s \text{B} p_s \text{B} 1$.

Next, we use the following Lemma, proved in Appendix E.3.3, to upper bound the RHS in Equation equation E.4. This Lemma provides a formula for the KL divergence of two shifted Brownian motions.

Lemma 76 (KL divergence between shifted Brownian motions)For linearly shifted Brownian motions $L_t + t$ B_t, the KL divergence between the measures corresponding L_t to and B_t in the interval $[0;1]$ is equal to

$$D(Q(L_{[0,1]}) \| Q(B_{[0,1]})) = \mu^2/2.$$

Applying Lemma 76, we get

$$D(Q(L_{[0,1]}) \| Q(B_{[0,1]})) = 2\mu^2,$$

Plugging this into Equation equation E.4:

$$TV(Q(\cap_{s>S_t}(L_{p_s} \neq L_{q_s})), Q(\cap_{s>S_t}(L_{p_s} \neq L_{q_s}))) \leq \sqrt{2}.$$

Therefore, similar to the proof of Theorem 13,

$$E[(x_t - x_t^*)^2 \mid C_{t-1}] \geq \frac{t}{4}.$$

by picking $\mu = 1/4$. Moreover, from Lemma 78 we have $q_t - p_t \geq \frac{\sqrt{t}}{2}$. Therefore,

$$\mathrm{Reg}(A) \geq \sum_{t=1}^{T} \frac{\sqrt{t}}{4} \cdot \frac{\sqrt{t}}{2} = \frac{P(T)}{16} \geq \frac{UB(S)}{16}.$$

□

We briefly remark on the connection between Theorem 45 and the independent set program of Section 4.4.2. Although we have presented Theorem 45 in a completely self-contained way, we can view the construction in the proof of this theorem as constructing a feasible point to the independent set program.

Indeed, in the proof of the above theorem, we associate to each round $t \in T$ an interval $[p_t, q_t]$ contained within the unit interval. The intersection of all these intervals induces a partition of the unit interval into sub-intervals, each of which is labeled with a subset of rounds of T. In addition, by our construction of these intervals, two intervals $[p_s, q_s]$ and $[p_t, q_t]$ can intersect only if s and t are not adjacent in S. Therefore, each sub-interval is actually labeled with some independent set I belonging to $I(S)$. Taking the weight w_I to be the square root of the length of this interval, the analysis in the above proof implies that w_I form a feasible solution to the independent set program with value equal to $P(T) = UB(S)$.

E.2 The lower bound $LB(S)$ is not tight

In this appendix, we show that the gap between the value of the upper bound program $UB(S)$ and the lower bound program $LB(S)$ can grow without bound.

Theorem 46. For any T, there exists a temporal feedback graph S on T rounds where $UB(S) - LB(S) = \Omega(T^{1/4})$.

Proof. Consider the feedback graph S formed by batched learning setting where the time horizon T is divided into $\sqrt{T}$ batches of $\sqrt{T}$ rounds each (so, S_t only contains rounds s @ $t - \sqrt{T} \ldots t$).

Since each order in S contains at most $\sqrt{T}$ rounds, one feasible solution for the upper bound dual program is to set $\alpha_t = T^{1/4}$ for all $t > T$, which implies $UB(S) \le T^{3/4}$.

On the other hand, note that in the lower bound program, we have that $\sum_{t : S_T \ni t} \alpha_t^2 \ge 1$. Since S_t contains at most T elements, this implies that $\sum_{t : S_T \ni t} \alpha_t \ge \sqrt{T}$. But on the other hand, S_T can see all the rounds except the rounds in the very last batch (of which there are at most $\sqrt{T}$). So $\sum_{t \notin S_t} \alpha_t \ge \sqrt{T}$, and therefore $LB(S) \ge \sum_t \alpha_t \ge 2\sqrt{T}$.

It follows that $UB(S) - LB(S) \ge T^{1/4} - 2 = \Omega(T^{1/4})$, as desired. □

E.3 Lemmas from probability and information theory

In this appendix, we establish some standard results from probability and information theory that we make use of in our proofs of our lower bounds.

E.3.1 Bound of KL-divergence for Bernoulli random variables

Lemma 77. Fix α such that $0 \le \alpha \le 1/4$. If $Q = \mathrm{Bern}(1/2 + \alpha)$, $Q' = \mathrm{Bern}(1/2 - \alpha)$, then $D(Q \| Q') \le 12\alpha^2$.

