Efficient and Robust Black-box Integral-approximation and Optimization

Yueming Lyu
Faculty of Engineering and Information Technology
University of Technology Sydney

A thesis submitted for the degree of
Doctor of Philosophy

September 2021

Certificate of Original Authorship

I hereby declare that this thesis, is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise reference or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis. The contents of this dissertation are original and have not been submitted in whole or in part for qualifications at any other academic institution. This research is supported by the Australian Government Research Training Program.

Signature: Production Note: Signature removed prior to publication.

Date: 01-October-2021

I would like to dedicate this thesis to my loving parents, who support me all the time.

Acknowledgements

First, I would like to express my deepest gratitude to my supervisor, Prof. Ivor W. Tsang, who gave me the research opportunity in his group. My research journey was not easy. I was depressed and suffered a lot. It was Prof. Ivor W. Tsang who provided me an opportunity to continue research on machine learning. Prof. Ivor W. Tsang always supported me to be patient to do in-depth analysis instead of shallow work. His insightful guidance helped me to focus on the key reasons to solve the problems. Prof. Ivor W. Tsang encouraged me to explore the research area that I am interested. He guided me to know how to think in both theoretical and empirical views of research. Prof. Ivor W. Tsang taught me how to divide and conquer problems and hierarchically solve them. His critical thinking and broad view gave me insightful guidance and let me know how a good researcher should be. Prof. Ivor W. Tsang's advice and support improved my skills and confidence to be an independent researcher. It is a great fortune for me to be a Ph.D. student under his supervision.

Second, I would like to thank many friends who have supported me. I want to thank Dr. Jiangchao Yao and Dr. Yuangang Pan for discussing both research and life. It was a memorable time for waiting for the subway and discussing research together. I appreciate the time with them and Xiaowei Zhou, Xingrui Yu, Xu Chen, Yan Zhang, Jin Li, Yaxin shi, Jinliang Deng, Yinghua Yao for discussion and lunch together. I am so grateful for such a relaxed life and beautiful experience in Sydney. I would also like to thank Dr. Yuan Yuan, Dr. Yanbin Liu, Dr. Xin Yu, Dr. Xiaolin Zhang, Fan Ma, Tianqi Tang, Guangrui Li, and Guang Li for their support and discussion. I would thank Dr. Peng Sun and Dr. Li Shen for the discussion and for working together. I may miss mentioning many others, but I thank them for their help.

Finally, I would like to dedicate this thesis to my dearest parents and family. Thank you so much for being there, supporting me all the time.

Abstract

Black-box optimization and black-box integral approximation are important techniques for machine learning, industrial design, and simulation in science. This thesis investigates black-box integral approximation and black-box optimization by considering the closed relationship between them. For integral approximation, we develop a simple closed-form rank-1 lattice construction method based on group theory. Our method reduces the number of distinct pairwise distance values to generate a more regular lattice. Furthermore, we investigate structured points set for integral approximation on hyper-sphere. Our structured point sets can serve as a good initialization for black-box optimization. Moreover, we propose stochastic black-box optimization with implicit natural gradients for black-box optimization. Our method is very simple and has only the step-size hyper-parameter. Furthermore, we develop a batch Bayesian optimization algorithm from the perspective of frequentist kernel methods, which is powerful for low-dimensional black-box optimization problems. We further apply our structured integral approximation techniques for kernel approximation. In addition, we develop structured approximation for robust deep neural network architecture, which results in an elegant and simple architecture that preserves optimization properties. Moreover, we develop adaptive loss as a tighter upper bound approximation for expected 0-1 risk, robust and trainable with SGD.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Black-box optimization

1.1.1 Sampling based Optimization

Given a proper function $f(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$ such that $f(\boldsymbol{x}) > -\infty$, we aim at minimizing $f(\boldsymbol{x})$ by using function queries only, which is known as black-box optimization. It has a wide range of applications, such as automatic hyper-parameters tuning in machine learning and computer vision problems [156], adjusting parameters for robot control and reinforcement learning [20, 36, 115], black-box architecture search in engineering design [168] and drug discovery [128].

Several kinds of approaches have been widely studied for black-box optimization, including Bayesian optimization (BO) methods [32, 118, 160], evolution strategies (ES) [17, 69] and genetic algorithms (GA) [158]. Among them, Bayesian optimization methods are good at dealing with low-dimensional expensive black-box optimization, while ES methods are better for relatively high-dimensional problems with cheaper evaluations compared with BO methods. ES-type algorithms can well support parallel evaluation, and have drawn more and more attention because of its success in reinforcement learning problems [38, 114, 149], recently.

CMA-ES [2, 69] is one of state-of-the-art ES methods with many successful applications. It uses second-order information to search candidate solutions by updating the mean and covariance matrix of the likelihood of candidate distributions. Despite its successful performance, the update rule combines several sophisticated components, which is not well understood. Wierstra et al. show that directly applying standard reinforce gradient descent is very sensitive to variance in high precision search for black-box optimization [173]. Thus, they propose Natural evolution strategies (NES) [173] to estimate the natural gradient for black-box optimization. However, they use the

Monte Carlo sampling to approximate the Fisher information matrix (FIM), which incurs additional error and computation cost unavoidably. Along this line, [3] show the connection between the rank-μ update of CMA-ES and NES [173]. [135] further show that several ES methods can be included in an unified framework. Despite these theoretical attempts, the practical performance of these methods is still inferior to CMA-ES. Moreover, these works do not provide any convergence rate analysis, which is the key insight to expedite black-box optimizations.

Another line of research for ES-type algorithms is to reduce the variance of gradient estimators. Choromanski et al. [38] proposed to employ Quasi Monte Carlo (QMC) sampling to achieve more accurate gradient estimates. Recently, they further proposed to construct gradient estimators based on active subspace techniques [37]. Although these works can reduce sample complexity, how does the variance of these estimators influence the convergence rate remains unclear.

To take advantage of second-order information for the acceleration of black-box optimizations, we propose a novel theoretical framework: stochastic Implicit Natural Gradient Optimization (INGO) algorithms, from the perspective of information geometry. Raskutti et al. [143] give a method to compute the Fisher information matrix implicitly using exact gradients, which is impossible for black-box optimization; while our methods and analysis focus on black-box optimization. To the best of our knowledge, we are the first to design stochastic implicit natural gradient algorithms for black-box optimization. Our methods take a stochastic black-box estimate instead of the exact gradient to update. Theoretically, this update is equivalent to a stochastic natural gradient step w.r.t. natural parameters of an exponential-family distribution. We present our INGO method in Chapter 3. Our contributions are summarized as follows:

- To the best of our knowledge, we are the first to design stochastic implicit natural gradient algorithms w.r.t natural parameters for black-box optimization. We propose efficient algorithms for both continuous and discrete black-box optimization. Our methods construct stochastic black-box update without computing the FIM. Our method can adaptively control the stochastic update by taking advantage of the second-order information, which is able to accelerate convergence and is primarily important for ill-conditioned problems. Moreover, our methods have fewer hyperparameters and are much simpler than CMA-ES.

- Theoretically, we prove the convergence rate of our continuous optimization methods for convex functions. We also show that reducing variance of the

black-box gradient estimators by orthogonal sampling can lead to a small regret bound.

- Empirically, our continuous optimization method achieves a competitive performances compared with the state-of-the-art method CMA-ES on benchmark problems. We find that our method with full matrix update can obtain higher optimization precision compared with IGO [135] on some challenging problems. We further show the effectiveness of our methods on RL control problems. Moreover, our discrete optimization algorithm outperforms a GA method on a benchmark problem.

1.1.2 Bayesian Optimization

To achieve greater efficiency in the batch selection, we propose to simultaneously select candidate queries of a batch in a holistic manner, rather than the previous sequential manner. In this paper, we analyze both the batch BO and the sequential BO from a frequentist perspective. For the batch BO, we propose a novel batch selection method that takes both the mean prediction value and the correlation of points in a batch into consideration. Our method leads to a novel batch acquisition function. By jointly maximizing the novel acquisition function with respect to all the points in a batch, the proposed method is able to attain a better exploitation/exploration trade-off.

Bayesian Optimization (BO) is another promising approach to address expensive black-box (non-convex) optimization problems. Applications of BO include automatic tuning of hyper-parameters in machine learning [156], gait optimization in robot control [115], molecular compounds identifying in drug discovery [128], and optimization of computation-intensive engineering design [168].

BO aims to find the optimum of an unknown, usually non-convex function f. Since little information is known about the underlying function f, BO requires to estimate a surrogate function to model the unknown function. Therefore, one major challenge of BO is to seek a balance between collecting information to model the function f (exploration) and searching for an optimum based on the collected information (exploitation).

For the sequential BO, we obtain a similar acquisition function as that in the GP-UCB [160], except that our function employs a constant weight for the deviation term. The constant weight is preferred over the previous theoretical weight proposed in GP-UCB, because the latter is overly conservative, which has been observed in many other works [25, 26, 160]. Moreover, for functions with a bounded norm in the

reproducing kernel Hilbert space (RKHS), we derive the non-trivial regret bounds for both the batch BO method and the sequential BO method.

At the beginning of the BO process, since little information is known, the initialization phase becomes vitally important. To obtain a good and robust initialization, we first study the properties which are necessary for a robust initialization through analyzing the adversarial regret. We prove that the regret bounds decrease with the decrease of the covering radius (named fill distance in [87]). Minimizing the covering radius of a lattice is equivalent to maximizing its packing radius (named separate distance in [87]) [42,89], we then propose a novel fast searching method to maximize the packing radius of a rank-1 lattice and obtain the points set with a small covering radius. All details are presented in Chapter 4. Our contributions are summarized as follows:

- We study the black-box optimization for functions with a bounded norm in RKHS and achieve deterministic regret bounds for both the noise-free setting and the perturbation setting. The study not only brings a new insight into the BO literature but also provides better guidance for designing new acquisition functions.

- We propose a more-efficient novel adaptive algorithm for batch optimization, which selects candidate queries of a batch in a holistic manner. Theoretically, we prove that the proposed methods achieve non-trivial regret bounds.

- We analyze the adversarial regret for a robust initialization of BO, and theoretically prove that the regret bounds decrease with the decrease of the covering radius, and provide a criterion for generating points set to minimize the bound for the initialization of BO.

- We propose a novel, fast searching algorithm to maximize the packing radius of a rank-1 lattice and generate a set of points with a small covering radius. The generated points set provides a robust initialization for BO. Moreover, the set of points can be used for integral approximation on domain $[0, 1]^d$. Experimental results show that the proposed method can achieve a larger packing radius (separate distance) compared with the baselines.

1.2 Black-box Integral Approximation

Black-box Integral-approximation is closely related to black-box optimization. For example, the point set for integral approximation can also be used for black-box optimization. Integral operation is critical in a large amount of interesting machine learning applications, e.g. kernel approximation with random feature maps [140], variational inference in Bayesian learning [22], generative modeling and variational autoencoders [94]. Directly calculating an integral is usually infeasible in these real applications. Instead, researchers usually try to find an approximation for the integral. A simple and conventional approximation is Monte Carlo (MC) sampling, in which the integral is approximated by calculating the average of the i.i.d. sampled integrand values. Monte Carlo (MC) methods [65] are widely studied with many techniques to reduce the approximation error, which includes importance sampling and variance reduction techniques and more [10].

To further reduce the approximation error, Quasi-Monte Carlo (QMC) methods utilize a low discrepancy point set instead of the i.i.d. sampled point set used in the standard Monte Carlo method. There are two main research lines in the area of QMC [46,130], i.e., the digital nets/sequences and lattice rules. The Halton sequence and the Sobol sequence are the widely used representatives of digital sequences [46]. Compared with digital nets/sequences, the points set of lattice rules preserve the properties of lattice. The points partition the space into small repeating cells. Among previous research on the lattice rules, Korobov introduced integration lattice rules in [95] for an integral approximation of the periodic integrands. [155] proves that there also exist good lattice rules for non-periodic integrands. According to general lattice rules, a point set is usually constructed by enumerating the integer vectors and multiplying them with an invertible generator matrix. A general lattice rule has to check each constructed point to see whether it is inside a unit cube and discard it if it is not. The process is repeated until we reach the desired number of points. This construction process is inefficient since the checking step is required for every point. Note that rescaling the unchecked points by the maximum norm of all the points may lead to non-uniform points set in the cube.

An interesting special case of the lattice rules is the rank-1 lattice, which only requires one generating vector to construct the whole point set. Given the generating vector, rank-1 lattices can be obtained by a very simple construction form. It is thus much more efficient to construct the point set with the simple construction form. Compared with the general lattice rule, the construction form of the rank-1 lattice

has already guaranteed the constructed point to be inside the unit cube, therefore, no further checks are required. We refer to [46] and [130] for a more detailed survey of QMC and rank-1 lattice.

To obtain a simple and fast QMC lattice, we propose a closed-form rank-1 lattice rule that directly computes a generating vector without any search process. To generate a more evenly spaced lattice, we propose to reduce the number of distinct pairwise distance in the lattice point set to make the lattice more regular w.r.t. the minimum toroidal distance [64]. Larger minimum toroidal distance means more regular. Based on group theory, we derive that if the generating vector $\boldsymbol{z}$ satisfies the condition that set $\{\boldsymbol{z}, -\boldsymbol{z}\} := \{z_1, \cdots, z_d, -z_1, \cdots, -z_d\}$ is a subgroup of the multiplicative group of integers modulo n, where n is the number of points, then the number of distinct pairwise distance can be efficiently reduced. We construct the generating vector by ensuring this condition. We presented all details of our subgroup-based rank-1 lattice in Chapter 5. Our contributions are summarized as follows:

- We propose a simple and efficient closed-form method for rank-1 lattice construction, which does not require the time-consuming exhaustive computer search that previous rank-1 lattice algorithms rely on. A side product is a closed-form method to generate QMC points set on sphere $\mathbb{S}^{d-1}$ with bounded mutual coherence, which is presented in Appendix.

- We generate a more regular lattice by reducing the number of distinct pairwise distances. We prove a lower and an upper bound for the minimum l_1-norm-based and l_2-norm-based toroidal distance of the rank-1 lattice. Theoretically, our constructed lattice is the optimal rank-1 lattice for maximizing the minimum toroidal distance when the number of points n is a prime number and $n = 2d+1$.

- Empirically, the proposed method generates near-optimal rank-1 lattice compared with the Korobov search method in maximizing the minimum of the l_1-norm-based and l_2-norm-based toroidal distance.

- Our method obtains better approximation accuracy on benchmark test problems and kernel approximation problem.

1.3 Connection Between Black-box Integral Approximation and Black-box Optimization

Black-box integral approximation and black-box optimization have a close relationship between each other. Specifically, an integral over a bounded domain $\mathcal{X}$

$$I(f) := \int_{\mathcal{X}} f(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}, \tag{1.1}$$

can be viewed as the mean value of infinite function values over the bounded domain $\mathcal{X}$. A quasi-Monte Carlo approximation (equal-weight approximation) is given as follows:

$$\widehat{I}(f) := \frac{1}{n} \sum_{i=1}^{n} f(\boldsymbol{x}_i). \tag{1.2}$$

It can be viewed as an approximation using the mean value over the function value of n samples. The objective of black-box integral approximation is to find n samples over domain $\mathcal{X}$, such that the mean estimator is a good approximation of the integral.

Similarly, an optimization over a bounded domain $\mathcal{X}$, i.e.,

$$M(f) := \min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}), \tag{1.3}$$

can be viewed as taking the minimum over infinite function values over the bounded domain $\mathcal{X}$.

A black-box optimization with n samples is given as follows:

$$\widehat{M}(f) := \min_{i \in \{1, \cdots, n\}} f(\boldsymbol{x}_i). \tag{1.4}$$

It can be viewed as an approximation using the min value over the function value of n samples. The objective of black-box optimization is to find n samples over domain $\mathcal{X}$, such that the min estimator is a good approximation of the true minimum over the entire domain $\mathcal{X}$.

Thus, the key of both black-box integral-approximation and black-box optimization is to construct n samples for a better approximation.Techniques for black-box integral approximation can be used for black-box optimization and vice-visa. For example, a good point set for integral-approximation can also be used as an initialization for black-box optimization.

1.4 Integral Approximation for Kernel Methods and Deep Learning

1.4.1 Integral Approximation for Kernel Approximation

Kernel methods such as Gaussian processes (GPs) [144, 157, 159] and support vector machines (SVMs) [33, 56] have been successfully used in many statistical modeling and machine learning tasks. Despite of strong expressive power, kernel methods usually cannot scale up to the large scale datasets due to the cubic time complexity. Integral approximation techniques can be used to approximate the kernel functions. Specifically, a shift invariant kernel $\mathrm{K}(\mathbf{x}, \mathbf{z})$ can be rewritten an integral

$$\mathrm{K}(\mathbf{x}, \mathbf{z}) = \int_{\mathbb{R}^{\mathbf{d}}} \mathbf{e}^{-\mathbf{i}(\mathbf{x}-\mathbf{z})^{\mathbf{T}}\mathbf{w}} \mathbf{dp}(\mathbf{w}).$$

In addition, a b-th order arc-cosine kernel can be reformulated as an integral

$$\mathrm{K}_b(\mathbf{x}, \mathbf{z}) = 2 \int_{\mathbb{R}^d} s(\mathbf{w}^T\mathbf{x}) s(\mathbf{w}^T\mathbf{z}) (\mathbf{w}^T\mathbf{x})^b (\mathbf{w}^T\mathbf{z})^b p(\mathbf{w}) d\mathbf{w}.$$

The idea of kernel approximation is applying QMC or MC techniques to approximate the integral. Namely, the kernel can be approximated as $\mathrm{K}(x, z) \approx \Psi(\mathbf{x})^\top \Psi(\mathbf{z})$, where Ψ is the explicit mapped feature constructed as $\Psi(\mathbf{x}) = f(\mathbf{W}^T\mathbf{x})/\sqrt{N}$, where $f(\cdot)$ denotes the nonlinear function, $\mathbf{W} \in \mathbb{R}^{d\times N}$ is constructed by N i.i.d samples drawn from a distribution determined by the kernel.

The training and inference of kernel methods can be greatly accelerated by working directly on the primal space of $\Psi(\cdot)$. For example, Gaussian Processes (GPs) have $O(L^3)$ computation and $O(L^2)$ storage complexity. By using feature maps, it reduces to $O(N^2L + N^3)$ computation and $O(NL + N^2)$ storage complexity. All these elegant properties make random feature maps promising for large scale kernel methods. Thus, many kernel methods [41, 106, 134] have been proposed to deal with large scale statistical learning by directly working on feature maps.

Generally, two aspects of random feature maps are mostly concerned by literature for scaling up kernel methods. One is the approximation accuracy of feature maps while the other is the computational cost of feature maps construction. To achieve better approximation accuracy, [14, 178] employ QMC [47] sampling instead of standard Monte Carlo sampling to construct feature maps. By mapping QMC points on $[0, 1]^d$ through the inverse cumulative distribution function, they construct more effective feature maps. To reduce time complexity, [105] propose Fastfood to construct feature maps. Benefiting from the special structured matrix multiplication, it reduces

time complexity of feature maps construction from $O(Nd)$ to $O(N \log d)$. However, it achieves computational efficiency at the expense of increasing the variance of approximation. Recently, [57] employ the property of circulant matrix to accelerate feature maps construction of Gaussian kernel without increasing the variance. [39] generalize the Fastfood and circulant feature maps to P model and particularly discuss the structured matrix with low-displacement rank. Despite of the success of P model, it still cannot achieve better approximation accuracy compared with feature maps obtained with fully Gaussian matrix.

To achieve better approximation accuracy and loglinear time complexity, we propose Spherical Structured Feature (SSF) maps to approximate shift and rotation invariant kernels as well as b^{th}-order arc-cosine kernels [34]. Specifically, We construct SSF maps based on the point set on $d-1$ dimensional sphere $\mathbb{S}^{d-1}$, where the points are columns of a particular structured matrix produced by a discrete Fourier matrix. The points on $\mathbb{S}^{d-1}$ for SSF maps construction can be generated by optimizing the discrete Riesz s-energy. According to [28], optimizing the discrete Riesz s-energy (for s in some ranges) can generate QMC designs on $\mathbb{S}^{d-1}$, which usually can achieve smaller approximation error compared with fully random methods. Moreover, because of special structure of the point set, SSF maps construction can achieve loglinear time complexity via Fast Fourier Transform (FFT).

We apply our spherical structured approximation techniques for kernel approximation in Chapter 6. Our contributions are summarized as follows:

- We present Spherical Structured Feature (SSF) maps to approximate shift and rotation invariant kernels as well as b^{th}-order arc-cosine kernels [34]. We prove that the inner product of SSF maps are unbiased estimates for above kernels if asymptotically uniformly distributed point set on $d-1$ dimensional sphere $\mathbb{S}^{d-1}$ is given.

- We propose an efficient coordinate decent method to find a local optimum of the discrete Riesz s-energy [29], thereby approximately generating asymptotically uniformly distributed points on $\mathbb{S}^{d-1}$.

- We can construct SSF maps with linear space complexity and loglinear time complexity. Empirically, SSF maps achieve superior performance compared with other methods.

1.4.2 Structured Integral Approximation for Robust Neural Network Architecture

Deep neural networks (DNNs) have obtained great success in many applications, including computer vision [71], natural language processing [175] (NLP), and reinforcement learning [124] etc. However, the theory of deep learning is much less explored compared with its great empirical success. A key challenge of deep learning theory is that deep neural networks are heavily overparameterized. Namely, the number of parameters is much larger than training samples. In practice, as the depth and width increasing, the performance of deep NN also becomes better [161,184], which is far beyond the traditional learning theory regime.

In the traditional neural networks and kernel methods literature, it is well known the connection between the infinite width neural networks and Gaussian process [75], and the universal approximation power of NN [112]. However, these theories cannot explain why the success of deep neural networks. A recent work, Neural Tangent Kernel [80] (NTK), shows the connection between training an infinite-width NN and performing functional gradient descent in a Reproducing Kernel Hilbert Space(RKHS) associated with the NTK. Because of the convexity of the functional optimization problem, Jacot et al. show the global convergence for infinite-width NN under the NTK regime. Along this direction, Hanin et al. [68] analyze the NTK with finite width and depth. Shankar et al. [153] empirically investigate the performance of some simple compositional kernels, NTKs, and deep neural networks. Nitanda et al. [132] further show the minimax optimal convergence rate of average stochastic gradient descent in a two-layer NTK regime.

Despite the success of NTK [80] on showing the global convergence of NN, its expressive power is limited. Zhu et al. [4] provide an example that shallow kernel methods (including NTK) need a much larger number of training samples to achieve the same small population risk compared with a three-layer ResNet. They further point out the importance of hierarchical learning in deep neural networks [5]. In [5], they give the theoretical analysis of learning a target network family with square activation function under deep NN regime. Besides, there are quite a few works focus on the analysis of two-layer networks [18,27,52,88,113,180] and shallow kernel methods without hierarchical learning [9,44,58,109,190].

Although some particular examples show deep models have more powerful expressive power than shallow ones [4,5,54], how and why deep neural networks benefit from the depth remain unclear. Zhu et al. [5] highlight the importance of a backward

feature correction. In this thesis, we investigate the deep neural networks from a different kernel method perspective. All details are presented in Chapter 7.

Our contributions are summarized as follows:

- We first propose a novel Neural Optimization Kernel (NOK) family, which enables kernel methods taking advantage of ResNet-type architecture.

- Theoretically, we show that the architecture of NOK performs optimization of regularized problems. We prove the monotonic descent property for a wide range of both convex and non-convex regularized problems. Moreover, we prove a $O(1/T)$ convergence rate for convex regularized problems. Namely, our NOK family performs an optimization through model architecture. A T-layer model performs T-step monotonic descent updates.

- We propose a novel data-dependent structured approximation method, which establishes the connection between training deep neural networks and kernel methods associated with NOKs. The resultant computation graph is a ResNet-type finite width NN. The activation function of NN specifies the regularization problem explicitly or implicitly. Our structured approximation preserved the monotonic descent property and $O(1/T)$ convergence rate. Furthermore, we propose both supervised and unsupervised learning schemes. For the unsupervised learning case and shared parameter case, as the width tends to infinity, training finite structured NN with GD tends to functional gradient descent in an RKHS associated with a NOK. Notably, for strongly convex regularized regression problems, functional gradient descent leads to the global convergence. For supervised learning cases with free parameters, training NN learns a kernel balanced between the supervised signal and the prior regularization. We prove the Rademacher complexity bound and generalization bound.

- Empirically, we show that our unsupervised data-dependent structured approximation block can serve as a simple plug-in of popular backbones for robust deep learning. Extensive experiments on CIFAR10 and CIFAR100 with ResNet and DenseNet backbones show the robustness of our structured approximated NOK against the Gaussian noise, Laplace noise, and FGSM adversarial attack [62].

1.4.3 Adaptive Loss As A Tighter Upper Bound Approximation of Expected 0-1 Risk

The empirical risk is a Monte Carlo integral approximation of the expected risk. When 0-1 loss is used for training, it is more robust to outliers compared with an unbounded (convex) loss (e.g. hinge loss) [122]. This is due to unbounded convex losses putting much weight on the outliers (with a large loss value) when minimizing the losses [122]. If the unbounded (convex) loss is employed in deep network models, this becomes more prominent. Since training loss of deep networks can often be minimized to zero, outlier with a large loss has a large impact on the model. On the other hand, the 0-1 loss treats each training sample equally. Thus, each sample does not have too much influence on the model. Therefore, the model is tolerant of a small number of outliers.

Although the 0-1 loss has many robust properties, its non-differentiability and zero gradients make it difficult to optimize. One possible way to alleviate this problem is to seek an upper bound of the 0-1 loss that is still efficient to optimize but tighter than conventional (convex) losses. Such a tighter upper bound of the 0-1 loss can reduce the influence of the noisy outliers compared with conventional (convex) losses. At the same time, it is easier to optimize compared with the 0-1 loss. When minimizing the upper bound surrogate, we expect that the 0-1 loss objective is also minimized.

To efficiently minimize the 0-1 loss while keeping the robust properties, we propose a novel loss that is a tighter upper bound of the 0-1 loss compared with conventional surrogate losses. Specifically, giving any base loss function $l(u) \geq \mathbf{1}\big(u < 0\big), u \in \mathbb{R}$, our loss $Q(\mathbf{u})$ satisfies $\sum_{i=1}^{n} \mathbf{1}\big(u_i < 0\big) \leq Q(\mathbf{u}) \leq \sum_{i=1}^{n} l(u_i)$, where $\mathbf{u} = [u_1, \cdots, u_n]$ with u_i being the classification margin of i^{th} sample, and $\mathbf{1}(\cdot)$ is an indicator function. We name it as Curriculum Loss (CL) because our loss automatically and adaptively selects samples for training, which can be deemed as a curriculum learning paradigm.

Our adaptive loss can be viewed as an adaptive tighter upper bound approximation of the expected 0-1 loss. More detailed discussion of our loss are presented in Chapter 8. Our contributions are listed as follows:

- We propose a novel loss (i.e. curriculum loss) for robust learning against label corruption. We prove that our CL is a tighter upper bound of 0-1 loss compared with conventional summation based surrogate loss. Moreover, CL can adaptively select samples for stagewise training, which bridges a connection between curriculum learning and robust learning.

- We prove that CL can be performed by a simple and fast selection algorithm with $\mathcal{O}(n \log n)$ time complexity. Moreover, our CL supports mini-batch update, which is convenient to be used as a plug-in in many deep models.
- We further propose a Noise Pruned Curriculum Loss (NPCL) to address label corruption problem by extending CL to a more general form. Our NPCL automatically prune the estimated noisy samples during training. Moreover, NPCL is also very simple and efficient, which can be used as a plug-in in deep models as well.

1.5 Research Objectives

The research objectives of this thesis is given as follows:

- Develop simple and efficient sampling-based black-box optimization framework with theoretical convergence guarantee. (Chapter 3)
- Develop efficient Bayesian optimization algorithms with theoretical convergence guarantee to address expensive black-box optimization problems. (Chapter 4)
- Develop efficient black-box integral approximation method on hypercube $[0, 1]^d$ domain. Employ the techniques as a robust initialization for black-box optimization. (Chapter 5)
- Develop efficient black-box integral approximation method on hypersphere $\mathbb{S}^{d-1}$. Apply the proposed method for kernel approximation. (Chapter 6)
- Apply the proposed integral-approximation techniques for robust deep learning. (Chapter 7 and Chapter 8)

1.6 Publications

C-1. **Yueming Lyu**, Yuan Yuan, Ivor W.Tsang "Subgroup-based Rank-1 Lattice Quasi-Monte Carlo." In *Neural Information Processing Systems (NeurIPS)*, 2020. (Chapter 5)

C-2. **Yueming Lyu**, Ivor W. Tsang. "Curriculum Loss: Robust Learning and Generalization against Label Corruption." In *International Conference on Learning Representations (ICLR)*, 2020. (Chapter 8)

C-3. Xingrui Yu, **Yueming Lyu**, Ivor W. Tsang. "Intrinsic reward driven imitation learning via generative model." In *International Conference on Machine Learning (ICML)*, 2020.

C-4. Yuan Yuan, **Yueming Lyu**, Xi Shen, Ivor W. Tsang, Dit-Yan Yeung. "Marginalized Average Attentional Network for Weakly-Supervised Learning." In *International Conference on Learning Representations (ICLR)*, 2019.

C-5. **Yueming Lyu**. "Spherical Structured Feature Maps for Kernel Approximation." In *International Conference on Machine Learning (ICML)*, 2017. (Chapter 6)

C-6. **Yueming Lyu**, Ivor W. Tsang. "Black-box Optimizer with Stochastic Implicit Natural Gradient." *ECML*, 2021. (Chapter 3)

Preprint

P-1. **Yueming Lyu**, Yuan Yuan, Ivor W.Tsang "Efficient Batch Black-box Optimization with Deterministic Regret Bounds.". *Preprint* arXiv:1905.10041, 2020. (Chapter 4)

P-2. **Yueming Lyu**, Ivor W. Tsang. "Neural Optimization Kernel: Towards Robust Deep Learning." *Preprint*, arXiv:2106.06097, 2021. (Chapter 7)

Chapter 2

Related Works

2.1 Black-box Optimization

Black-box optimization has been investigated by different communities for several decades. In the mathematics optimization community, derivative-free optimization (DFO) methods are widely studied for black-box optimization. These methods can be further divided into three categories: the direct search methods, the model-based methods, and the random search methods. Among them, the model-based methods guide the searching procedure by using the model prediction as to the surrogate, which is quite similar to the Bayesian optimization methods. We refer to [147], [12] and [104] for detailed survey of the derivative-free optimization methods. In the evolutionary computation community, researchers have developed the evolutionary algorithm [158] and evolutionary strategy methods [17] for the black-box optimization, where the latter is similar to the Nesterov random search [129] in the DFO methods since both the evolutionary strategy methods and the Nesterov random search employ the Gaussian smoothing technique to approximate the gradient. In the machine learning community, investigating the black-box optimization from the aspect of Bayesian optimization (BO) has attracted more and more attention recently. BO has been successfully applied to address many expensive black-box optimization problems, such as hyper-parameter tuning for deep networks [156], parametric policy optimization for Reinforcement learning [174], and so on.

Among different methods, Bayesian optimization methods and sampling-based derivative-free methods are promising directions for solving black-box optimization problems. We thus review the Bayesian optimization methods and sampling-based derivative-free methods in detail.

2.1.1 Bayesian Optimization

Bayesian Optimization (BO) aims to find the optimum of an unknown, usually non-convex function f. Since little information is known about the underlying function f, BO requires to estimate a surrogate function to model the unknown function. Therefore, one major challenge of BO is to seek a balance between collecting information to model the function f (exploration) and searching for an optimum based on the collected information (exploitation).

Typically, BO assumes that the underlying function f is sampled from a Gaussian process (GP) prior. BO selects the candidate solutions for evaluation by maximizing an acquisition function [85,102,125]), which balances the exploration and exploitation given all previous observations. In practice, BO can usually find an approximate maximum solution with a remarkably small number of function evaluations [150,156].

The research of BO for black-box optimization can be dated back to [126]. It becomes popular since the efficient global optimization method [86] for black-box optimization has been proposed. After that, various acquisition functions have been widely investigated both empirically and theoretically. Acquisition functions are important in BO as they determine the searching behavior. Among them, the expected improvement, probability improvement and upper confidence bound of the Gaussian process (GP-UCB) are the most widely used acquisition functions in practice [156]. Specifically, Bull [32] has proved a simple regret bound of the expected improvement method. Srinivas et al. [160] have theoretically analyzed both the cumulative regret and the simple regret bounds of the GP-UCB method.

Recently, many sophisticated acquisition functions have been studied. Hennig and Schuler [73] propose entropy search (ES) method, Hernández-Lobato et al. [74] further propose a predictive entropy search (PES) method. Both ES and PES select the candidate query by maximizing the mutual information between the query point and the global optimum in the input space. As a result, they need intensive Monte Carlo sampling that depends on the dimension of the input space. To reduce the cost of sampling, Wang et al. [170] propose a max-value entropy search method, selecting the candidate query by maximizing the mutual information between the prediction of the query and the maximum value. The mutual information is computed in one dimension, which is much easier to approximate compared to the Monte Carlo sampling. Along the line of GP-UCB, Desautels et al. [45] propose the GP-BUCB method to address the black-box optimization in a batch setting. In each batch, GP-BUCB selects the candidate queries point by point sequentially until reaching a preset batch size, according to upper confidence bound criterion [13, 160] with

a fixed mean function and an updated covariance function. Desautels et al. [45] prove the sub-linear growth bounds on the cumulative regret, which guarantees a bound on the number of required iterations to reaching close enough to the optimum. Contal et al. [40] further propose the GP-UCB-PE method, which combines the upper confidence bound strategy and the pure exploration [30] in the evaluations of the same batch. The GP-UCB-PE achieves a better upper bound on the cumulative regret compared with the GP-BUCB. Most recently, Berkenkamp et al. [24] propose a GP-UCB based method (A-GP-UCB) to handle BO with unknown hyper-parameters.

In many applications, e.g, hyper-parameter tuning and RL, it is usually preferred to process multiple function evaluations in parallel to achieve the time efficiency. In the setting of batch BO for batch black-box optimization, besides the batch, GP-UCB methods discussed above, Shah and Ghahramani [152] propose a parallel predictive entropy search method by extending the PES method [74] to the batch case. Wu et al. [176] extend the knowledge gradient method to the parallel knowledge gradient method. González et al. [61] propose a penalized acquisition function for batch selection. However, these batch methods are limited in low dimensional problems. To address the batch BO under the high-dimensional setting, Wang et al. [169] propose an ensemble BO method by integrating multiple additive Gaussian process models. However, no regret bound is analyzed in [169].

2.1.2 Sampling-based Derivative-free Optimization

CMA-ES [2, 69] is one of state-of-the-art ES methods with many successful applications. It uses second-order information to search candidate solutions by updating the mean and covariance matrix of the likelihood of candidate distributions. Despite its successful performance, the update rule combines several sophisticated components, which is not well understood. Wierstra et al. show that directly applying standard reinforce gradient descent is very sensitive to variance in high precision search for black-box optimization [173]. Thus, they propose Natural evolution strategies (NES) [173] to estimate the natural gradient for black-box optimization. However, they use the Monte Carlo sampling to approximate the Fisher information matrix (FIM), which incurs additional error and computation cost unavoidably. Along this line, [3] show the connection between the rank-μ update of CMA-ES and NES [173]. [135] further show that several ES methods can be included in an unified framework. Despite these theoretical attempts, the practical performance of these methods is still inferior to CMA-ES. Moreover, these works do not provide any convergence rate analysis, which is the key insight to expedite black-box optimizations.