Proof. Computing the KL divergence explicitly, we have that:

$$\begin{aligned} D(Q \| Q') &= \left(\frac{1}{2} + \alpha\right) \log \frac{\frac{1}{2} + \alpha}{\frac{1}{2} - \alpha} + \left(\frac{1}{2} - \alpha\right) \log \frac{\frac{1}{2} - \alpha}{\frac{1}{2} + \alpha} \\ &= 2\alpha \left(\log(1 + 2\alpha) - \log(1 - 2\alpha)\right) \\ &\le 2\alpha \left(6\alpha + 12\alpha^2\right). \end{aligned}$$

Here we have used the fact that $\log(1 + x) - \log(1 - x) \le 3x$ for $x \in (0, 1/2]$. □

E.3.2 From Gaussian to biased Bernoulli

Lemma 78. Given a Gaussian variable $Y \sim \mathcal{N}(c, 1)$ with $c > 0$, and Bernoulli variable $X = \mathbb{1}_{Z > 0}$, we have

$$P(X = 1) \geq 1/2 + c/\sqrt{2\pi}.$$

Proof. Note that

$$\begin{aligned} P(Z > 0) &= \frac{1}{2} \int_0^c \frac{1}{\sqrt{2\pi}} e^{-(z-c)^2/2} \\ &\quad + \frac{1}{2} \int_0^c \frac{1}{\sqrt{2\pi}} e^{-c^2/2} \\ &\quad + \frac{1}{2} \frac{c}{\sqrt{2\pi}} (1 - c^2/2) \\ &\quad + \frac{1}{2} \frac{c}{2}. \end{aligned}$$

This completes the proof. □

E.3.3 KL divergence in Gaussian processes

Lemma 79 (KL divergence between two Gaussians)

$$D(\mathcal{N}(\mu_1, 1) \| \mathcal{N}(\mu_2, 1)) = (\mu_1 - \mu_2)^2/2.$$

Proof. We can write

$$\begin{aligned} D(\mathcal{N}(\mu_1, 1) \| \mathcal{N}(\mu_2, 1)) &= E_{\mathcal{N}(\mu_1, 1)} \left[\ln \frac{e^{-(y-\mu_1)^2/2}}{e^{-(y-\mu_2)^2/2}} \right] \\ &= E_{\mathcal{N}(\mu_1, 1)} [(y-\mu_2)^2/2 - (y-\mu_1)^2/2] \\ &= (\mu_1 - \mu_2)^2/2. \end{aligned}$$

□

Lemma 80 (Restatement of Lemma 76) For a linearly shifted Brownian motion $L_t = t + B_t$, the KL divergence between the measures corresponding to L_t and B_t in the interval $[0, 1]$ is equal to

$$D(Q[L_{[0,1]}] \| Q[B_{[0,1]}]) = 1^2/2.$$

Proof. From the Girsanov theorem applied to the exponential martingale of the process B_t, we have

$$\frac{dQ[B_{[0,1]}]}{dQ[L_{[0,1]}]} = e^{\int_0^1 dB_t - \frac{1}{2}\int_0^1 1^2 dt},$$

which implies

$$D(Q[L_{[0,1]}] \| Q[B_{[0,1]}]) = E_{B_{[0,1]}} \left[\int_0^1 dB_t - \frac{1}{2} \int_0^1 1^2 dt \right] = \frac{1^2}{2}.$$

□

E.4 Omitted proofs

E.4.1 Proof of Theorem 10

Proof. We will show that the upper bound dual program arises from Lagrangifying the original upper bound program (and thus this equality follows as a consequence of strong duality).

We will begin by weakening the upper bound program equation 4.3 by replacing the strict equality $\sum_{c=1}^{N} \gamma_{c,t} = 1$ with the weak inequality $\sum_{c=1}^{N} \gamma_{c,t} \leq 1$. Note that this does not change the optimal value of the LP (as if this inequality is strict for a specific t, one can always improve the objective by decreasing one of the non-zero $\gamma_{c,t}$ while not altering any of the other constraints). By Lagrangifying these constraints, we have that

$$UB(S) = \min_{\gamma \geq 0} \max_{\alpha \geq 0} \left(\sum_{c=1}^{N} \sqrt{\sum_{t \in C_c} \gamma_{c,t}^2} + \sum_{t=1}^{T} \alpha_t \left(1 - \sum_{c=1}^{N} \gamma_{c,t} \right) \right) \tag{E.5}$$

Now, note that in this convex program the objective is convex, all the constraints are affine, and the program is always feasible (for every round t there is at least one order containing it, namely the singleton order (t)). Therefore we can apply the theorem of strong duality (see Chapter 28 of [332]), and interchange the order of minimum and maximum in equation E.5.

$$UB(S) = \max_{\alpha \geq 0} \min_{\gamma \geq 0} \left(\sum_{c=1}^{N} \sqrt{\sum_{t \in C_c} \gamma_{c,t}^2} + \sum_{t=1}^{T} \alpha_t \left(1 - \sum_{c=1}^{N} \gamma_{c,t} \right) \right) \tag{E.6}$$

We can in turn rewrite equation E.6 in the following form:

$$UB(S) = \max_{\alpha \geq 0} \left(\sum_{t=1}^{T} \alpha_t + \min_{\gamma \geq 0} \sum_{c=1}^{N} \left(\sqrt{\sum_{t \in C_c} \gamma_{c,t}^2} - \sum_{t \in C_c} \gamma_{c,t} \alpha_t \right) \right) \tag{E.7}$$

Note that the terms in the internal sum pertaining to C_c only depend on the variables $\gamma_{c,t}$. We can therefore interchange the order of this sum and min in equation E.7 and obtain:

$$UB(S) = \max_{\alpha \geq 0} \left(\sum_{t=1}^{T} \alpha_t + \sum_{c=1}^{N} \min_{\gamma_c \geq 0} \left(\sqrt{\sum_{t \in C_c} \gamma_{c,t}^2} - \sum_{t \in C_c} \gamma_{c,t} \alpha_t \right) \right) \tag{E.8}$$

(where γ_p represents the $|C_c|$ variables of the form $\gamma_{c,t}$).

Let us now consider each of these inner optimization problems of the form

$$\min_{\gamma_c \geq 0} \left(\sqrt{\sum_{t \in C_c} \gamma_{c,t}^2} - \sum_{t \in C_c} \gamma_{c,t} \alpha_t \right).$$

Note that if we restrict the domain of $_c$ to the subdomain where $P_{t>C_c} \ ^2_{c;t} \ R^2$, this expression is minimized when each $_{c;t}$ is proportional to $_t$. Speci cally, it is minimized when

$$_{c;t} \ \frac{t}{P_{t>C_c} \ ^2_{t}} \ R;$$

at which point the expression has value equal to

$$R \ 1 \ \frac{}{Q_{t>C_c} \ ^2_t}:$$

We therefore have two cases depending on our choice of

- If $P_{t>C_c} \ ^2_t \ 1$, then the value of this inner minimization problem is (we can set R arbitrarily large).
- Otherwise, the value of this inner minimization problem is 0, attained when $R \ 0$.

It follows that any feasible (that causes the outer-maximization problem to have nite value) must satisfy $P_{t>C_c} \ ^2_t \ 1$ for each $c > N$. If is feasible, then the value of the inner expression is just $P^T_{t \ 1} \ _t$. But note that this is precisely a description of the upper bound dual program equation 4.7. The result follows. □

E.4.2 Proof of Lemma 8

Proof. As in the proof of Theorem 10, we write the Lagrangi ed objective:

$$L(\ ; \ ) \ \sum_{c \ 1}^{N} \ \frac{}{Q_{t>C_c} \ ^2_{c;t}} \ \sum_{t \ 1}^{T} \ _t \ \sum_{c \ 1}^{N} \ _{c;t} \ \sum_{t \ 1}^{T} \ _t \ \sum_{c \ 1}^{N} \ \frac{}{Q_{t>C_c} \ ^2_{c;t}} \ \sum_{t>C_c} \ _{c;t}:$$

From the properties of strong duality, we know that any optimal primal solution ‡ must satisfy $L(\ ^{\ddagger}; \ ^{\ddagger}) \ L(\ ; \ ^{\ddagger})$ for any other 0 (not necessarily feasible). Assume to the contrary that the inequality equation 4.8 holds but we have that $^{\ddagger}_{c;t} \ 0$ for some t. Then by Cauchy-Schwartz, we have that

$$\sum_{t>C_c} \ _{c;t} \ \frac{}{Q_{t>C_c} \ ^2_t} \ \frac{}{Q_{t>C_c} \ ^2_{c;t}} \ \frac{}{Q_{t>C_c} \ ^2_{c;t}}:$$

It follows that if we construct $^{\circ}$ by taking ‡ and setting $^{\circ}_{c;t} \ 0$ for all $t > T$, then $L(\ ^{\ddagger}; \ ^{\ddagger}) \ L(\ ^{\circ}; \ ^{\ddagger})$, contradicting our assumption. Similarly, if the equality equation 4.9 holds but $^{\ddagger}_{c;t}$ is not proportional to $_t$, then if we set $^{\circ}_{c;t} \ _t \ \frac{}{P_{t>C_c} \ ^2_{t^{\circ}c}} \ P_{t>C_c} \ ^2_{t^{\circ}}$, we again nd that $L(\ ^{\ddagger}; \ ^{\ddagger}) \ L(\ ^{\circ}; \ ^{\ddagger})$, contradicting our assumption. □