Another line of research for ES-type algorithms is to reduce the variance of gradient estimators. Choromanski et al. [38] proposed to employ Quasi Monte Carlo (QMC) sampling to achieve more accurate gradient estimates. Recently, they further proposed to construct gradient estimators based on active subspace techniques [37]. Although these works can reduce sample complexity, how does the variance of these estimators influence the convergence rate remains unclear.

2.2 Black-box Integral Approximation

Integral operation is critical in a large amount of interesting machine learning applications, e.g. kernel approximation with random feature maps [140], variational inference in Bayesian learning [22], generative modeling and variational autoencoders [94]. Directly calculating an integral is usually infeasible in these real applications. Instead, researchers usually try to find an approximation for the integral. A simple and conventional approximation is Monte Carlo (MC) sampling, in which the integral is approximated by calculating the average of the i.i.d. sampled integrand values. Monte Carlo (MC) methods [65] are widely studied with many techniques to reduce the approximation error, which includes importance sampling and variance reduction techniques and more [10].

To further reduce the approximation error, Quasi-Monte Carlo (QMC) methods utilize a low discrepancy point set instead of the i.i.d. sampled point set used in the standard Monte Carlo method. There are two main research lines in the area of QMC [46,130], i.e., the digital nets/sequences and lattice rules. The Halton sequence and the Sobol sequence are the widely used representatives of digital sequences [46]. Compared with digital nets/sequences, the points set of lattice rules preserve the properties of lattice. The points partition the space into small repeating cells. Among previous research on the lattice rules, Korobov introduced integration lattice rules in [95] for an integral approximation of the periodic integrands. [155] proves that there also exist good lattice rules for non-periodic integrands. According to general lattice rules, a point set is usually constructed by enumerating the integer vectors and multiplying them with an invertible generator matrix. A general lattice rule has to check each constructed point to see whether it is inside a unit cube and discard it if it is not. The process is repeated until we reach the desired number of points. This construction process is inefficient since the checking step is required for every point. Note that rescaling the unchecked points by the maximum norm of all the points may lead to non-uniform points set in the cube.

An interesting special case of the lattice rules is the rank-1 lattice, which only requires one generating vector to construct the whole point set. Given the generating vector, rank-1 lattices can be obtained by a very simple construction form. It is thus much more efficient to construct the point set with the simple construction form. Compared with the general lattice rule, the construction form of the rank-1 lattice has already guaranteed the constructed point to be inside the unit cube, therefore, no further checks are required. We refer to [46] and [130] for a more detailed survey of QMC and rank-1 lattice.

Although the rank-1 lattice can derive a simple construction form, obtaining the generating vector remains difficult. Most methods [48, 97, 103, 108, 111, 133, 137] in the literature rely on an exhaustive computer search by optimizing some criteria to find a good generating vector. Korobov [97] suggests searching the generating vector in a form of $[1, \alpha, \alpha^2, \cdots, \alpha^{d-1}]$ with $\alpha \in \{1, \cdots, n-1\}$, where d is the dimension and n is the number of points, such that the greatest common divisor of α and n equals to 1. Sloan et al. study the component-by-component construction for the lattice rules [154]. It is a greedy search that is faster than an exhaustive search. Nuyens et al. [133] propose a fast algorithm to construct the generating vector using a component-by-component search method. Although the exhaustive checking steps are avoided compared with general lattice rules, the rank-1 lattice still requires a brute-force search for the generating vector, which is still very time-consuming, especially when the dimension and the number of points are large.

2.3 Integral Approximation for Kernel Methods and Deep Learning

2.3.1 Kernel Approximation

Kernel methods such as Gaussian processes (GPs) [144, 157, 159] and support vector machines (SVMs) [33, 56] have been successfully used in many statistical modeling and machine learning tasks. Despite of strong expressive power, kernel methods usually cannot scale up to the large scale datasets with L samples due to the need of manipulating $L \times L$ Gram matrix. Recently, random feature maps [141, 142, 164] have demonstrated their effectiveness on kernel approximation to scale up kernel methods. Roughly speaking, a shift invariant kernel $\mathrm{K}(\mathbf{x}, \mathbf{z}) = \mathbf{K}(\mathbf{x} - \mathbf{z}) : \mathbb{R}^{\mathbf{d}} \to \mathbb{C}$ can be approximated by $\mathrm{K}(x, z) \approx \Psi(\mathbf{x})^{\mathbf{T}} \Psi(\mathbf{z})$, where Ψ is the explicit mapped feature constructed as $\Psi(\mathbf{x}) = f(\mathbf{W}^T \mathbf{x})/\sqrt{N}$, where $f(\cdot)$ denotes the nonlinear function,

$\mathbf{W} \in \mathbb{R}^{d \times N}$ is constructed by N i.i.d samples drawn from a distribution defined by K. Therefore, the training and inference of kernel methods can be greatly accelerated by working directly on the primal space of $\Psi(\cdot)$. For example, Gaussian Processes (GPs) have $O(L^3)$ computation and $O(L^2)$ storage complexity. By using feature maps, it reduces to $O(N^2L + N^3)$ computation and $O(NL + N^2)$ storage complexity. All these elegant properties make random feature maps promising for large scale kernel methods. Thus, many kernel methods [41, 106, 134] have been proposed to deal with large scale statistical learning by directly working on feature maps.

Generally, two aspects of random feature maps are mostly concerned by literature for scaling up kernel methods. One is the approximation accuracy of feature maps while the other is the computational cost of feature maps construction. To achieve better approximation accuracy, [14, 178] employ QMC [47] sampling instead of standard Monte Carlo sampling to construct feature maps. By mapping QMC points on $[0, 1]^d$ through the inverse cumulative distribution function, they construct more effective feature maps. To reduce time complexity, [105] propose Fastfood to construct feature maps. Benefiting from the special structured matrix multiplication, it reduces time complexity of feature maps construction from $O(Nd)$ to $O(N \log d)$. However, it achieves computational efficiency at the expense of increasing the variance of approximation. Recently, [57] employ the property of circulant matrix to accelerate feature maps construction of Gaussian kernel without increasing the variance. [39] generalize the Fastfood and circulant feature maps to P model and particularly discuss the structured matrix with low-displacement rank. Despite of the success of P model, it still cannot achieve better approximation accuracy compared with feature maps obtained with fully Gaussian matrix.

2.3.2 Deep Learning Theory from Optimization and Kernel Perspective

Deep neural networks (DNNs) have obtained great success in many applications, including computer vision [71], natural language processing [175], and reinforcement learning [124], etc. However, the theory of deep learning is much less explored compared with its great empirical success. A key challenge of deep learning theory is that deep neural networks are heavily overparameterized. Namely, the number of parameters is much larger than training samples. In practice, as the depth and width increasing, the performance of deep NN also becomes better [161, 184], which is far beyond the traditional learning theory regime.

In the traditional neural networks and kernel methods literature, it is well known the connection between the infinite width neural networks and Gaussian process [75], and the universal approximation power of NN [112]. However, these theories cannot explain why the success of deep neural networks. A recent work, Neural Tangent Kernel [80] (NTK), shows the connection between training an infinite-width NN and performing functional gradient descent in a Reproducing Kernel Hilbert Space(RKHS) associated with the NTK. Because of the convexity of the functional optimization problem, Jacot et al. show the global convergence for infinite-width NN under the NTK regime. Along this direction, Hanin et al. [68] analyze the NTK with finite width and depth. Shankar et al. [153] empirically investigate the performance of some simple compositional kernels, NTKs, and deep neural networks. Nitanda et al. [132] further show the minimax optimal convergence rate of average stochastic gradient descent in a two-layer NTK regime.

Despite the success of NTK [80] on showing the global convergence of NN, its expressive power is limited. Zhu et al. [4] provide an example that shallow kernel methods (including NTK) need a much larger number of training samples to achieve the same small population risk compared with a three-layer ResNet. They further point out the importance of hierarchical learning in deep neural networks [5]. In [5], they give the theoretical analysis of learning a target network family with square activation function under deep NN regime. Besides, there are quite a few works focus on the analysis of two-layer networks [18,27,52,88,113,180] and shallow kernel methods without hierarchical learning [9,44,58,109,190].

Although some particular examples show deep models have more powerful expressive power than shallow ones [4,5,54], how and why deep neural networks benefit from the depth remain unclear. Zhu et al. [5] highlight the importance of a backward feature correction. In this thesis, we investigate the deep neural networks from a different kernel method perspective.

2.3.3 Robust Deep Learning under Label Noise

Noise corruption is a common phenomenon in our daily life. For instance, noisy corrupted (wrong) labels may be resulted from annotating for similar objects [162, 183], crawling images and labels from websites [76,165] and creating training sets by program [93,145]. Learning with noisy labels is thus an promising area.

Deep neural networks (DNNs) have great expressive power (model complexity) to learn challenging tasks. However, DNNs also undertake a higher risk of overfitting to the data. Although many regularization techniques, such as adding regularization

terms, data augmentation, weight decay, dropout and batch normalization, have been proposed, generalization is still vitally important for deep learning to fully exploit the super-expressive power. [185] show that DNNs can even fully memorize samples with incorrectly corrupted labels. Such label corruption significantly degenerates the generalization performance of deep models. This calls a lot of attention on robustness in deep learning with noisy labels.

Robustness of 0-1 loss: The problem resulted from data corruption or label corruption is that test distribution is different from training distribution. [77] analyzed the adversarial risk that the test distribution density is adversarially changed within a limited f-divergence (e.g. KL-divergence) from the training distribution density. They show that there is a monotonic relationship between the (empirical) risk and the (empirical) adversarial risk when the 0-1 loss function is used. This suggests that minimizing the empirical risk with the 0-1 loss function is equivalent to minimize the empirical adversarial risk (worst-case risk). When we train a model based on the corrupted training distribution, we want our model to perform well on the clean distribution. Since we do not know the clean distribution, we want our model to perform well for the worst case estimate of the clean distribution in some constrained set. It is thus natural to employ the worst-case classification risk of the estimated clean distribution as the objective. Note that the worst-case classification risk is an upper bound of the classification risk of the true clean distribution, minimizing the worst-case risk can usually decrease the true risk. When we employ the 0-1 loss, because of the equivalence between the classification risk and the worst-case classification risk, we can directly minimize the classification risk under the corrupted training distribution instead of minimizing the worst-case classification risk.

From the learning perspective, the 0-1 loss is more robust to outliers compared with an unbounded (convex) loss (e.g. hinge loss) [122]. This is due to unbounded convex losses putting much weight on the outliers (with a large loss value) when minimizing the losses [122]. If the unbounded (convex) loss is employed in deep network models, this becomes more prominent. Since training loss of deep networks can often be minimized to zero, outlier with a large loss has a large impact on the model. On the other hand, the 0-1 loss treats each training sample equally. Thus, each sample does not have too much influence on the model. Therefore, the model is tolerant of a small number of outliers.

Although the 0-1 loss has many robust properties, its non-differentiability and zero gradients make it difficult to optimize. One possible way to alleviate this problem is to seek an upper bound of the 0-1 loss that is still efficient to optimize but tighter than

conventional (convex) losses. Such a tighter upper bound of the 0-1 loss can reduce the influence of the noisy outliers compared with conventional (convex) losses. At the same time, it is easier to optimize compared with the 0-1 loss. When minimizing the upper bound surrogate, we expect that the 0-1 loss objective is also minimized.

Learnability under large noise rate: The 0-1 loss cannot deal with large noise rate. When the noise rate becomes large, the systematic error (due to label corruption) grows up and becomes not negligible. As a result, the model's generalization performance will degenerate due to this systematic error. To reduce the systematic error produced by training with noisy labels, several methods have been proposed. They can be categorized into three kinds: transition matrix based method [59, 138, 163], regularization based method [123] and sample selection based method [67, 84]. Among them, sample selection based method is one promising direction that selects samples to reduce noisy ratio for training. These methods are based on the idea of curriculum learning [23] which is one successful method that trains the model gradually with samples ordered in a meaningful sequence. Although they achieve success to some extents, most of these methods are heuristic based.

To efficiently minimize the 0-1 loss while keeping the robust properties, this thesis proposes a novel loss that is a tighter upper bound of the 0-1 loss compared with conventional surrogate losses. Specifically, giving any base loss function $l(u) \geq \mathbf{1}\big(u < 0\big), u \in \mathbb{R}$, our loss $Q(\mathbf{u})$ satisfies $\sum_{i=1}^{n} \mathbf{1}\big(u_i < 0\big) \leq Q(\mathbf{u}) \leq \sum_{i=1}^{n} l(u_i)$, where $\mathbf{u} = [u_1, \cdots, u_n]$ with u_i being the classification margin of i^{th} sample, and $\mathbf{1}(\cdot)$ is an indicator function. We name it as Curriculum Loss (CL) because our loss automatically and adaptively selects samples for training, which can be deemed as a curriculum learning paradigm.

Curriculum Learning: Curriculum learning is a general learning methodology that achieves success in many area. The very beginning work of curriculum learning [23] trains a model gradually with samples ordered in a meaningful sequence, which has improved performance on many problems. Since the curriculum in [23] is predetermined by prior knowledge and remained fixed later, which ignores the feedback of learners, Kumar et al. [101] further propose Self-paced learning that selects samples by alternative minimization of an augmented objective. Jiang et al. [82] propose a self-paced learning method to select samples with diversity. After that, Jiang et al. [83] propose a self-paced curriculum strategy that takes different priors into consideration. Although these methods achieve success, the relation between the augmented objective of self-paced learning and the original objective (e.g. cross en-

tropy loss for classification) is not clear. In addition, as stated in [84], the alternative update in self-paced learning is not efficient for training deep networks.

Learning with Noisy Labels: The most related works are the sample selection based methods for robust learning. This kind of works are inspired by curriculum learning [23]. Among them, Jiang et al. [84] propose to learn the curriculum from data by a mentor net. They use the mentor net to select samples for training with noisy labels. Co-teaching [67] employs two networks to select samples to train each other and achieve good generalization performance against large rate of label corruption. Co-teaching+ [182] extends Co-teaching by selecting samples with disagreement of prediction of two networks. Compared with Co-teaching/Co-teaching+, our CL is a simple plugin for a single network. Thus both space and time complexity of CL are half of Co-teaching's. Recently, [188] propose a generalized Cross-entropy loss for robust learning.

Chapter 3

Implicit Natural Gradient Optimization

3.1 Chapter Abstract

This chapter presents a novel theoretical framework for black-box optimization, in which our method performs stochastic updates with an implicit natural gradient of an exponential-family distribution. Theoretically, we prove the convergence rate of our framework with full matrix update for convex functions under Gaussian distribution. Our methods are very simple and contain fewer hyper-parameters than CMA-ES [69]. Empirically, our method with full matrix update achieves competitive performance compared with one of the state-of-the-art methods CMA-ES on benchmark test problems. Moreover, our methods can achieve high optimization precision on some challenging test functions (e.g., l_1-norm ellipsoid test problem and Levy test problem), while methods with explicit natural gradient, i.e., IGO [135] with full matrix update can not. This shows the efficiency of our methods. Furthermore, efficient algorithms for b discrete black-box optimization are proposed under the proposed framework.

3.2 Optimization with Exponential-family Sampling

Notation and Symbols: Denote $\|\cdot\|_2$ and $\|\cdot\|_F$ as the spectral norm and Frobenius norm for matrices, respectively. Define $\|Y\|_{tr} := \sum_i |\lambda_i|$, where λ_i denotes the i^{th} eigenvalue of matrix Y. Notation $\|\cdot\|_2$ will also denote l_2-norm for vectors. Symbol $\langle \cdot, \cdot \rangle$ denotes inner product under l_2-norm for vectors and inner product under Frobenius norm for matrices. Define $\|\boldsymbol{x}\|_C := \sqrt{\langle \boldsymbol{x}, C\boldsymbol{x} \rangle}$. Denote $\mathcal{S}^+$ and $\mathcal{S}^{++}$ as the set of positive semi-definite matrices and the set of positive definite matrices, respectively.

Denote $\Sigma^{\frac{1}{2}}$ as the symmetric positive semi-definite matrix such that $\Sigma = \Sigma^{\frac{1}{2}}\Sigma^{\frac{1}{2}}$ for $\Sigma \in \mathcal{S}^+$.

We aim at minimizing a proper function $f(\boldsymbol{x})$, $\boldsymbol{x} \in \mathcal{X}$ with only function queries, which is known as black-box optimization.

Due to the lack of gradient information for black-box optimization, we here present an exponential-family sampling trick to relax any black-box optimization problem. Specifically, the objective is relaxed as the expectation of $f(\boldsymbol{x})$ under a parametric distribution $p(\boldsymbol{x};\eta)$ with parameter η, i.e., $J(\boldsymbol{\eta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\eta})}[f(\boldsymbol{x})]$ [173]. The optimal parameter $\boldsymbol{\eta}$ is found by minimizing $J(\boldsymbol{\eta})$ as

$$\min_{\boldsymbol{\eta}} \left\{ \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\eta})}[f(\boldsymbol{x})] \right\}. \tag{3.1}$$

This relaxed problem is minimized when the probability mass is all assigned on the minimum of $f(\boldsymbol{x})$. The distribution p is the sampling distribution for black-box function queries. Note that p can be either continuous or discrete distribution.