E.4.3 Proof of Theorem 11

Proof. Let $_t$ be an optimal solution to the lower bound program equation 4.11. Set $1~10$, and consider the distribution over loss vectors$\grave{s}$ induced by the process described above (where the adversary rst selects$B$ uniformly from $\tilde{}$ $1;1\bullet$, selects each$\hat{X}_t$ independently from Bern$\hat{}1~2$ B $_t\bullet$, and sets$\grave{}_t$ $\hat{}X_t;1~2\bullet$). We will show that when faced with a loss vector$\grave{}$ sampled from this process, any$S$-learning algorithm A must incur $\hat{}LB\hat{}S\bullet\bullet$ expected regret (from which it follows that there exists a speci c loss vector$\grave{}$ on which it incurs this much regret).

To prove this, note that when A is deciding the actionx_t to take at time t, it can see the variables X_s for $s > S_t$ we will denote this set of random variables asX_{S_t}. If B 1, these r.v.s are distributed according to a distributionQ_{S_t}, and if B 1, these r.v.s are distribution according to a di erent distribution Q_{S_t}. We will upper bound the KL divergence $D\hat{}Q_{S_t}$ $\tilde{O}Q_{S_t}\bullet$; this in turn will allow us to apply Pinsker's inequality to upper bound the probability A can successfully distinguish between the cases$B$ 1 and B 1, from which we can lower bound the regret incurred by$A$.

SinceQ_{S_t} is a product distribution overS_t independent Bernoulli r.v.s (and likewise for Q_{S_t}), by the chain rule for KL divergence we have that

$$D\hat{}Q_{S_t}\ \tilde{O}Q_{S_t}\bullet\ \ \underset{s>S_t}{Q}\ D\hat{}Q_s\ \tilde{O}Q_s\bullet;$$

whereQ_s is simply the Bernoulli distribution Bern$\hat{}1~2$ $_s\bullet$ and Q_s is the oppositely biased Bernoulli distribution Bern$\hat{}1~2$ $_s\bullet$. We can compute that for these distributions,$D\hat{}Q_s$ $\tilde{O}$ $Q_s\bullet$ B 12 ^{2} ^2_s (see Lemma 77 in the Appendix), so it follows that$D\hat{}Q_{S_t}$ $\tilde{O}Q_{S_t}\bullet$ B $P_{s>S_t}$ 12 ^{2} ^2_s, which in turn is at most $0{:}12$ (since $0{:}1$ and $_s$ satisfy equation 4.11).

By Pinsker's inequality, the total variation distance betweenQ_{S_t} and Q_{S_t} is therefore at most

$$\frac{1}{4}\sqrt{\hat{}1~2\bullet D\hat{}Q_{S_t}\ \tilde{O}Q_{S_t}\bullet}\ \ B\ \ \sqrt{0{:}06}\ B\ 0{:}3:$$

Now, consider the probability$x_t > 0;1$ that A plays the rst action at round t. Let $x^{\ddagger}_t$ 0 if B 1 and $x^{\ddagger}_t$ 1 if B 1. Then the expected regret$A$ incurs in this round against this adversary is$\hat{}X_t$ $1~2\bullet\hat{}x_t$ $x^{\ddagger}_t\bullet$. In expectation, this is at least $_t E Sx_t$ $x^{\ddagger}_t S$. But (applying Le Cam's method),$E Sx_t$ $x^{\ddagger}_t S$ $\hat{}1~2\bullet E\ x_t\ SB$ 1 $\hat{}1~2\bullet E\ 1$ $x_t\ SB$ 1 $\hat{}1~2\bullet$ $\hat{}1~2\bullet\hat{}E\ x_t\ SB$ 1 $E\ x_t\ SB$ $1\bullet$. Sincex_t depends solely on$\hat{X}_{S_t}$ (which are drawn from Q_{S_t} when B 1 and from Q_{S_t} when B 1), this second term is at most the total variation distance in magnitude, and thus$E Sx_t$ $x^{\ddagger}_t S$ $C0{:}5$ $0{:}5$ $0{:}3$ $C0{:}3$.