In this work, we assume that the distribution $p(\boldsymbol{x};\boldsymbol{\eta})$ is an exponential-family distribution:

$$p(\boldsymbol{x};\boldsymbol{\eta}) = h(\boldsymbol{x}) \exp\left\{ \langle \phi(\boldsymbol{x}), \boldsymbol{\eta} \rangle - A(\boldsymbol{\eta}) \right\}, \tag{3.2}$$

where $\boldsymbol{\eta}$ and $\phi(\boldsymbol{x})$ are the natural parameter and sufficient statistic, respectively. And $A(\eta)$ is the log partition function defined as:

$$A(\eta) = \log \int \exp\left\{ \langle \phi(\boldsymbol{x}), \boldsymbol{\eta} \rangle h(\boldsymbol{x}) d\boldsymbol{x}. \right. \tag{3.3}$$

We call an exponential-family distribution minimal when there is a one-to-one mapping between the mean parameter $\boldsymbol{m} := \mathbb{E}_p[\phi(\boldsymbol{x})]$ and natural parameter $\boldsymbol{\eta}$. This one-to-one mapping ensures that we can reparameterize $J(\boldsymbol{\eta})$ as $\tilde{J}(\boldsymbol{m}) = J(\boldsymbol{\eta})$ [7, 90]. $\tilde{J}$ is w.r.t parameter $\boldsymbol{m}$, while J is w.r.t parameter $\boldsymbol{\eta}$.

To minimize the objective $\tilde{J}(\boldsymbol{m})$, we desire the updated distribution lying in a trust region of the previous distribution at each step. Formally, we update the mean parameters by solving the following optimization problem.

$$\boldsymbol{m}_{t+1} = \arg\min_{\boldsymbol{m}} \left\langle \boldsymbol{m}, \nabla_{\boldsymbol{m}} \tilde{J}(\boldsymbol{m}_t) \right\rangle + \frac{1}{\beta_t} \mathrm{KL}\left(p_m \| p_{m_t}\right), \tag{3.4}$$

where $\nabla_{\boldsymbol{m}} \tilde{J}(\boldsymbol{m}_t)$ denotes the gradient at $\boldsymbol{m} = \boldsymbol{m}_t$.

The KL-divergence term measures how close the updated distribution and the previous distribution. For an exponential-family distribution, the KL-divergence term in (3.4) is equal to Bregman divergence between $\boldsymbol{m}$ and $\boldsymbol{m}_t$ [16]:

$$\mathrm{KL}\left(p_m \| p_{m_t}\right) = A^*(\boldsymbol{m}) - A^*(\boldsymbol{m}_t) - \langle \boldsymbol{m} - \boldsymbol{m}_t, \nabla_{\boldsymbol{m}} A^*(\boldsymbol{m}_t) \rangle, \tag{3.5}$$

where $A^*(\boldsymbol{m})$ is the convex conjugate of $A(\boldsymbol{\eta})$. Thus, the problem (3.4) is a convex optimization problem, and it has a closed-form solution.

3.3 Implicit Natural Gradient

Intractability of Natural Gradient for Black-box Optimization: Natural gradient [6] can capture information geometry structure during optimization, which enables us to take advantage of the second-order information to accelerate convergence. Direct computation of natural gradient needs the inverse of Fisher information matrix (FIM), which needs to estimate the FIM. The method in [143] provides an alternative way to compute natural gradient without computation of FIM. However, it relies on the exact gradient, which is impossible for black-box optimization.

Hereafter, we propose a novel stochastic implicit natural gradient algorithms for black-box optimization of continuous and discrete variables in Section 3.4 and Section 3.5, respectively. We first show how to compute the implicit natural gradient. In problem Eq.(3.4), we take the derivative w.r.t $\boldsymbol{m}$, and set it to zero, also note that $\nabla_{\boldsymbol{m}} A^*(\boldsymbol{m}) = \boldsymbol{\eta}$ [143], we can obtain that

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \nabla_{\boldsymbol{m}} \tilde{J}(\boldsymbol{m}_t) \tag{3.6}$$

Natural parameters $\boldsymbol{\eta}$ of the distribution lies on a Riemannian manifold with metric tensor specified by the Fisher Information Matrix:

$$\boldsymbol{F}(\boldsymbol{\eta}) := \mathbb{E}_p \left[\nabla_\eta \log p(\boldsymbol{x}; \boldsymbol{\eta}) \nabla_\eta \log p(\boldsymbol{x}; \boldsymbol{\eta})^\top \right] \tag{3.7}$$

For exponential-family with the minimal representation, the natural gradient has a simple form for computation.

Theorem 1. *[91, 143] For an exponential-family in the minimal representation, the natural gradient w.r.t $\boldsymbol{\eta}$ is equal to the gradient w.r.t. $\boldsymbol{m}$, i.e.,*

$$\boldsymbol{F}(\boldsymbol{\eta})^{-1} \nabla_{\boldsymbol{\eta}} J(\boldsymbol{\eta}) = \nabla_{\boldsymbol{m}} \tilde{J}(\boldsymbol{m}) \tag{3.8}$$

Remark: Theorem 1 can be easily obtained by the chain rule and the fact $\boldsymbol{F}(\boldsymbol{\eta}) = \frac{\partial^2 A(\boldsymbol{\eta})}{\partial \boldsymbol{\eta} \partial \boldsymbol{\eta}^\top}$. It enables us to compute the natural gradient implicitly without computing the inverse of the Fisher information matrix. As shown in Theorem 1, the update rule in (3.6) is equivalent to the natural gradient update w.r.t $\boldsymbol{\eta}$ in (3.9):

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \boldsymbol{F}(\boldsymbol{\eta}_t)^{-1} \nabla_\eta J(\boldsymbol{\eta}_t) \tag{3.9}$$

Thus, update rule in (3.6) selects the steepest descent direction along the Riemannian manifold induced by the Fisher information matrix as natural gradient descent. It can take the second-order information to accelerate convergence.

3.4 Update Rule for Gaussian Sampling

We first present an update method for the case of Gaussian sampling for continuous optimization. For other distributions, we can derive the update rule in a similar manner.

For a Gaussian distribution $p := \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ with mean $\boldsymbol{\mu}$ and covariance matrix Σ, the natural parameters $\eta = \{\boldsymbol{\eta}_1, \boldsymbol{\eta}_2\}$ are given as follows:

$$\boldsymbol{\eta}_1 := \Sigma^{-1}\boldsymbol{\mu} \tag{3.10}$$

$$\boldsymbol{\eta}_2 := -\frac{1}{2}\Sigma^{-1} \tag{3.11}$$

The related mean parameters $\boldsymbol{m} = \{\boldsymbol{m}_1, \boldsymbol{m}_2\}$ are given as:

$$\boldsymbol{m}_1 := \mathbb{E}_p[\boldsymbol{x}] = \boldsymbol{\mu} \tag{3.12}$$

$$\boldsymbol{m}_2 := \mathbb{E}_p[\boldsymbol{x}\boldsymbol{x}^\top] = \boldsymbol{\mu}\boldsymbol{\mu}^\top + \Sigma \tag{3.13}$$

Using the chain rule, the gradient with respect to mean parameters can be expressed in terms of the gradients w.r.t $\boldsymbol{\mu}$ and Σ [90, 92] as:

$$\nabla_{\boldsymbol{m}_1}\tilde{J}(\boldsymbol{m}) = \nabla_{\boldsymbol{\mu}}\tilde{J}(\boldsymbol{m}) - 2[\nabla_\Sigma \tilde{J}(\boldsymbol{m})]\boldsymbol{\mu} \tag{3.14}$$

$$\nabla_{\boldsymbol{m}_2}\tilde{J}(\boldsymbol{m}) = \nabla_\Sigma \tilde{J}(\boldsymbol{m}) \tag{3.15}$$

It follows that

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + 2\beta_t \nabla_\Sigma \tilde{J}(\boldsymbol{m}_t) \tag{3.16}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \Sigma_{t+1} \nabla_{\boldsymbol{\mu}} \tilde{J}(\boldsymbol{m}_t) \tag{3.17}$$

Note that $\tilde{J}(\boldsymbol{m}) = \mathbb{E}_p[f(\boldsymbol{x})]$, the gradients of $\tilde{J}(\boldsymbol{m})$ w.r.t μ and Σ can be obtained by log-likelihood trick as Theorem 2.

Theorem 2. *[173] The gradient of the expectation of an integrable function $f(\boldsymbol{x})$ under a Gaussian distribution $p := \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ with respect to the mean $\boldsymbol{\mu}$ and the covariance Σ can be expressed as Eq.(3.18) and Eq.(3.19), respectively.*

$$\nabla_{\boldsymbol{\mu}}\mathbb{E}_p[f(\boldsymbol{x})] = \mathbb{E}_p\left[\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})f(\boldsymbol{x})\right] \tag{3.18}$$

$$\nabla_\Sigma \mathbb{E}_p[f(\boldsymbol{x})] = \frac{1}{2}\mathbb{E}_p\left[\left(\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^\top\Sigma^{-1} - \Sigma^{-1}\right) f(\boldsymbol{x})\right] \tag{3.19}$$

Together Theorem 2 with Eq. (3.16) and (3.17), we present the update with only function queries as:

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \beta_t \mathbb{E}_p\left[\left(\Sigma_t^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_t)(\boldsymbol{x}-\boldsymbol{\mu}_t)^\top\Sigma_t^{-1} - \Sigma_t^{-1}\right) f(\boldsymbol{x})\right] \tag{3.20}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \Sigma_{t+1}\mathbb{E}_p\left[\Sigma_t^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_t)f(\boldsymbol{x})\right] \tag{3.21}$$

Remark: Our method updates the inverse of the covariance matrix instead of the covariance matrix itself.

3.4.1 Stochastic Update

The above gradient update needs the expectation of a black-box function. However, this expectation does not have a closed-form solution. Here, we estimate the gradient w.r.t μ and Σ by Monte Carlo sampling. Eq.(3.20) and (3.21) enable us to estimate the gradient by the function queries of $f(x)$ instead of $\nabla f(x)$. This property is very crucial for black-box optimization because gradient ($\nabla f(x)$) is not available.

Update rules using Monte Carlo sampling are given as:

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \frac{\beta_t}{N}\sum_{i=1}^{N}\left[\left(\Sigma_t^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_t)(\boldsymbol{x}_i - \boldsymbol{\mu}_t)^\top \Sigma_t^{-1} - \Sigma_t^{-1}\right) f(\boldsymbol{x}_i)\right] \tag{3.22}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \frac{\beta_t}{N}\sum_{i=1}^{N}\left[\Sigma_{t+1}\Sigma_t^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_t) f(\boldsymbol{x}_i)\right] \tag{3.23}$$

To avoid scaling problem of $f(x)$, we use a monotonic score function $h(\cdot)$ to transform $f(x)$ as:

$$h(f(\boldsymbol{x}_i)) = N\frac{\log \widehat{i}}{\sum_{j=1}^{N}\log j} \tag{3.24}$$

where $\widehat{i}$ denotes the ranking index of $f(\boldsymbol{x}_i)$ (index after non-decreasingly sort) among N samples $f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N)$. (Break ties by some determinate rules).

In addition, we can use a normalization as the monotonic transformation:

$$g(f(\boldsymbol{x}_i)) = \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{\widehat{\sigma}} \tag{3.25}$$

where $\widehat{\mu}$ and $\widehat{\sigma}$ denote mean and stand deviation of function values in a batch of samples.

Plug Eq.(3.25) into Eq.(3.22) and (3.23), we obtain that

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \beta\sum_{i=1}^{N}\frac{h(f(\boldsymbol{x}_i))}{N}\left(\Sigma_t^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_t)(\boldsymbol{x}_i - \boldsymbol{\mu}_t)^\top \Sigma_t^{-1}\right) \tag{3.26}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t\sum_{i=1}^{N}\frac{h(f(\boldsymbol{x}_i))}{N}\Sigma_{t+1}\Sigma_t^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_t) \tag{3.27}$$

The update rule in Eq.(3.26) and Eq.(3.27) does not require the exact value. It only needs the ranking of the observations for a set of samples. This is useful when the

Algorithm 1 INGO

Input: Number of Samples N, step-size β.
while Termination condition not satisfied **do**
  Take i.i.d samples $\boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ for $i \in \{1, \cdots N\}$.
  Set $\boldsymbol{x}_i = \boldsymbol{\mu}_t + \Sigma_t^{\frac{1}{2}} \boldsymbol{z}_i$ for $i \in \{1, \cdots N\}$.
  Query the batch observations $\{f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N)\}$
  Compute $\widehat{\sigma} = \text{std}(f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N))$.
  Compute $\widehat{\mu} = \frac{1}{N} \sum_{i=1}^{N} f(\boldsymbol{x}_i)$.
  Set $\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \beta \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{N\widehat{\sigma}} \Sigma_t^{-\frac{1}{2}} \boldsymbol{z}_i \boldsymbol{z}_i^\top \Sigma_t^{-\frac{1}{2}}$.
  Set $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{N\widehat{\sigma}} \Sigma_{t+1} \Sigma_t^{-\frac{1}{2}} \boldsymbol{z}_i$
end while

function value is not observable, e.g., only a preference list of customers is given. The downside of the update based on the ranking is that the gradient estimator is biased. In addition, the information of function is lost. To take advantage of the smoothness of function while avoiding scale problem, we present an unbiased estimator for gradient update as follows:

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \beta \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{N\widehat{\sigma}} \left(\Sigma_t^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_t)(\boldsymbol{x}_i - \boldsymbol{\mu}_t)^\top \Sigma_t^{-1} \right) \tag{3.28}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{N\widehat{\sigma}} \Sigma_{t+1} \Sigma_t^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_t) \tag{3.29}$$

where $\widehat{\sigma}$ denotes the stand deviation of function values in a batch of samples, i.e., $\widehat{\sigma} = \text{std}(f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N))$.

We present our black-box optimization algorithm in Alg. 1. We can select a set of random samples $[\boldsymbol{Z}, -\boldsymbol{Z}]$, this trick still leads to an unbiased estimator for the Gaussian distribution [117]. We can use the structured samples in [117] to reduce the variance of estimators. Alg. 1 employs the normalization transformation, we can also employ the ranking scores to update Σ and function values to update $\boldsymbol{\mu}$.

The update of mean $\boldsymbol{\mu}$ in Alg. 1 is properly scaled by Σ. Moreover, our method updates the inverse of the covariance matrix instead of the covariance matrix itself, which provides us a stable way to update covariance independent of its scale. Thus, our method can update properly when the algorithm adaptively reduces variance for high precision search. In contrast, directly applying standard reinforce type gradient update is unstable as shown in [173].

Algorithm 2 Fast INGO-u

Input: Number of Samples N, step-size β.
while Termination condition not satisfied **do**
Take i.i.d samples $\boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ for $i \in \{1, \cdots N/2\}$.
Set $\boldsymbol{z}_{i+N/2} = -\boldsymbol{z}_i$ for $i \in \{1, \cdots N/2\}$.
Set $\boldsymbol{x}_i = \boldsymbol{\mu}_t + \boldsymbol{\sigma}_t \odot \boldsymbol{z}_i$ for $i \in \{1, \cdots N\}$.
Query the batch observations $\{f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N)\}$
Compute weights $w_i = \log \widehat{i} / \sum_{j=1}^{N} \log j$ for $i \in \{1, \cdots, N\}$ according to Eq.(3.25).
Set $\boldsymbol{\sigma}_{t+1}^{-2} = (1-\beta)\boldsymbol{\sigma}_t^{-2} + \beta \boldsymbol{\sigma}_t^{-2} \odot \left(\sum_{i=1}^{N} w_i \boldsymbol{z}_i^2 \right)$.
Compute $\widehat{\sigma} = \mathrm{std}(f_1, \cdots f_N)$.
Set $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta \boldsymbol{\sigma}_{t+1}^2 \odot \boldsymbol{\sigma}_t^{-1} \odot \left(\sum_{i=1}^{N} \frac{f_i}{N\widehat{\sigma}} \boldsymbol{z}_i \right)$.
end while

3.4.2 Mean field approximation for acceleration

Alg. 1 needs to compute the inverse of covariance matrix, which has $\mathcal{O}(d^3)$ complexity. In this section, we present a fast algorithm for separate problems.

Suppose the covariance matrix is diagonal with diagonal elements denoted as $\boldsymbol{\sigma}^2$. From (3.26) and (3.27), we can obtain the update rule as

$$\boldsymbol{\sigma}_{t+1}^{-2} = (1-\beta_t)\boldsymbol{\sigma}_t^{-2} + \beta_t \boldsymbol{\sigma}_t^{-2} \odot \big(\sum_{i=1}^{N} w_i \boldsymbol{z}_i^2 \big) \tag{3.30}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \boldsymbol{\sigma}_{t+1}^2 \odot \boldsymbol{\sigma}_t^{-1} \odot \big(\sum_{i=1}^{N} w_i \boldsymbol{z}_i \big) \tag{3.31}$$

where $\odot$ denotes element-wise product. And the power operation denotes the element-wise operation. And $\boldsymbol{x}_i = \boldsymbol{\mu}_t + \boldsymbol{\sigma}_t \odot \boldsymbol{z}_i$. And $w_i = \log \widehat{i} / \sum_{j=1}^{N} \log j$.

We present an algorithm using unbiased gradient estimator in Alg. 2. It only involves element-wise operation in vectors, which is very simple to implement. The complexity in per iteration is $\mathcal{O}(Nd)$, which scales well for high-dimensional optimization.

3.4.3 Direct Update for $\boldsymbol{\mu}$ and Σ

We provide an alternative updating equation with simple concept and derivation. The implicit natural gradient algorithms are working on the natural parameter space. Alternatively, we can also directly work on the $\boldsymbol{\mu}$ and Σ parameter space. Formally,

we derive the update rule by solving the following trust region optimization problem.

$$\boldsymbol{\theta}_{t+1} = \arg\min_{\boldsymbol{\theta}} \left\langle \boldsymbol{\theta}, \nabla_{\boldsymbol{\theta}} \bar{J}(\boldsymbol{\theta}_t) \right\rangle + \frac{1}{\beta_t} \mathrm{KL}\left(p_\theta \| p_{\theta_t}\right) \tag{3.32}$$

where $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \Sigma\}$ and $\bar{J}(\boldsymbol{\theta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})] = J(\boldsymbol{\eta})$.