It follows that the expected regret$A$ incurs this round is at least$0{:}3$ $_t$ C $_t~100$. Over all rounds, the expected regret$A$ incurs is therefore at leastP_t $_t~100$ $LB\hat{}S\bullet~100$, as desired. □

E.4.4 Proof of Theorem 13

Proof. The basic idea is similar to the proof of Theorem 11; the adversary again begins by sampling uniformly a random bit $B \in \{-1, 1\}$. For each independent set $I \in \mathcal{I}$, the adversary samples a Gaussian random variable $Y_I \sim \mathcal{N}(B w_I^2, w_I^2)$ (for some fixed $\alpha > 0$ to be decided later). Then, for each time t, she defines the Gaussian random variable Z_t as the sum of Gaussian random variables Y_I of the independent sets I that include t (letting $\mathcal{I}_t = \{I \in \mathcal{I} : t \in I\}$ denoting this sub-collection of independent sets):

$$Z_t = \sum_{I \in \mathcal{I}_t} Y_I.$$

Note that the mean and variance of Z_t is equal to the sum of the means and variances of the Y_I r.v.s, which are $B \sum_{I \in \mathcal{I}_t} w_I^2$ and $\sum_{I \in \mathcal{I}_t} w_I^2$, respectively. Finally, the adversary defines the loss at time t as $\ell_t(\hat{x}_t, x_t) = |\hat{x}_t - x_t|$, where $x_t = \frac{1}{2}(1 - Z_t) \in [0, 1]$ is a Bernoulli random variable with some bias μ_t.

By Lemma 78 (proved in Appendix E.3), we can bound $|\mu_t| \leq \frac{1}{2}\sqrt{\sum_{I \in \mathcal{I}_t} w_I^2}$ from our mean / variance calculations for Y_I.

Let Q_t^+ and Q_t^- be the distribution of X_t given $B = 1$ and $B = -1$, respectively. For a subset of times $S \subseteq [T]$, let Q_S^+ and Q_S^- be the product of distributions Q_t^+ and Q_t^- for all $t \in S$, respectively. The first component of the proof, similar to the proof of Theorem 11, is to bound the KL divergence $D(Q_{S_t}^+ \| Q_{S_t}^-)$. But by the data processing inequality, we can relate it to the distribution of the Y_I:

$$D(Q_{S_t}^+ \| Q_{S_t}^-) \leq D\left(\bigotimes_{S_t \cap I \neq \emptyset} Q^{+}(I) \,\middle\|\, \bigotimes_{S_t \cap I \neq \emptyset} Q^{-}(I)\right),$$

where $Q^{+}(I)$ and $Q^{-}(I)$ refer to the distribution of Y_I given $B = 1$ and $B = -1$, respectively. Now from the independence of the Y_I,

$$D\left(\bigotimes_{S_t \cap I \neq \emptyset} Q^{+}(I) \,\middle\|\, \bigotimes_{S_t \cap I \neq \emptyset} Q^{-}(I)\right) = \sum_{S_t \cap I \neq \emptyset} D(Q^{+}(I) \| Q^{-}(I)).$$

But defining P_s^+ and P_s^- to be the distributions of Z_s given $B = 1$ and $B = -1$, the data processing inequality implies

$$D(Q^{+}(I) \| Q^{-}(I)) \leq D(P^{+}(I) \| P^{-}(I)) = 4\alpha^2 w_I^2,$$

where the last equality follows from the formula for the KL divergence between two Gaussians, as we state in Appendix E.3. Hence,

$$D\left(\bigotimes_{S_t \cap I \neq \emptyset} Q^{+}(I) \,\middle\|\, \bigotimes_{S_t \cap I \neq \emptyset} Q^{-}(I)\right) \leq 4\alpha^2 \sum_{S_t \cap I \neq \emptyset} w_I^2.$$

Therefore, due to Pinsker's inequality, the total variation distance between $\bigotimes_{S_t \cap I \neq \emptyset} Q^{+}(I)$ and $\bigotimes_{S_t \cap I \neq \emptyset} Q^{-}(I)$ is at most

$$\mathrm{TV}\left(\bigotimes_{S_t \cap I \neq \emptyset} Q^{+}(I), \bigotimes_{S_t \cap I \neq \emptyset} Q^{-}(I)\right) \leq 2\alpha \sqrt{\sum_{S_t \cap I \neq \emptyset} w_I^2} \leq 2\alpha.$$

where the last equality follows from feasibility of w for $\mathrm{LB}(\cdot)$. Then the regret incured by A is $(X_t - \frac{1}{2})(x_t - x_t^\ddagger)$ with expectation $\mathbb{E}_t[(x_t - x_t^\ddagger)]$. But by Le Cam's method, since x_t is only a function of $Y_{<t}$,

$$\begin{aligned}\mathbb{E}[(x_t - x_t^\ddagger)] &\geq (1/2)\mathbb{E}[|x_t - x_t^\ddagger|] \geq (1/2)\mathbb{E}[\min(|x_t|, |1 - x_t|)] \\ &\geq (1/2)\left(\mathbb{E}[|x_t|] - \mathbb{E}[|x_t|] + 1\right) \\ &\geq (1/2)\left(1 - \mathrm{TV}\left(A_{S_t \mid x}, A_{S_t \mid 1-x}\right)\right) \geq (1/2)\left(1 - \sqrt{\mathrm{KL}(\cdots)}\right).\end{aligned}$$

Therefore, picking $\varepsilon = 1/4$, we get $\mathbb{E}[(x_t - x_t^\ddagger)] \geq 1/4$, which implies the expected regret at round t is at least $\mathbb{E}_t[(x_t - x_t^\ddagger)] \geq 1/4$. Summing over t and using Lemma 78, we get

$$\mathrm{Reg}(A) \geq \frac{\sum_{t=1}^{T} \sqrt{\sum_{o} P_{S_t \mid x}(o)\, w_t^2}}{4\sqrt{2}} \geq \frac{\sum_{t=1}^{T} \sqrt{\sum_{o} P_{S_t \mid x}(o)\, w_t^2}}{16\sqrt{2}} \geq \mathrm{LB}(w)/50.$$

□

Appendix F

Sampling Polytopes with Riemannian HMC: Faster Mixing via The Lewis Weights Barrier

Structure of the appendices. The appendices are organized as follows:

- In Appendix F.1 we discuss the basic tools and notation that we use throughout the paper.
- In Appendix F.2, we prove that the hybrid barrier is third-order ℓ_a-self-concordant. This section relies on higher-order derivative estimates. We give an overview of these in Section F.2.2 and defer their proofs to Section F.6.
- In Appendix F.3, we prove one-step coupling and bound the mixing time by combining the Hamiltonian smoothness bounds for our hybrid barrier, the isoperimetry of the stationary distribution with respect to the chosen metric, and the stability of the Hamiltonian curves.
- In Appendix F.4, we develop our abstract framework on relating the third-order ℓ_a-self-concordance to control the smoothness of the Hamiltonian fields on the manifold.
- In Appendix F.5, we prove the stability of the smoothness properties of the Hamiltonian curves as we start varying the initial location and velocity of the curve.
- In Appendix F.6, we prove the higher-order derivative bounds for Lewis weights and related objects that we need for Section F.2.
- In Appendix G.7.4, we prove an isoperimetric inequality on the Riemannian manifold M equipped with metric g, the Hessian of our hybrid barrier.

F.1 Preliminaries

F.1.1 Notation

We denote the target probability distribution inside the polytope by $\hat{\pi} \propto e^{-}$. Recall that for the LP polytope description $Ax \geq b$, we define the rescaling A by the slack variables, namely

$$A_x := \operatorname{diag}(a_i^\top x - b_i)^{-1} A.$$

For a vector v in the tangent space of x, we also work with the reparameterization of v defined as

$$\begin{aligned} s_{v;x} &:= A_x v; \\ S_{x;v} &:= \operatorname{diag}(s_{x;v}). \end{aligned} \tag{F.1}$$

which is the speed that v approaches the facets of the polytope normalized by the slacks. In our derivations we treat hadamard product of matrices with higher priority than matrix multiplications, namely $AB \circ C$ means $A(B \circ C)$. We refer to the p-Lewis weights vector of A_x by w_x and its diagonal matrix version by $W_x := \operatorname{diag}(w_x)$.

Define the log barrier by ϕ:

$$\phi(x) := \sum_{i=1}^{m} \log(a_i^\top x - b_i).$$

We define the metric g_2 as the Hessian of the $n-m$-rescaled log barrier, $g_2(x) = \frac{n}{m}\nabla^2 \phi(x)$. It is easy to check that g_2-norm of v is given by the ℓ_2 norm of $s_{x,v}$:

$$\|v\|_{g_2}^2 \quad v^\top g_2(x) v \quad v^\top A_x^\top A_x v \quad \|s_{x,v}\|_2^2.$$

For a given point x inside polytope P, we define the symmetrized polytope $P - x + P$ around x as the following: we reflect P around x and intersect it with the P namely $P - x + P$, as illustrated in Figure F.1. The approximation of the symmetrized body by the ellipsoids corresponding to the Hessian of the barrier function plays a key role in bounding the isoperimetry constant, as we describe in Section G.7.4.