For Gaussian sampling, the optimization problem in (3.32) is a convex optimization problem. We can achieve a closed-form update given in Theorem 3:

Theorem 3. *For Gaussian distribution with parameter $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \Sigma\}$, problem (3.32) is convex w.r.t $\boldsymbol{\theta}$. The optimum of problem (3.32) leads to closed-form update (3.33) and (3.34):*

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + 2\beta_t \nabla_\Sigma \bar{J}(\boldsymbol{\theta}_t) \tag{3.33}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \Sigma_t \nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t) \tag{3.34}$$

Remark: Comparing the update rule in Theorem 3 with Eq.(3.16) and (3.17), we can observe that the only difference is in the update of $\boldsymbol{\mu}$. In Eq.(3.34), the update employs Σ_t, while the update in Eq.(3.17) employs Σ_{t+1}. The update in Eq.(3.17) takes one step look ahead information, it helps to improve sample efficiency.

We can obtain the black-box update for $\boldsymbol{\mu}$ and Σ by Theorem 3 and Theorem 2. The update rule is given as follows:

$$\begin{aligned}
\Sigma_{t+1}^{-1} &= \Sigma_t^{-1} + \beta_t \mathbb{E}_p\left[\left(\Sigma_t^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_t)(\boldsymbol{x}-\boldsymbol{\mu}_t)^\top \Sigma_t^{-1} - \Sigma_t^{-1}\right) f(\boldsymbol{x})\right] \\
\boldsymbol{\mu}_{t+1} &= \boldsymbol{\mu}_t - \beta_t \mathbb{E}_p\left[(\boldsymbol{x}-\boldsymbol{\mu}_t) f(\boldsymbol{x})\right]
\end{aligned} \tag{3.35}$$

Using the score function $g(\cdot)$, we can obtain Monte Carlo approximation update as

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \beta \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{N\widehat{\sigma}} \left(\Sigma_t^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_t)(\boldsymbol{x}_i - \boldsymbol{\mu}_t)^\top \Sigma_t^{-1}\right) \tag{3.36}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{N\widehat{\sigma}} (\boldsymbol{x}_i - \boldsymbol{\mu}_t) \tag{3.37}$$

From Eq.(3.37), we can see that the update rule for $\boldsymbol{\mu}$ is similar to that of CMA-ES. In contrast, the update rule for covariance matrix Σ is the same as the implicit natural gradient update in Eq.(3.26), which updates the inverse covariance matrix instead of the covariance matrix itself in CMA-ES. The algorithm is summarized in Algorithm 3.

Algorithm 3 INGOstep

Input: Number of Samples N, step-size β.
while Termination condition not satisfied **do**
  Take i.i.d samples $\boldsymbol{z}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ for $i \in \{1, \cdots N\}$.
  Set $\boldsymbol{x}_i = \boldsymbol{\mu}_t + \Sigma_t^{\frac{1}{2}} \boldsymbol{z}_i$ for $i \in \{1, \cdots N\}$.
  Query the batch observations $\{f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N)\}$
  Compute $\widehat{\sigma} = \text{std}(f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N))$.
  Compute $\widehat{\mu} = \frac{1}{N} \sum_{i=1}^N f(\boldsymbol{x}_i)$.
  Set $\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \beta \sum_{i=1}^N \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{N\widehat{\sigma}} \Sigma_t^{-\frac{1}{2}} \boldsymbol{z}_i \boldsymbol{z}_i^\top \Sigma_t^{-\frac{1}{2}}$.
  Set $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta \sum_{i=1}^N \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{N\widehat{\sigma}} \Sigma_t^{\frac{1}{2}} \boldsymbol{z}_i$
end while

3.5 Optimization for Discrete Variable

Binary Optimization: For function $f(\boldsymbol{x})$ over binary variable $\boldsymbol{x} \in \{0, 1\}^d$, we employ the Bernoulli distribution with parameter $\boldsymbol{p} = [p_1, \cdots, p_d]^\top$ as the underlying distribution, where p_i denote the probability of $x_i = 1$. Let $\boldsymbol{\eta}$ denote the natural parameter, then we know the inverse parameter mapping $\boldsymbol{p} = \frac{1}{1+e^{-\boldsymbol{\eta}}}$. The mean parameter is $\boldsymbol{m} = \boldsymbol{p}$ [131].

From Eq.(3.6), we know that

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \nabla_{\boldsymbol{p}} \mathbb{E}_{\boldsymbol{p}}[f(\boldsymbol{x})] \tag{3.38}$$

$$= \boldsymbol{\eta}_t - \beta_t \mathbb{E}_p [f(\boldsymbol{x}) \boldsymbol{h}] \tag{3.39}$$

where $\boldsymbol{h}_i = \frac{1}{p_i} \boldsymbol{1}(\boldsymbol{x}_i = 1) - \frac{1}{1-p_i} \boldsymbol{1}(\boldsymbol{x}_i = 0)$. Detailed derivation can be found in Appendix 10.8.

Approximate the gradient by Monte Carlo sampling, we obtain that

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \frac{1}{N} \sum_{n=1}^N f(\boldsymbol{x}^n) \boldsymbol{h}^n \tag{3.40}$$

where $\boldsymbol{h}_i^n = \frac{1}{p_i} \boldsymbol{1}(\boldsymbol{x}_i^n = 1) - \frac{1}{1-p_i} \boldsymbol{1}(\boldsymbol{x}_i^n = 0)$.

In order to achieve stable update, we normalize function value by its standard deviation $\widehat{\sigma}$ in a batch, i.e., $\widehat{\sigma} = \text{std}(f_1, ..., f_N)$. The normalized update is given as follows.

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \frac{1}{N\widehat{\sigma}} \sum_{n=1}^N f(\boldsymbol{x}^n) \boldsymbol{h}^n \tag{3.41}$$

General Discrete Optimization: Similarly, for function $f(\boldsymbol{x})$ over discrete variable $\boldsymbol{x} \in \{1, \cdots, K\}^d$, we employ categorical distribution with parameter $\boldsymbol{P} =$

Algorithm 4 General Framework

Input: Number of Samples N, step-size β.
while Termination condition not satisfied **do**
  Construct unbiased estimator $\widehat{g}_t$ of gradient w.r.t $\boldsymbol{\mu}$.
  Construct unbiased/biased estimator $\widehat{G}_t \in \mathcal{S}^{++}$ such that $bI \preceq \widehat{G}_t \preceq \frac{\gamma}{2} I$
  Set $\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + 2\beta \widehat{G}_t$.
  Set $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta \Sigma_{t+1} \widehat{g}_t$.
end while

$[\boldsymbol{p}_1, \cdots, \boldsymbol{p}_d]^\top$ as the underlying distribution, where the ij-th element of $\boldsymbol{P}$ ($\boldsymbol{P}_{ij}$) denote the probability of $\boldsymbol{x}_i = j$. Let $\boldsymbol{\eta} \in \mathcal{R}^{d \times K}$ denote the natural parameter, then we know the inverse parameter mapping $\boldsymbol{P}_{ij} = \frac{e^{\boldsymbol{\eta}_{ij}}}{\sum_{j=1}^{K} e^{\boldsymbol{\eta}_{ij}}}$. The mean parameter is $\boldsymbol{m} = \boldsymbol{P}$ [131].

From Eq.(3.6), we know that

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \nabla_{\boldsymbol{P}} \mathbb{E}_{\boldsymbol{P}}[f(\boldsymbol{x})] \tag{3.42}$$

$$= \boldsymbol{\eta}_t - \beta_t \mathbb{E}_p \left[f(\boldsymbol{x}) \boldsymbol{H} \right] \tag{3.43}$$

where $\boldsymbol{H}_{ij} = \frac{1}{\boldsymbol{P}_{ij}} \mathbf{1}(\boldsymbol{x}_i = j)$. Detailed derivation can be found in Appendix 10.8.

Approximate the gradient by Monte Carlo sampling,

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \frac{1}{N} \sum_{n=1}^{N} f(\boldsymbol{x}^n) \boldsymbol{H}^n \tag{3.44}$$

where $\boldsymbol{H}_{ij}^n = \frac{1}{\boldsymbol{P}_{ij}} \mathbf{1}(\boldsymbol{x}_i^n = j)$. We can also normalize the update by the std $\widehat{\sigma}$. It is worth noting that for each row of $\boldsymbol{\eta}$, plus an offset constant does not change the probability $\boldsymbol{P}$. Thus, we can plus a constant to each row of $\boldsymbol{\eta}$ to avoid numerical problem.

3.6 Convergence Rate

We first show a general framework for continuous optimization in Alg. 4. Alg. 4 employs an unbiased estimator ($\widehat{g}_t$) for gradient $\nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t)$. In contrast, it can employ both the unbiased and biased estimators $\widehat{G}_t$ for update. It is worth noting that $\widehat{g}_t$ can be both the first-order estimate (stochastic gradient) and the zeroth-order estimate (function value based estimator).

The update step of $\boldsymbol{\mu}$ and Σ is achieved by solving the following convex minimization problem.

$$\boldsymbol{m}^{t+1} = \arg\min_{\boldsymbol{m} \in \boldsymbol{\mathcal{M}}} \beta_t \left\langle \boldsymbol{m}, \widehat{v}_t \right\rangle + \mathrm{KL}\left(p_{\boldsymbol{m}} \| p_{\boldsymbol{m}^t} \right) \tag{3.45}$$

where $\boldsymbol{m} := \{\boldsymbol{m}_1, \boldsymbol{m}_2\} = \{\boldsymbol{\mu}, \Sigma + \boldsymbol{\mu}\boldsymbol{\mu}^\top\} \in \boldsymbol{\mathcal{M}}$, $\boldsymbol{\mathcal{M}}$ denotes a convex set, and $\widehat{v}_t = \{\widehat{g}_t - 2\widehat{G}_t\boldsymbol{\mu}_t, \widehat{G}_t\}$.

The optimum of problem (3.45) leads to closed-form update (3.46) and (3.47):

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + 2\beta_t\widehat{G}_t \tag{3.46}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t\Sigma_{t+1}\widehat{g}_t \tag{3.47}$$

General Stochastic Case: The convergence rate of Algorithm 4 is shown in Theorem 4.

Theorem 4. *Given a convex function $f(\boldsymbol{x})$, define $\bar{J}(\boldsymbol{\theta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})]$ for Gaussian distribution with parameter $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}}\} \in \boldsymbol{\Theta}$ and $\boldsymbol{\Theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}} \big| \ \boldsymbol{\mu} \in \mathcal{R}^d, \Sigma \in \mathcal{S}^+\}$. Suppose $\bar{J}(\boldsymbol{\theta})$ be γ-strongly convex. Let $\widehat{G}_t$ be positive semi-definite matrix such that $bI \preceq \widehat{G}_t \preceq \frac{\gamma}{2}I$. Suppose $\Sigma_1 \in \mathcal{S}^{++}$ and $\|\Sigma_1\| \leq \rho$, $\mathbb{E}\widehat{g}_t = \nabla_{\boldsymbol{\mu}=\boldsymbol{\mu}_t}\bar{J}$. Assume furthermore $\|\nabla_{\Sigma=\Sigma_t}\bar{J}\|_{tr} \leq B_1$ and $\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_1\|^2_{\Sigma_1^{-1}} \leq R$, $\mathbb{E}\|\widehat{g}_t\|_2^2 \leq \mathcal{B}$. Set $\beta_t = \beta$, then Algorithm 4 can achieve*

$$\frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}f(\boldsymbol{\mu}_t)\right] - f(\boldsymbol{\mu}^*) \leq \tag{3.48}$$

$$\frac{2bR + 2b\beta\rho(4B_1 + \beta\mathcal{B}) + 4B_1(1+\log T) + (1+\log T)\beta\mathcal{B}}{4\beta bT}$$

$$= \mathcal{O}\left(\frac{\log T}{T}\right) \tag{3.49}$$

Remark: Theorem 4 does not require the function $f(\boldsymbol{x})$ be differentiable. It holds for non-smooth function $f(\boldsymbol{x})$. Theorem 4 holds for convex function $f(\boldsymbol{x})$, as long as $\bar{J}(\boldsymbol{\theta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})]$ be γ-strongly convex. Particularly, when $f(\boldsymbol{x})$ is γ-strongly convex, we know $\bar{J}(\boldsymbol{\theta})$ is γ-strongly convex [49]. Thus, the assumption here is weaker than strongly convex assumption of $f(\boldsymbol{x})$. Moreover, Theorem 4 does not require the boundedness of the domain. It only requires the boundedness of the distance between the initialization point and an optimal point. Theorem 4 shows that the bound depends on the bound of $\mathbb{E}\|\widehat{g}_t\|_2^2$, which means that reducing variance of the gradient estimators can leads to a small regret bound.

Black-box Case: For black-box optimization, we can only access the function value instead of the gradient. We give an unbiased estimator of $\nabla_{\boldsymbol{\mu}}\bar{J}(\boldsymbol{\theta}_t)$ using function values as below

$$\widehat{g}_t = \Sigma_t^{-\frac{1}{2}}\boldsymbol{z}\left(f(\boldsymbol{\mu}_t + \Sigma_t^{\frac{1}{2}}\boldsymbol{z}) - f(\boldsymbol{\mu}_t)\right) \tag{3.50}$$

where $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$.

The estimator $\widehat{g}_t$ is unbiased, i.e., $\mathbb{E}[\widehat{g}_t] = \nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t)$. The proof of unbiasedness of the estimator $\widehat{g}_t$ is given in Lemma 7 in the Appendix. With this estimator, we give the convergence rate of Algorithm 4 for convex black-box optimization as in Theorem 5.

Theorem 5. *For a L-Lipschitz continuous convex black box function $f(\boldsymbol{x})$, define $\bar{J}(\boldsymbol{\theta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})]$ for Gaussian distribution with parameter $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}}\} \in \boldsymbol{\Theta}$ and $\boldsymbol{\Theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}} \big| \ \boldsymbol{\mu} \in \mathcal{R}^d, \Sigma \in \mathcal{S}^+\}$. Suppose $\bar{J}(\boldsymbol{\theta})$ be γ-strongly convex. Let $\widehat{G}_t$ be positive semi-definite matrix such that $b\boldsymbol{I} \preceq \widehat{G}_t \preceq \frac{\gamma}{2}\boldsymbol{I}$. Suppose $\Sigma_1 \in \mathcal{S}^{++}$ and $\|\Sigma_1\|_2 \leq \rho$, Assume furthermore $\|\nabla_{\Sigma=\Sigma_t} \bar{J}\|_{tr} \leq B_1$ and $\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_1\|^2_{\Sigma_1^{-1}} \leq R$, . Set $\beta_t = \beta$ and employ estimator $\widehat{g}_t$ in Eq.(3.50), then Algorithm 4 can achieve*

$$\frac{1}{T}\left[\sum_{t=1}^{T} \mathbb{E} f(\boldsymbol{\mu}_t)\right] - f(\boldsymbol{\mu}^*) \tag{3.51}$$

$$\leq \frac{bR + b\beta\rho(4B_1 + 2\beta L^2(d+4)^2)}{2\beta bT} \tag{3.52}$$

$$+ \frac{4B_1(1+\log T) + (1+\log T)\beta L^2(d+4)^2}{4\beta bT}$$

$$= \mathcal{O}\left(\frac{d^2 \log T}{T}\right) \tag{3.53}$$

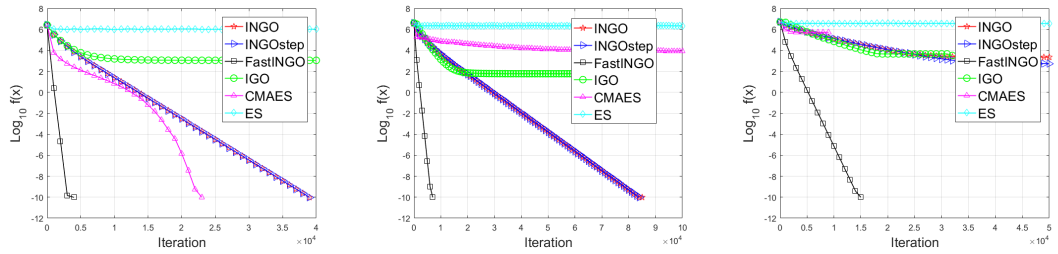
Remark: Theorem 5 holds for non-differentiable function $f(\boldsymbol{x})$. Thus, Theorem 5 can cover more interesting cases e.g. sparse black box optimization. In contrast, Balasubramanian et al. ([19]) require function $f(\boldsymbol{x})$ has Lipschitz continuous gradients.

Both Alg. 1 and Alg. 2 employ an unbiased gradient estimator $\widehat{g}$ for $\boldsymbol{\mu}$ update and biased estimator $\widehat{G}$ for variance Σ (σ^2) update. When further ensure $b\boldsymbol{I} \preceq \widehat{G}_t \preceq \frac{\gamma}{2}\boldsymbol{I}$, Theorem 5 holds for Alg. 1 and Alg. 2. Theorem 5 is derived for single sample per iteration. We can reduce the variance of estimators by constructing a set of structured samples that are conjugate of inverse covariance matrix in a batch, i.e., $\boldsymbol{z}_i \Sigma_t^{-1} \boldsymbol{z}_j = 0, i \neq j$. Particularly, when we use $\widehat{\Sigma}_t = \sigma_t \boldsymbol{I}$, sampling $N = d$ orthogonal samples [38] per iteration can lead to a convergence rate $\mathcal{O}\left(\frac{d \log T}{T}\right)$. For $N > d$ samples, we can use the method in [117] with a random rotation to reduce variance.