Throughout the proof, we use the notation $\lesssim$ to indicate an inequality that is true up to logarithmic factors. We use ∇ for Euclidean derivative and ∇_x and ∇_t for covariant differentiation with respect to the metric structure on the manifold. Moreover, we use $\preceq$ to show Löwner inequalities up to universal constants. We use $\|\cdot\|$ with various subindices to refer to different vector norms, and $\|\cdot\|_a$ and $\|\cdot\|_a$ to refer to the infinity to infinity operator norm and the usual operator norm of a matrix, respectively. Throughout the paper, by high probability we mean with probability $1 - 1/\mathrm{poly}(m)$.

F.1.2 John Ellipsoid and Lewis Weights

Proving good isoperimetry for a specific barrier can be reduced to how well the ellipsoids corresponding to the Hessian of the barrier at each point x inside the polytope approximate the symmetrized polytope around x. A natural way to approximate a symmetric polytope is via its John Ellipsoid, i.e. the ellipsoid of maximum volume contained in the polytope. Parametrizing the John ellipsoid as $A_x^\top W A_x$ for a positive diagonal matrix W, i.e., a weighted sum of the outer product of the rows of A_x, the weights are characterized by the following optimization problem:

$$\max_{w \in \mathbb{R}^n_{\geq 0}} \log\det(A_x^\top W A_x) \qquad \text{(F.2)}$$
$$\text{s.t. } 1^\top w \quad n.$$

where $W \quad \mathrm{diag}(w)$ is the diagonal matrix corresponding to the vector w. The John ellipsoid approximates the symmetrized polytope in the sense that (1) it is inside the ellipsoid and (2) scaling it up by $\sqrt{n}$ will make it contains the symmetrized polytope.

On the other hand, in order to prove smoothness of the HMC curves, we need to pick a barrier whose Hessian does not change too fast as a function of x. Unfortunately the John ellipsoid is not stable. In particular, the weights w which maximize equation F.2 are not even continuous with respect to x. An alternative is to use the p-Lewis weights to define the ellipsoid, obtained as the solution to a relaxation of the program in equation F.2:

$$w_x < \mathrm{argmax}_{w \in \mathbb{R}^n_{\geq 0}} \quad \log\det(A_x^\top W^{1-2/p} A_x) \quad 1 \quad 2/p \, 1^\top w; \qquad \text{(F.3)}$$

where $W \quad \mathrm{diag}(w)$. Moreover, the optimal value of the program in equation F.3 is denoted by the Lewis weight barrier at x as defined next.

De nition 30 (Lewis weight barrier). The Lewis weight barrier can be de ned as the solution of the following optimization problem:

$$_p(x) < \max_{w>R^n_{A0}} \log\det(A_x W^{1\ 2/p} A_x) \quad 1 \quad 2/p1w; \tag{F.4}$$

Let g_1 $©^2$ $_p$ be the metric de ned by the Hessian of the Lewis weight barrier which constitute the rst part of our hybrid barrier . Hence, the metric with respect to our hybrid barrier can be written as g $_0(g_1$ $g_2)$.

$$\begin{aligned} g_1 &< ©^2 \log\det(A_x W^{\frac{1}{x}\ 2/p} A_x); \\ g_2 &< \frac{n}{m} A_x A_x: \\ g &< \ _0(g_1 \quad g_2): \end{aligned} \tag{F.5}$$

Although $_p$ is de ned as the volume of the ellipsoid when the each a_i is reweighted by 1 $2/p$ power of the p-Lewis weights, it is not clear if the Hessian of this barrier can be estimated explicitly by Lewis weights. It turns out that this is the case, the ellipsoid corresponding to g_1 is roughly the same as the one de ned by $A_x W_x A_x$ (Lemma 31 in [2]).

Figure F.1: The unit ball of the local norm $Y_{x;a}$ is the symmetrized polytope around $x > P$.

Lemma 81 (Restatement of Lemma 31 in [2]) For the Lewis weight barrier $_p$ we can bound the local norm of its Hessian as

$$\sum_{i\ 1}^{m} w_i(x)(s_{x;v})_i^2 \ B_v g_1(x)v \ B1 \quad p \sum_{i\ 1}^{m} w_i(x)(s_{x;v})_i^2: \tag{F.6}$$