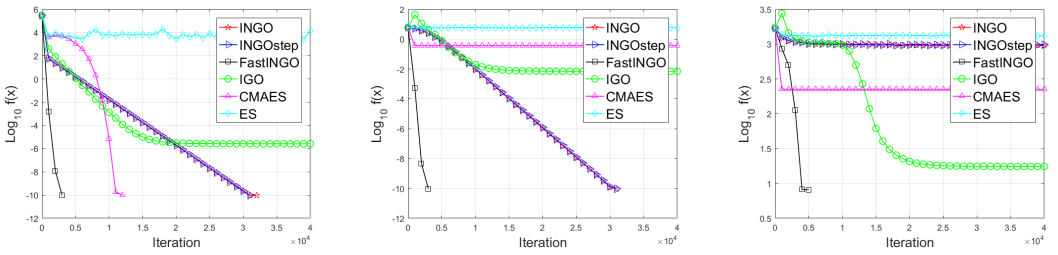
3.7 Empirical Evaluation

3.7.1 Evaluation on synthetic continuous test benchmarks

We evaluate the proposed INGO , INGOstep and Fast-INGO (diagonal case of INGO) by comparing with one of the state-of-the-art method CMA-ES [69] and IGO [135]



(a) Ellipsoid (b) ℓ_1-Ellipsoid (c) $\ell_{\frac{1}{2}}$-Ellipsoid



(d) Discus (e) Levy (f) Rastrigin10

Figure 3.1: Mean value of $f(\boldsymbol{x})$ in $\log_{10}$ scale over 20 independent runs for 100-dimensional problems.

with full covariance matrix update, and vanilla ES with antithetic gradient estimators [149] on several synthetic benchmark test problems. All the test problems are listed in Table 3.1.

Table 3.1: Test functions

name	function
Ellipsoid	$f(\boldsymbol{x}) := \sum_{i=1}^{d} 10^{\frac{6(i-1)}{d-1}} x_i^2$
Discus	$f(\boldsymbol{x}) := 10^6 x_1 + \sum_{i=2}^{d} x_i^2$
ℓ_1-Ellipsoid	$f(\boldsymbol{x}) := \sum_{i=1}^{d} 10^{\frac{6(i-1)}{d-1}} \lvert x_i \rvert$
$\ell_{\frac{1}{2}}$-Ellipsoid	$f(\boldsymbol{x}) := \sum_{i=1}^{d} 10^{\frac{6(i-1)}{d-1}} \lvert x_i \rvert^{\frac{1}{2}}$
Levy	$f(\boldsymbol{x}) := \sin^2(\pi w_1) + \sum_{i=1}^{d-1} (w_i - 1)^2 (1 + 10\sin^2(\pi w_i + 1)) + (w_d - 1)^2 (1 + \sin^2(2\pi w_d))$ where $w_i = 1 + (x_i - 1)/4,\ i \in \{1, ..., d\}$
Rastrigin10	$f(\boldsymbol{x}) := 10d + \sum_{i=1}^{d} (10^{\frac{i-1}{d-1}} x_i)^2 - 10\cos\left(2\pi 10^{\frac{i-1}{d-1}} x_i\right)$

Parameter Settings: For INGO, INGOstep and IGO, we use the same normalization transformation $h(f(\boldsymbol{x}_i) = \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{\widehat{\sigma}}$ and all same hyper-parameters to test the effect of implicit natural gradient. We set step size $\beta = 1/d$ for all of them. For FastINGO, we set step size $\beta = 1/\sqrt{d}$, where d is the dimension of the test problems. The number of samples per iteration is set to $N = 2\lfloor 3 + \lfloor 3 \times \ln d \rfloor / 2 \rfloor$ for all the methods,

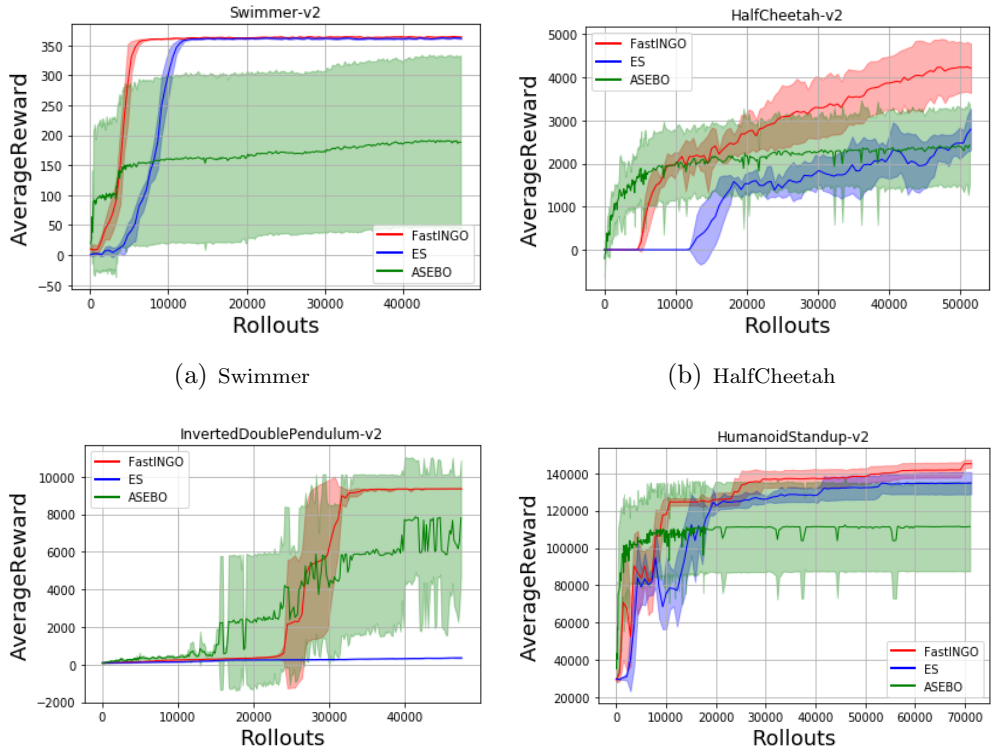
where $\lfloor \cdot \rfloor$ denotes the floor function. This setting ensures N to be an even number. We set $\boldsymbol{\sigma}_1 = 0.5 \times \mathbf{1}$ and sample $\boldsymbol{\mu}_1 \sim Uni[\mathbf{0}, \mathbf{1}]$ as the same initialization for all the methods, where $Uni[0, 1]$ denotes the uniform distribution in $[0, 1]$. For ES [149], we use the default step-size hyper-parameters.

The mean value of $f(\boldsymbol{x})$ over 20 independent runs for 100-dimensional problems are show in Figure 3.1. From Figure 3.1, we can see that INGO, INGOstep and Fast-INGO converge linearly in log scale. Fast-INGO can arrive 10^{-10} precision on five cases except the highly non-convex Rastrigin10 problem. Fast-INGO employs the separate structure of the problems, thus it obtains better performance than the other methods with full matrix update. It is worth to note that Fast-INGO is not rotation invariant compared with Full-INGO. The INGO and INGOstep (with full matrix update) can arrive 10^{-10} on four cases, while IGO with full matrix update can not achieve high precision. This shows that the update of inverse of covariance matrix is more stable. Moreover, CMA-ES converge linearly in log scale for the convex Ellipsoid problem but slower than Fast-INGO. In addition, CMAES converge slowly on the non-smooth ℓ_1-Ellipsoid and the non-convex $\ell_{\frac{1}{2}}$-Ellipsoid problem. Furthermore, CMAES fails on the non-convex Levy problem, while INGO, INGOstep and Fast-INGO obtain 10^{-10}. CMAES converges faster or achieves smaller value than ES. On the non-convex Rastrigin10 problem, all methods fail to obtain 10^{-10} precision. Fast-INGO obtains smaller value. The results on synthetic test problems show that methods employing second-order information converge faster than first-order method ES. And employing second-order information is important to obtain high optimization precision, i.e., 10^{-10}. Moreover, taking stochastic implicit natural gradient update can converge faster than IGO. The test functions are highly ill-conditioned and non-convex; the experimental results show that it is challenging for ES to optimize them well without adaptively update covariance and mean.

3.7.2 Evaluation on RL test problems

We further evaluate the proposed Fast-INGO by comparing AESBO [37] and ES with antithetic gradient estimators [149] on MuJoCo control problems: Swimmer, HalfCheetah, HumanoidStandup, InvertedDoublePendulum, in Open-AI Gym environments. CMA-ES is too slow due to the computation of eigendecomposition for high-dimensional problems.

We use one hidden layer feed-forward neural network with tanh activation function as policy architecture. The number of hidden unit is set to $h = 16$ for all problems. The goal is to find the parameters of this policy network to achieve large reward.



(a) Swimmer (b) HalfCheetah

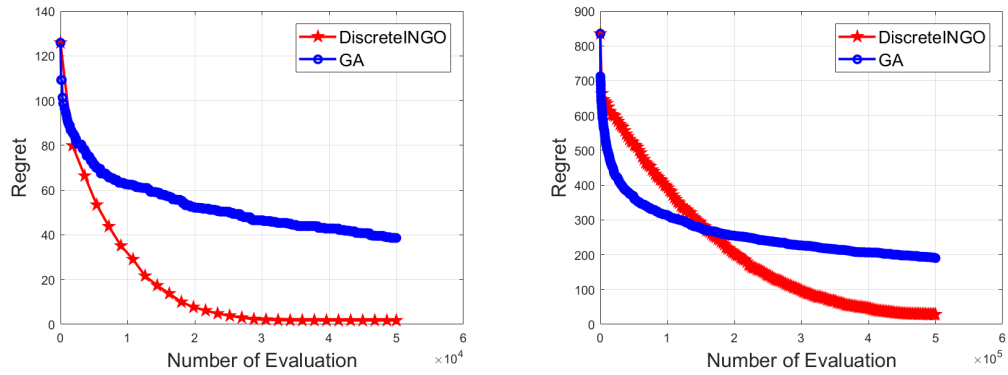
(c) InvertedDoublePendulum (d) HumanoidStandup

Figure 3.2: Average Reward over 5 independent runs on benchmark RL environments

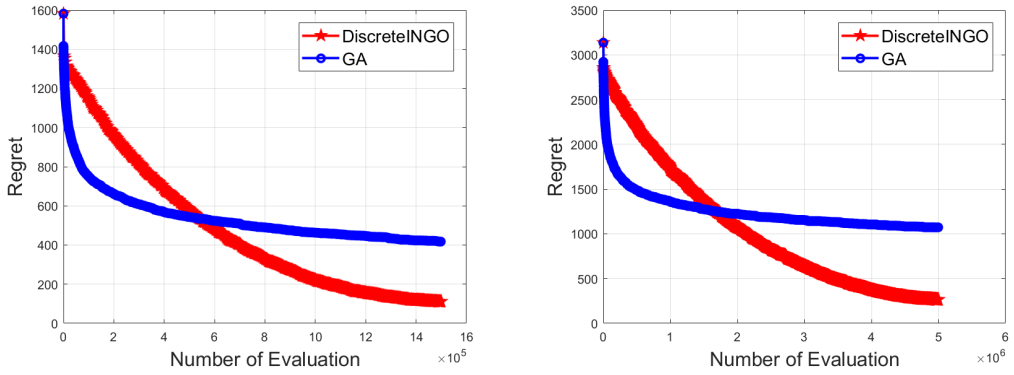
The same policy architecture is used for all the methods on all test problems. The number of samples per iteration is set to $N = 20+4\lfloor\lfloor 3\times\ln d\rfloor/2\rfloor$ for all the methods. For Fast-INGO, we set step-size $\beta = 0.3$. We set $\boldsymbol{\sigma}_1 = 0.1 \times \mathbf{1}$ and $\boldsymbol{\mu}_1 = \mathbf{0}$ as the initialization for both Fast-INGO and ES. For ES [149], we use the default step-size hyper-parameters. Five independent runs are performed. The experimental results are shown in Figure 3.2. We can observe that Fast-INGO increase AverageReward faster than ES on all four cases. This shows that the update using seconder order information in Fast-INGO can help accelerate convergence.

3.7.3 Evaluation on discrete test problems

We evaluate our discrete INGO by comparing with GA method on binary reconstruction benchmark problem, i.e., $f(\boldsymbol{x}) := \|\text{sign}(\boldsymbol{x} - 0.5) - \boldsymbol{w}\|_2^2 - \|\text{sign}(\boldsymbol{w}) - \boldsymbol{w}\|_2^2$ with $\boldsymbol{x} \in \{0,1\}^d$. We construct $\boldsymbol{w}$ by sampling from standard Gaussian. The dimension d of test problem is set to $\{100, 500, 1000, 2000\}$, respectively. For our discrete INGO, we set the stepsize $\beta = 1/d$. The number of samples per iteration is same as INGO, i.e., $N = 20 + 4\lfloor 3 + \lfloor 3 \times \ln d\rfloor/2\rfloor$.



(a) 100-dimensional problem (b) 500-dimensional problem



(c) 1000-dimensional problem (d) 2000-dimensional problem

Figure 3.3: Mean value of regret over 10 independent runs for different dimensional discrete optimization problems

The experimental resutls are shown in Fig. 3.3. We can observe that our discrete INGO achieves much smaller regret compared with GA. Our discrete INGO converges to near zero regret on 100-dimensional and 500-dimensional test problems, while GA decrease very slowly after a short initial greedy phase.

3.8 Summary

In this Chapter, I proposed a novel stochastic implicit natural gradient frameworks for black-box optimization. Under this framework, I presented algorithms for both continuous and discrete black-box optimization. Theoretically, I proved the $\mathcal{O}(\log T/T)$ convergence rate of our continuous algorithms with stochastic update for non-differentiable convex function under expectation γ-strongly convex assumption. I proved $\mathcal{O}(d^2 \log T/T)$ converge rate for black-box function under same assumptions above. For isometric Gaussian case, we proved the $\mathcal{O}(d \log T/T)$ converge rate when using d orthogonal

samples per iteration, which well supports parallel evaluation. Our method is very simple, and it contains less hyper-parameters than CMA-ES. Empirically, our methods obtain a competitive performance compared with CMA-ES. Moreover, our INGO and INGOstep with full matrix update can achieve high precision on Levy test problem and Ellipsoid problems, while IGO [135] with full matrix update can not. This shows the efficiency of our methods. On RL control problems, our algorithms increase average reward faster than ASEBO [37] and ES, which shows employing second order information can help accelerate convergence. Moreover, our discrete algorithm outperforms than GA on test functions.

Chapter 4

Batch Bayesian Optimization

4.1 Chapter Abstract

In this chapter, we investigate black-box optimization from the perspective of frequentist kernel methods. We propose a novel batch optimization algorithm, which jointly maximize the acquisition function and select points from a whole batch in a holistic way. Theoretically, we derive regret bounds for both the noise-free and perturbation settings irrespective of the choice of kernel. Moreover, we analyze the property of the adversarial regret that is required by a robust initialization for Bayesian Optimization (BO). We prove that the adversarial regret bounds decrease with the decrease of covering radius, which provides a criterion for generating a point set to minimize the bound. We then propose fast searching algorithms to generate a point set with a small covering radius for the robust initialization. Experimental results on both synthetic benchmark problems and real-world problems show the effectiveness of the proposed algorithms.

4.2 Problem Setup

Notations and Symbols: Let $\mathcal{H}_k$ denote a separable reproducing kernel Hilbert space associated with the kernel $k(\cdot,\cdot)$, and Let $\|\cdot\|_{\mathcal{H}_k}$ denote the RKHS norm in $\mathcal{H}_k$. $\|\cdot\|$ denotes the l_2 norm (Euclidean distance). Let $\mathcal{B}_k = \{f : f \in \mathcal{H}_k, \|f\|_{\mathcal{H}_k} \leq B\}$ denotes a bounded subset in the RKHS, and $\mathcal{X} \subset \mathbb{R}^d$ denote a compact set in $\mathbb{R}^d$. Symbol $[N]$ denotes the set $\{1, ..., N\}$. $\mathbb{N}$ and $\mathbb{P}$ denote the integer set and prime number set, respectively. Bold capital letters are used for matrices.

Let $f : \mathcal{X} \to \mathbb{R}$ be the unknown black-box function to be optimized, where

$\mathcal{X} \subset \mathbb{R}^d$ is a compact set. BO aims to find a maximum x^* of the function f, i.e.,

$$f(x^*) = \max_{x \in \mathcal{X}} f(x).$$

In sequential BO, a single point $x_t \in \mathcal{X}$ is selected to query an observation at round t. Batch BO is introduced in the literature for the benefits of parallel execution. Batch BO methods select a batch of points $X_n = \{x_{(n-1)L+1}, ..., x_{nL}\}$ simultaneously at round n, where L is the batch size. The batch BO is different from the sequential BO because the observation is delayed for batch BO during the batch selection phase. An additional challenge is introduced in batch BO since it needs to select a batch of points at one time, without knowing the latest information about the function f.

In BO, the effectiveness of a selection policy can be measured by the cumulative regret R_T and the simple regret (minimum regret) r_T over T steps. The cumulative regret R_T and simple regret r_T are defined as follows,

$$R_T = \sum_{t=1}^{T} \left(f(x^*) - f(x_t) \right), \tag{4.1}$$

$$r_T = f(x^*) - \max_{1 \le t \le T} f(x_t). \tag{4.2}$$

The regret bound introduced in numerous theoretical works is based on the maximum information gain defined as

$$\gamma_T = \max_{\mathbf{x_1}, ..., \mathbf{x_T}} \frac{1}{2} \log \det(\mathbf{I}_T + \sigma^{-2} \mathbf{K}_T), \tag{4.3}$$

where $\mathbf{K}_T = [k(\mathbf{x}_i, \mathbf{x}_j)]_{1 \le i,j \le T}$ denotes the kernel matrix, and $\mathbf{x}_1, \cdots, \mathbf{x}_T \in \mathcal{X}$ denotes the T data points in the input domain.