Equivalently

$$A_x^\top W_x A_x \preceq g_1(x) \preceq (1+p)A_x^\top W_x A_x. \tag{F.7}$$

Moreover, we have the following formula for$g_1(x)$,

Lemma 82 (Equation (5.5) in [2]). The Lewis weight metric $g_1(x) = \log\det(A_x^\top W_x^{1-2/p} A_x)$ can be written in the following form

$$g_1(x) = A_x^\top W_x - 2\Pi_x A_x + (1-2/p)A_x^\top \Pi_x G_x^{-1}\Pi_x A_x, \tag{F.8}$$

where we define

$$\Pi_x = W_x - P_x^{(2)},$$
$$G_x = W_x + (1-2/p)\Pi_x. \tag{F.9}$$

In the above Lemma, Π_x, G_x, $r_{x,v}$, and $R_{x,v}$ are all functions of the location variable$x$. A useful fact about Π_x and G_x is that they can be estimated byW_x. It is easy to see that $-W_x \preceq \Pi_x \preceq W_x$ and $\frac{1}{p}W_x \preceq G_x \preceq W_x$ (see Lemma 154 for a proof). This enables us to estimate$g_1(x)$ by the simpler form $A_x^\top W_x A_x$. On the other hand, it is clear from Equation equation F.8 that in order to estimate the first derivative of g_1 in direrction v, we need to study the derivative $DW_x(v)$. In Lemma 83 we illustrate the form of the Jacobian of the Lewis weights as a function of$x$, by taking its directional derivative in direction v based on fundamental matrices Π_x, G_x, for any point x inside the polytope. Before that, we start by defining the projection matrix P_x with respect to A_x when reweighted by$W_x^{1/2-1/p}$.

Definition 31 (Projection matrix). we define the projection matrixP_x, implicitly depending on x, as

$$P_x = P(W_x^{1/2-1/p}A_x) = W_x^{1/2-1/p}A_x(A_x^\top W_x^{1-2/p}A_x)^{-1}A_x^\top W_x^{1/2-1/p},$$

whereW_x is the p-Lewis weights calculated at$x$. Moreover, we denote the Hadamard square $P_x^{\circ 2}$ of the projection matrix by$P_x^{(2)}$:

$$(P_x^{(2)})_{ij} = (P_x^{\circ 2})_{ij} = (P_x)_{ij}^2.$$

Next, we state a formula for the derivative of the Lewis weights.

Lemma 83 (Derivative of the Lewis weights) For arbitrary direction $v \perp M$, the directional derivative $DW_x(v)$ can be calculated as

$$DW_x(v) = 2\mathrm{diag}(\Pi_x G_x^{-1} W_x s_{x,v}).$$

Due to the importance and repetition of the vector $\Pi_x G_x^{-1} W_x s_v$ in our calculations later on, we give it a separate notation

$$r_{x,v} = G_x^{-1} W_x s_{x,v},$$
$$R_{x,v} = \mathrm{diag}(r_{x,v}). \tag{F.10}$$

Then, the derivative ofW_x can be written as

$$DW_x(v) = 2\mathrm{diag}(\Pi_x r_{x,v}).$$

Furthermore, when$v$ is clear from the context, we denote$DW_x(v)$ in short by $W'_{x,v}$.

Figure F.2: The Lewis weight barrier with $p = \log(n)$, with $p = 4$, and our hybrid barrier which is regularized with the log barrier.

F.1.3 Markov Chains

For a Markov chain with state space M, stationary distribution Q and next step distribution $p_u(\cdot)$ for any $u \in M$, the conductance of the Markov chain is defined as

$$\Phi = \inf_{S \subset M} \frac{\int_S p_u(M \setminus S)\, dQ(u)}{\min(Q(S), Q(M \setminus S))}.$$

The conductance of an ergodic Markov chain allows us to bound its mixing time, i.e., the rate of convergence to its stationary distribution, e.g., via the following theorem of Lovász and Simonovits. However, we will need a more refined notion of conductance here, to be able to ignore small subsets of small measure in bounding the conductance.

Definition 32 (s-conductance) Consider a Markov chain with a state space M, a transition distribution T_x and stationary distribution π. For any $s \in (0, 1/2]$, the s-conductance of the Markov chain is defined by

$$\Phi_s = \inf_{S : \pi(S) \in [s, 1/2]} \frac{\int_S T_x(S^c)\, d\pi(x)}{\min(\pi(S) - s, \pi(S^c) - s)}.$$

A lower bound on the s-conductance of a Markov chain leads to an upper bound on its mixing rate.

Lemma 84. [110] Let π_t be the distribution of the points obtained after t steps of a lazy reversible Markov chain with the stationary distribution π. For $0 < s < 1/2$ and $H_s = \sup(S : \pi(S) \le s)$, it follows that

$$d_{TV}(\pi_t, \pi) \le H_s - \frac{H_s}{s}\left(1 - \frac{\Phi_s^2}{2}\right)^t.$$