The bounds of γ_T for commonly used kernels are studied in [160]. Specifically, [160] state that $\gamma_T = \mathcal{O}(d \log T)$ for the linear kernel, $\gamma_T = \mathcal{O}((\log T)^{d+1})$ for the squared exponential kernel and $\gamma_T = \mathcal{O}(T^{\alpha}(\log T))$ for the Matérn kernels with $\nu > 1$, where $\alpha = \frac{d(d+1)}{2\nu + d(d+1)} \le 1$. We employ the term γ_T to build the regret bounds of our algorithms.

In this work, we consider two settings: noise-free setting and perturbation setting:

Noise-Free Setting: We assume the underlying function f belongs to an RKHS associated with kernel $k(\cdot, \cdot)$, i.e., $f \in \mathcal{H}_k$, with $\|f\|_{\mathcal{H}_k} < \infty$. In the noise-free setting, we can directly observe $f(x), x \in \mathcal{X}$ without noise perturbation.

Perturbation Setting: In the perturbation setting, we cannot observe the function evaluation $f(x)$ directly. Instead, we observe $y = h(x) = f(x) + g(x)$, where $g(x)$ is an unknown perturbation function.

Define $k^{\sigma}(x,y) := k(x,y) + \sigma^2\delta(x,y)$ for $x, y \in \mathcal{X}$, where $\delta(x,y) = \begin{cases} 1 & x = y \\ 0 & x \neq y \end{cases}$ and $\sigma \geq 0$. We assume $f \in \mathcal{H}_k$, $g \in \mathcal{H}_{\sigma^2\delta}$ with $\|f\|_{\mathcal{H}_k} < \infty$ and $\|g\|_{\mathcal{H}_{\sigma^2\delta}} < \infty$, respectively. Therefore, we know $h \in \mathcal{H}_{k^{\sigma}}$ and $\|h\|_{\mathcal{H}_{k^{\sigma}}} < \infty$. The same point is assumed never selected twice, this can be ensured by the deterministic selection rule.

4.3 BO in Noise-Free Setting

In this section, we will first present algorithms and theoretical analysis in the sequential case. We then discuss our batch selection method. All detailed proofs are included in the supplementary material.

4.3.1 Sequential Selection in Noise Free Setting

Define $m_t(x)$ and $\sigma_t(x)$ as follows:

$$m_t(x) = \mathbf{k}_t(x)^T \mathbf{K}_t^{-1} \mathbf{f}_t \tag{4.4}$$

$$\sigma_t^2(x) = k(x,x) - \mathbf{k}_t(x)^T \mathbf{K}_t^{-1} \mathbf{k}_t(x), \tag{4.5}$$

where $\mathbf{k}_t(x) = [k(x,x_1), ..., k(x,x_t)]^T$ and the kernel matrix $\mathbf{K}_t = [k(x_i,x_j)]_{1 \leq i,j \leq t}$. These terms are closely related to the posterior mean and variance functions of GP with zero noise. We use them in the deterministic setting. A detailed review of the relationships between GP methods and kernel methods can be found in [87].

The sequential optimization method in the noise-free setting is described in Algorithm 5. It has a similar form to GP-UCB [160], except that it employs a constant weight of the term $\sigma_{t-1}(x)$ to balance exploration and exploitation. In contrast, GP-UCB uses a $\mathcal{O}(\log(t))$ increasing weight. In practice, a constant weight is preferred in the scenarios where an aggressive selection manner is needed. For example, only a small number of evaluations can be done in the hyperparameter tuning in RL algorithms due to limited resources. The regret bounds of Algorithm 5 are given in Theorem 6.

Theorem 6. *Suppose $f \in \mathcal{H}_k$ associated with $k(x,x) \leq 1$ and $\|f\|_{\mathcal{H}_k} < \infty$. Let $C_1 = \frac{8}{\log(1+\sigma^{-2})}$. **Algorithm 5** achieves a cumulative regret bound and a simple regret bound given as follows:*

$$R_T \leq \|f\|_{\mathcal{H}_k} \sqrt{T C_1 \gamma_T} \tag{4.6}$$

$$r_T \leq \|f\|_{\mathcal{H}_k} \sqrt{\frac{C_1 \gamma_T}{T}}. \tag{4.7}$$

where $0 < c < +\infty$.

Algorithm 5 Sequential Noise-free Algorithm

for $t = 1$ **to** T **do**
  Obtain $m_{t-1}(\cdot)$ and $\sigma^2_{t-1}(\cdot)$ via equations (4.4) and (4.5).
  Choose $x_t = \arg\max_{x \in \mathcal{X}} m_{t-1}(x) + \|f\|_{\mathcal{H}_k} \sigma_{t-1}(x)$.
  Query the observation $f(x_t)$ at location x_t.
end for

Remark: We can achieve concrete bounds w.r.t T by replacing γ_T with the specific bound for the corresponding kernel. For example, for SE kernels, we can obtain that $R_T = \mathcal{O}(\sqrt{T}(\log T)^{d+1})$ and $r_T = \mathcal{O}(\frac{(\log T)^{d+1}}{\sqrt{T}})$, respectively. [32] presents bounds for Matérn type kernels. The bound in Theorem 6 is tighter than Bull's bound of pure EI (Theorem 4 in [32]) when the smoothness parameter of the Matérn kernel $\nu > \frac{d(d+1)}{d-2} = \mathcal{O}(d)$. This is no better than the bound of mixed strategies (Theorem 5) in Bull's work. Nevertheless, the bound in Theorem 6 makes fewer assumptions about the kernels, and covers more general results (kernels) compared with Bull's work.

4.3.2 Batch Selection in Noise-Free Setting

Let N and L be the number of batches and the batch size, respectively. Without loss of generality, we assume $T = NL$. Let $X_n = \{x_{(n-1)L+1}, ..., x_{nL}\}$ and $\overline{X}_n = \{X_1, ..., X_n\} = \{x_1, ..., x_{nL}\}$ be the n^{th} batch of points and all the n batches of points, respectively. The covariance function of $X \in \mathbb{R}^{d \times L}$ for the noise free case is given as follows:

$$\text{cov}_n(X, X) = \mathbf{K}(X, X) - \mathbf{K}(\overline{X}_n, X)^T \mathbf{K}(\overline{X}_n, \overline{X}_n)^{-1} \mathbf{K}(\overline{X}_n, X) \tag{4.8}$$

where $\mathbf{K}(\mathbf{X}, \mathbf{X})$ is the $L \times L$ kernel matrix, $\mathbf{K}(\overline{X}_n, X)$ denotes the $nL \times L$ kernel matrix between $\overline{X}_n$ and X. When $n = 0$, $\text{cov}_0(X, X) = \mathbf{K}(X, X)$ is the prior kernel matrix. We assume that the kernel matrix is invertible in the noise-free setting.

The proposed batch optimization algorithm is presented in Algorithm 8. It employs the mean prediction value of a batch together with a term of covariance to balance the exploration/exploitation trade-off. The covariance term in Algorithm 8 penalizes the batch with over-correlated points. Intuitively, for SE kernels and Matérn kernels, it penalizes the batch with points that are too close to each other (w.r.t Euclidean distance). As a result, it encourages the points in a batch to spread out for better exploration. The regret bounds of our batch optimization method are summarized in Theorem 7.

Algorithm 6 Batch Noise-free Algorithm

for $n = 1$ **to** N **do**

Obtain $m_{(n-1)L}(\cdot)$ and $\mathrm{cov}_{n-1}(\cdot)$ via equations (4.4) and (4.8), respectively.

Choose $X_n = \arg\max_{X \subset \mathcal{X}} \frac{1}{L} \sum_{i=1}^{L} m_{(n-1)L}(X_{\cdot,i}) + \|h\|_{\mathcal{H}_k} \left(2\sqrt{\frac{\mathrm{tr}(\mathrm{cov}_{n-1}(X,X))}{L}} - \sqrt{\frac{\mathbf{1}^T \mathrm{cov}_{n-1}(X,X)\mathbf{1}}{L^2}} \right)$.

Query the batch observations $\{f(x_{(n-1)L+1}), ..., f(x_{nL})\}$ at locations X_n.

end for

Theorem 7. *Suppose $f \in \mathcal{H}_k$ associated with $k(x,x) \leq 1$ and $\|f\|_{\mathcal{H}_k} < \infty$. Let $T = NL$, $\beta = \max_{n \in \{1,...,N\}} \|\widehat{\mathrm{cov}}_{n-1}(X_n, X_n)\|_2$ and $C_2 = \frac{8\beta}{\log(1+\beta\sigma^{-2})}$. **Algorithm 8** with batch size L achieves a cumulative regret bound and a simple regret bound given by equations (4.9) and (4.10), respectively:*

$$R_T \leq \|f\|_{\mathcal{H}_k} \sqrt{T C_2 \gamma_T} \tag{4.9}$$

$$r_T \leq \|f\|_{\mathcal{H}_k} \sqrt{\frac{C_2 \gamma_T}{T}}. \tag{4.10}$$

Remark: (1) A large β leads to a large bound, while a small β attains a small bound. Algorithm 2 punishes the correlated points and encourages the uncorrelated points in a batch, which can attain a small β in general. (2) A trivial bound of β is $\beta \leq L$.

To prove Theorem 7, the following Lemma is proposed. The detailed proof can be found in the Appendix.

Lemma 1. *Suppose $f \in \mathcal{H}_k$ associated with kernel $k(x,x)$ and $\|f\|_{\mathcal{H}_k} < \infty$, then we have $\left(\sum_{i=1}^{L} m_t(\widehat{x}_i) - \sum_{i=1}^{L} f(\widehat{x}_i) \right)^2 \leq \|f\|^2_{\mathcal{H}_k} (\mathbf{1}^T \mathbf{A} \mathbf{1})$, where $\mathbf{A}$ denotes the kernel covariance matrix with $\mathbf{A}_{ij} = k(\widehat{x}_i, \widehat{x}_j) - \mathbf{k}_t(\widehat{x}_i)^T \mathbf{K}_t^{-} \mathbf{k}_t(\widehat{x}_j)$.*

Remark: Lemma 1 provides a tighter bound for the deviation of the summation of a batch than directly applying the bound for a single point L times.

The intuition of the batch selection scheme: the selection acquisition function consists of two parts, the mean of prediction in a batch and the variance/covariance terms in a batch. The mean of prediction provides an estimation of the black-box function. The variance/covariance terms encourage exploration in a batch. These two parts balance between exploits of current estimation for optimization and maintain a diversity exploration to gain information.

Algorithm 7 Sequential Optimization with Perturbation

for $t = 1$ **to** T **do**
  Obtain $\widehat{m}_{t-1}(\cdot)$ and $\widehat{\sigma}^2_{t-1}(\cdot)$ via equation (4.11) and (4.12).
  Choose $x_t = \arg\max_{x\in\mathcal{X}} \widehat{m}_{t-1}(x) + \|h\|_{\mathcal{H}_{k^\sigma}} \widehat{\sigma}_{t-1}(x)$
  Query the observation $y_t = h(x_t)$ at location x_t.
end for

4.4 BO in Perturbation Setting

In the perturbation setting, we cannot observe the function evaluation $f(x)$ directly. Instead, we observe $y = h(x) = f(x) + g(x)$, where $g(x)$ is an unknown perturbation function. We will discuss the sequential selection and batch selection methods in the following sections, respectively.

4.4.1 Sequential Selection in Perturbation Setting

Define $\widehat{m}_t(x)$ and $\widehat{\sigma}_t(x)$ as follows:

$$\widehat{m}_t(x) = \mathbf{k}_t(x)^T(\mathbf{K}_t + \sigma^2 I)^{-1}\mathbf{y}_t \tag{4.11}$$

$$\widehat{\sigma}^2_t(x) = k(x,x) - \mathbf{k}_t(x)^T(\mathbf{K}_t + \sigma^2 I)^{-1}\mathbf{k}_t(x), \tag{4.12}$$

where $\mathbf{k}_t(x) = [k(x,x_1), ..., k(x,x_t)]^T$ and the kernel matrix $\mathbf{K}_t = [k(x_i,x_j)]_{1\le i,j\le t}$.

The sequential selection method is presented in Algorithm 7. It has a similar formula to Algorithm 5; while Algorithm 7 employs a regularization $\sigma^2 I$ to handle the uncertainty of the perturbation. The regret bounds of Algorithm 7 are summarized in Theorem 8.

Theorem 8. *Define* $k^\sigma(x,y) := k(x,y) + \sigma^2\delta(x,y) \le B$, *where* $\delta(x,y) = \begin{cases} 1 & x = y \\ 0 & x \ne y \end{cases}$ *and* $\sigma \ge 0$. *Suppose* $f \in \mathcal{H}_k$, $g \in \mathcal{H}_{\sigma^2\delta}$ *associated with kernel* k *and kernel* $\sigma^2\delta$ *with* $\|f\|_{\mathcal{H}_k} < \infty$ *and* $\|g\|_{\mathcal{H}_{\sigma^2\delta}} < \infty$, *respectively. Let* $C_3 = \frac{8B}{\log(1+B\sigma^{-2})}$. ***Algorithm 7*** *achieves a cumulative regret bound and a simple regret bound given by equations (4.13) and (4.14), respectively.*

$$R_T \le \|h\|_{\mathcal{H}_{k^\sigma}}\sqrt{TC_3\gamma_T} + 2T\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{4.13}$$

$$r_T \le \|h\|_{\mathcal{H}_{k^\sigma}}\sqrt{\frac{C_3\gamma_T}{T}} + 2\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{4.14}$$

Remark: In the perturbation setting, the unknown perturbation function g results in some unavoidable dependence on σ in the regret bound compared with GP-UCB [160]. Note that the bounds in [160] are probabilistic bounds. There is always

a positive probability that the bounds in [160] fail. In contrast, the bounds in Theorem 8 are deterministic.

Corollary 1. *Suppose $h = f \in \mathcal{H}_k$ associated with $k(x,y) \leq 1$ and $\|f\|_{\mathcal{H}_k} < \infty$. Let $C_1 = \frac{8}{\log(1+\sigma^{-2})}$. **Algorithm 7** achieves a cumulative regret bound and a simple regret bound given by equations (4.15) and (4.16), respectively:*

$$R_T \leq \|f\|_{\mathcal{H}_k}\sqrt{TC_1\gamma_T} + 2T\|f\|_{\mathcal{H}_k}\sigma \tag{4.15}$$

$$r_T \leq \|f\|_{\mathcal{H}_k}\sqrt{\frac{C_1\gamma_T}{T}} + 2\|f\|_{\mathcal{H}_k}\sigma. \tag{4.16}$$

Proof. Setting $g = 0$ and $B = 1$ in Theorem 8, we can achieve the results. □

Remark: In practice, a small constant $\sigma^2 I$ is added to the kernel matrix to avoid numeric problems in the noise-free setting. Corollary 1 shows that the small constant results in an additional biased term in the regret bound. Theorem 6 employs (4.4) and (4.5) for updating, while Corollary 1 presents the regret bound for the practical updating by (4.11) and (4.12).

4.4.2 Batch Selection in Perturbation Setting

The covariance kernel function of $X \in \mathbb{R}^{d\times L}$ for the perturbation setting is defined as equation (4.17),

$$\widehat{\mathrm{cov}}_n(X,X) = \mathbf{K}(X,X) - \mathbf{K}(\overline{X}_n, X)^T\left(\sigma^2 I + \mathbf{K}(\overline{X}_n, \overline{X}_n)\right)^{-1}\mathbf{K}(\overline{X}_n, X), \tag{4.17}$$

where $\mathbf{K}(\mathbf{X},\mathbf{X})$ is the $L \times L$ kernel matrix, and $\mathbf{K}(\overline{X}_n, X)$ denotes the $nL \times L$ kernel matrix between $\overline{X}_n$ and X. The batch optimization method for the perturbation setting is presented in Algorithm 8. The regret bounds of Algorithm 8 are summarized in Theorem 9.

Theorem 9. *Define $k^\sigma(x,y) := k(x,y) + \sigma^2\delta(x,y) \leq B$, where $\delta(x,y) = \begin{cases} 1 & x = y \\ 0 & x \neq y \end{cases}$ and $\sigma \geq 0$. Suppose $f \in \mathcal{H}_k$ and $g \in \mathcal{H}_{\sigma^2\delta}$ associated with kernel k and kernel $\sigma^2\delta$ with $\|f\|_{\mathcal{H}_k} < \infty$ and $\|g\|_{\mathcal{H}_{\sigma^2\delta}} < \infty$, respectively. Let $T = NL$, $\beta = \max_{n\in\{1,\ldots,N\}}\|\widehat{\mathrm{cov}}_{n-1}(X_n, X_n)\|_2$ and $C_4 = \frac{8\beta}{\log(1+\beta\sigma^{-2})}$. **Algorithm 8** with batch size L achieves a cumulative regret bound and a simple regret bound given by equations (4.18) and (4.19), respectively:*

$$R_T \leq \|h\|_{\mathcal{H}_{k^\sigma}}\sqrt{TC_4\gamma_T} + 2T\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{4.18}$$

$$r_T \leq \|h\|_{\mathcal{H}_{k^\sigma}}\sqrt{\frac{C_4\gamma_T}{T}} + 2\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma. \tag{4.19}$$

Remark: When the batch size is one, the regret bounds reduce to the sequential case.

Algorithm 8 Batch Optimization with Perturbation

for $n = 1$ **to** N **do**

Obtain $\widehat{m}_{(n-1)L}(\cdot)$ and $\widehat{\mathrm{cov}}_{n-1}(\cdot)$ via equation (4.11) and (4.17) respectively.

Choose $X_n = \arg\max_{X \subset \mathcal{X}} \frac{1}{L} \sum_{i=1}^{L} \widehat{m}_{(n-1)L}(X_{\cdot,i}) + \|h\|_{\mathcal{H}_{k^\sigma}} \left(2\sqrt{\frac{\mathrm{tr}(\widehat{\mathrm{cov}}_{n-1}(X,X))}{L}} - \sqrt{\frac{\mathbf{1}^T \widehat{\mathrm{cov}}_{n-1}(X,X)\mathbf{1}}{L^2}} \right)$.

Query the batch observations $\{h(x_{(n-1)L+1}), ..., h(x_{nL})\}$ at locations $X_n = \{x_{(n-1)L+1}, ..., x_{nL}\}$.

end for

4.5 Robust Initialization for BO

In practice, the initialization phase of BO is important. In this section, we will discuss how to achieve robust initialization by analyzing regret in the adversarial setting. We will first show that algorithms that attain a small covering radius (fill distance) can achieve small adversarial regret bounds. Based on this insight, we provide a robust initialization to BO.

Let $f_t : \mathcal{X} \to \mathbb{R}$, $t \in [T]$ be the black-box function to be optimized at round t. Let $f_t(x_t^*) = \max_{x \in \mathcal{X}} f_t(x)$ with $f_t \in \mathcal{B}_k$. The simple adversarial regret $\widetilde{r}_T$ is defined as:

$$\widetilde{r}_T = \min_{t \in [T]} \sup_{\substack{f_t \in \mathcal{B}_k, \forall i \in [t-1], \\ f_t(x_i) = f_i(x_i)}} \{f_t(x_t^*) - f_t(x_t)\}, \tag{4.20}$$

where the constraints ensure that each f_t has the same observation values as the history at previous query points $X_{t-1} = \{x_1, ..., x_{t-1}\}$. This can be viewed as an adversarial game. During each round t, the opponent chooses a function f_t from a candidate set, and we then choose a query x_t to achieve a small regret. A robust initialization setting can be viewed as the batch of points that can achieve a low simple adversarial regret irrespective of the access order.

Define covering radius (fill distance [87]) and packing radius (separate distance [87]) of a set of points $X = \{x_1, ..., x_T\}$ as follows:

$$h_X = \sup_{x \in \mathcal{X}} \min_{x_t \in X} \|x - x_t\| \tag{4.21}$$

$$\rho_X = \frac{1}{2} \min_{\substack{x_i, x_j \in X, \\ x_i \neq x_j}} \|x_i - x_j\|. \tag{4.22}$$

Our method for robust initialization is presented in Algorithm 9, which constructs an initialization set X_{T-1} by minimizing the covering radius. We present one such method in Algorithm 10 in the next section. The initialization set X_{T-1} can be evaluated in a batch manner, which is able to benefit from parallel evaluation. The regret bounds of Algorithm 9 are summarized in Theorem 10 and Theorem 11.

Theorem 10. *Define* $\mathcal{B}_k = \{f : f \in \mathcal{H}_k, \|f\|_{\mathcal{H}_k} \leq B\}$ *associated with* $k(x,x)$ *for* $x \in \mathcal{X} \subset \mathbb{R}^d$. *Suppose* $f \in \mathcal{B}_k$ *and* $\mathcal{H}_k$ *is norm-equivalent to the Sobolev space of order* s. *Then there exits a constant* $C > 0$, *such that the query point set generated by* ***Algorithm 5*** *with a sufficiently small covering radius (fill distance)* h_X *achieves a regret bound given by equation (4.23):*

$$\widetilde{r}_T \leq BCh_X^{s-d/2}. \tag{4.23}$$

Remark: The regret bound decreases as the covering radius becomes smaller. This means that a query set with a small covering radius can guarantee a small regret. [32] gives bounds of fixed points set for Matérn kernels (Theorem 1). However, it does not link to the covering radius. The bound in Theorem 10 directly links to the covering radius, which provides a criterion for generating points to achieve small bounds.

Theorem 11. *Define* $\mathcal{B}_k = \{f : f \in \mathcal{H}_k, \|f\|_{\mathcal{H}_k} \leq B\}$ *associated with square-exponential* $k(x,x)$ *on unit cube* $\mathcal{X} \subset \mathbb{R}^d$. *Suppose* $f \in \mathcal{B}_k$. *Then there exits a constant* $c > 0$, *such that the query point set generated by* ***Algorithm 5*** *with a sufficiently small covering radius (fill distance)* h_X *achieves a regret bound given by equation (4.24):*

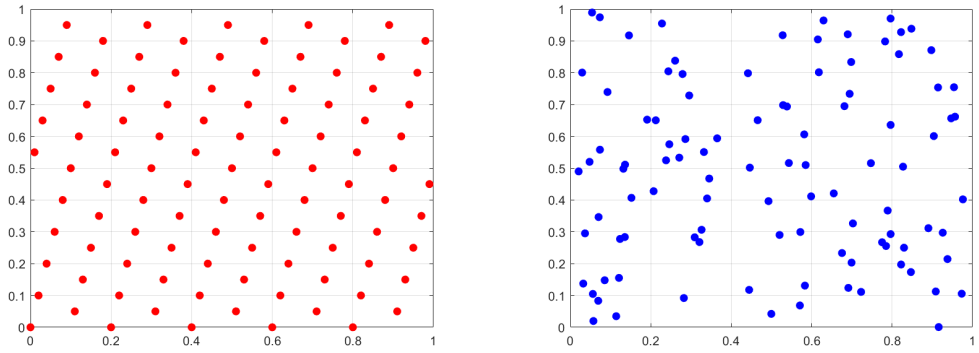
$$\widetilde{r}_T \leq B\exp(c\log(h_X)/(2\sqrt{h_X})). \tag{4.24}$$

Remark: Theorem 11 presents a regret bound for the SE kernel. It attains higher rate w.r.t covering radius h_X compared with Theorem 10, because functions in RKHS with SE kernel are more smooth than functions in Sobolev space.

We analyze the regret under a more adversarial setting. This relates to a more robust requirement. The regret bounds under a fully adversarial setting when little information is known are summarized in Theorem 12.

Algorithm 9 Greedy Batch Optimization

Construct Candidate set X_{T-1} with $T-1$ points by minimizing the fill distance (e.g.Algorithm 10).
Query the observations at X_{T-1}.
Obtain $m_{T-1}(\cdot)$ and $\sigma^2_{T-1}(\cdot)$ via equation (4.4) and (4.5).
Choose $x_T = \arg\max_{x\in\mathcal{X}} m_{T-1}(x) + B\sigma_{T-1}(x)$
Query the observation $y_T = f(x_T)$ at location x_T.



(a) 100 lattice points (b) 100 random points

Figure 4.1: Lattice Points and Random Points on $[0,1]^2$

Theorem 12. *Define* $\mathcal{B}_k = \{f : f \in \mathcal{H}_k, \|f\|_{\mathcal{H}_k} \leq B\}$ *associated with a shift invariant kernel* $k(x,y) = \Phi(\|x-y\|) \leq 1$ *that decreases w.r.t* $\|x-y\|$. *Suppose* $\exists x^*$ *such that* $f_t(x^*) = \max_{x\in\mathcal{X}} f_t(x)$ *with* $f_t \in \mathcal{B}_k$ *for* $t \in [T]$. *Then the query point set* $X = \{x_1, ..., x_T\}$ *generated by* ***Algorithm 9*** *with covering radius (fill distance)* h_X *achieves a regret bound as*

$$\bar{r}_T = \min_{t\in\{1,\dots,T\}} \sup_{f_t\in\mathcal{B}_k} \{f_t(x^*) - f_t(x_t)\} \leq B\sqrt{2-2\Phi(h_X)}.$$

Remark: Theorem 9 gives a fully adversarial bound. Namely, the opponent can choose functions from $\mathcal{B}_k$ without the same history. The regret bound decreases with the decrease of the covering radius (fill distance). The assumption requires each f_t to have the x^* as one of its maximum. Particularly, it is satisfied when $f_1 = \cdots = f_T = f$.

Corollary 2. *Define* $\mathcal{B}_k = \{f : f \in \mathcal{H}_k, \|f\|_{\mathcal{H}_k} \leq B\}$ *associated with squared exponential kernel. Suppose* $\exists x^*$ *such that* $f_t(x^*) = \max_{x\in\mathcal{X}} f_t(x)$ *with* $f_t \in \mathcal{B}_k$ *for* $t \in [T]$. *Then the query point set* $X = \{x_1, ..., x_T\}$ *generated by* ***Algorithm 9*** *with covering radius (fill distance)* h_X *achieves a regret bound as*

$$\bar{r}_T = \min_{t\in\{1,\dots,T\}} \sup_{f_t\in\mathcal{B}_k} \{f_t(x^*) - f_t(x_t)\} \leq \mathcal{O}(h_X). \tag{4.25}$$

Remark: For a regular grid, $h_X = \mathcal{O}(T^{-\frac{1}{d}})$ [172], we then achieve $\bar{r}_T = \mathcal{O}(T^{-\frac{1}{d}})$. Computer search can find a point set with a smaller covering radius than that of a regular grid.

All the adversarial regret bounds discussed above decrease with the decrease of the covering radius. Thus, the point set generated by Algorithm 9 with a small covering radius can serve as a good robust initialization for BO.

Algorithm 10 Rank-1 Lattice Construction

Input: Number of primes M, dimension d, number of lattice points N
Output: Lattice points X^*, base vector $\mathbf{b}^*$
Set $p_0 = 2 \times d + 1$, initialize $\rho^* = -1$.
Construct set $U := \{p | p \in \mathbb{P}, p \geq p_0\}$ containing M primes.
for each $p \in U$ **do**
  for $i = 0$ **to** $p - 1$ **do**
    Set $\mathbf{g} = \text{mod}(\mathbf{q} + i, p)$, where $\mathbf{q} \in \mathbb{R}^{\mathbf{d-1}}$ and $\mathbf{q}_j = j$.
    Set $\mathbf{g} = \text{round}(N \times \text{mod}(|2cos(\frac{2\pi \mathbf{g}}{p})|, 1))$.
    Set $\mathbf{b}$ as $[1, \mathbf{g}]$ by concatenating vector 1 and $\mathbf{g}$.
    Generate lattice X given base vector $\mathbf{b}$ as Eq.(4.26).
    Calculate the packing radius (separate distance) ρ_X of X as Eq.(5.8).
    if $\rho_X > \rho^*$ **then**
      Set $\mathbf{b}^* = \mathbf{b}$ and $\rho^* = \rho_X$.
    end if
  end for
end for
Generate lattice X^* given base vector $\mathbf{b}^*$ as Eq.(4.26).

4.6 Fast Rank-1 Lattice Construction

In this section, we describe the procedure of generating a query points set that has a small covering radius (fill distance). Since minimizing the covering radius of the lattice is equivalent to maximizing the packing radius (separate distance) [89], we generate the query points set through maximizing the packing radius (separate distance) of the rank-1 lattice. An illustration of the rank-1 lattice constructed by Algorithm 10 is given in Fig. 4.1

4.6.1 The rank-1 lattice construction given a base vector

Rank-1 lattice is widely used in the Quasi-Monte Carlo (QMC) literature for integral approximation [89, 96]. The lattice points of the rank-1 lattice in $[0, 1]^d$ are generated by a base vector. Given an integer base vector $\mathbf{b} \in \mathbb{N}^d$, a lattice set X that consists of N points in $[0, 1]^d$ is constructed as

$$X := \{\mathbf{x}_i := \text{mod}(i \times \mathbf{b}, N)/N | i \in \{0, ..., N-1\}\}, \tag{4.26}$$

where $\text{mod}(a, b)$ denotes the component-wise modular function, i.e., $a\%b$. We use $\text{mod}(a, 1)$ to denote the fractional part of number a in this work.

Algorithm 11 Rank-1 Lattice Construction with Successive Coordinate Search (SCS)

Input: Number of primes M, dimension d, number of lattice points N, number of iteration of SCS search subroutine T.
Output: Lattice points X^*, base vector $\mathbf{b}^*$
Set $p_0 = 2 \times d + 1$, initialize $\rho^* = -1$.
Construct set $U := \{p | p \in \mathbb{P}, p \geq p_0\}$ containing M primes.
for each $p \in U$ **do**
  for $i = 0$ **to** $p - 1$ **do**
    Set $\mathbf{g} = \text{mod}(\mathbf{q} + i, p)$, where $\mathbf{q} \in \mathbb{R}^{\mathbf{d-1}}$ and $\mathbf{q}_j = j$.
    Set $\mathbf{g} = \text{round}(N \times \text{mod}(|2cos(\frac{2\pi \mathbf{g}}{p})|, 1))$.
    Set $\mathbf{b}$ as $[1, \mathbf{g}]$ by concatenating vector 1 and $\mathbf{g}$.
    Perform SCS search [1, 117] with $\mathbf{b}$ as the initialization base vector to get a better base $\widehat{\mathbf{b}}$ and ρ_X.
    if $\rho_X > \rho^*$ **then**
      Set $\mathbf{b}^* = \widehat{\mathbf{b}}$ and $\rho^* = \rho_X$.
    end if
  end for
end for
Generate lattice X^* given base vector $\mathbf{b}^*$ as Eq.(4.26).

4.6.2 The separate distance of a rank-1 lattice

Denote the toroidal distance [64] between two lattice points $\mathbf{y} \in [0, 1]^d$ and $\mathbf{z} \in [0, 1]^d$ as:

$$\|\mathbf{y} - \mathbf{z}\|_T := \sqrt{\sum_{i=1}^{d} (\min(|y_i - z_i|, 1 - |y_i - z_i|))^2}. \tag{4.27}$$

Because the difference (subtraction) between two lattice points is still a lattice point, and a rank-1 lattice has a period of 1, the packing radius (separate distance) ρ_X of a rank-1 lattice with set X in $[0, 1]^d$ can be calculated as

$$\rho_X = \min_{\mathbf{x} \in X \setminus \mathbf{0}} \frac{1}{2} \|\mathbf{x}\|_T, \tag{4.28}$$

where $\|\mathbf{x}\|_T$ can be seen as the toroidal distance between $\mathbf{x}$ and $\mathbf{0}$. This formulation calculates the packing radius (separate distance) with a time complexity of $\mathcal{O}(Nd)$ rather than $\mathcal{O}(N^2 d)$ in pairwise computation.

Table 4.1: Minimum distance ($2\rho_X$) of 1,000 lattice points in $[0,1]^d$ for $d = 10$, $d = 20$, $d = 30$, $d = 40$ and $d = 50$.

	$d = 10$	$d = 20$	$d = 30$	$d = 40$	$d = 50$
Algorithm 10	0.59632	1.0051	1.3031	1.5482	1.7571
Korobov	0.56639	0.90139	1.0695	1.2748	1.3987
SCS	0.60224	1.0000	1.2247	1.4142	1.5811
Algorithm 11	**0.62738**	**1.0472**	**1.3620**	**1.6175**	**1.8401**

Table 4.2: Minimum distance ($2\rho_X$) of 2,000 lattice points in $[0,1]^d$ for $d = 10$, $d = 20$, $d = 30$, $d = 40$ and $d = 50$.

	$d = 10$	$d = 20$	$d = 30$	$d = 40$	$d = 50$
Algorithm 10	0.54658	0.95561	1.2595	1.4996	1.7097
Korobov	0.51536	0.80039	0.96096	1.1319	1.2506
SCS	0.57112	0.98420	1.2247	1.4142	1.5811
Algorithm 11	**0.58782**	**1.0144**	**1.3221**	**1.5758**	**1.8029**

4.6.3 Searching the rank-1 lattice with maximized separate distance

Given the number of primes M, the dimension d, and the number of lattices points N, we try to find the optimal base vector b^* and its corresponding lattice points X^* such that the separation distance ρ_{X^*} is maximized over a candidate set. We adopt the algebra field based construction formula in [78] to construct the base vector of a rank-1 lattice. Instead of using the same predefined form as [78], we adopt a searching procedure as summarized in Algorithm 10. The main idea is a greedy search starting from a set of M prime numbers. For each prime number p, it also searches the p offset from 0 to $p-1$ to construct the possible base vector b and its corresponding X. After the greedy search procedure, the algorithm returns the optimal base vector b^* and the lattice points set X^* that obtains the maximum separation distance. Algorithm 10 can be extended by including successive coordinate search (SCS) [1, 117] as an inner searching procedure. The extended method is summarized in Algorithm 11. This method can achieve superior performance compared to other baselines.

4.6.4 Comparison of minimum distance generated by different methods

We evaluate the proposed Algorithm 10 and Algorithm 11 by comparing them with searching in Korobov form [96] and SCS [1,117]. We fix $M = 50$ for Algorithm 10 and Algorithm 11 in all the experiments. The number of iterations of SCS search [1,117] is

Table 4.3: Minimum distance ($2\rho_X$) of 3,000 lattice points in $[0,1]^d$ for $d = 10$, $d = 20$, $d = 30$, $d = 40$ and $d = 50$.

	$d = 10$	$d = 20$	$d = 30$	$d = 40$	$d = 50$
Algorithm 10	0.53359	0.93051	1.2292	1.4696	1.7009
Korobov	0.50000	0.67185	0.82285	0.95015	1.0623
SCS	0.52705	0.74536	0.91287	1.0541	1.1785
Algorithm 11	**0.56610**	**0.98601**	**1.2979**	**1.5553**	**1.7771**

set to $T = 150$, and number of iterations of SCS search as a subroutine in Algorithm 11 is set to $T = 3$.

The minimum distances ($2\rho_X$) of $1,000$ points, $2,000$ points and $3,000$ points generated by different methods are summarized in Tables 4.1, 4.2 and 4.3, respectively. Algorithm 11 can achieve a larger separate (minimum) distance than other searching methods. This means that Algorithm 11 can generate points set with a smaller covering radius (fill distance). Thus, it can generate more robust initialization for BO. Moreover, Algorithm 11 can also be used to generate points for integral approximation on $[0,1]^d$.

4.6.5 Comparison between lattice points and random points

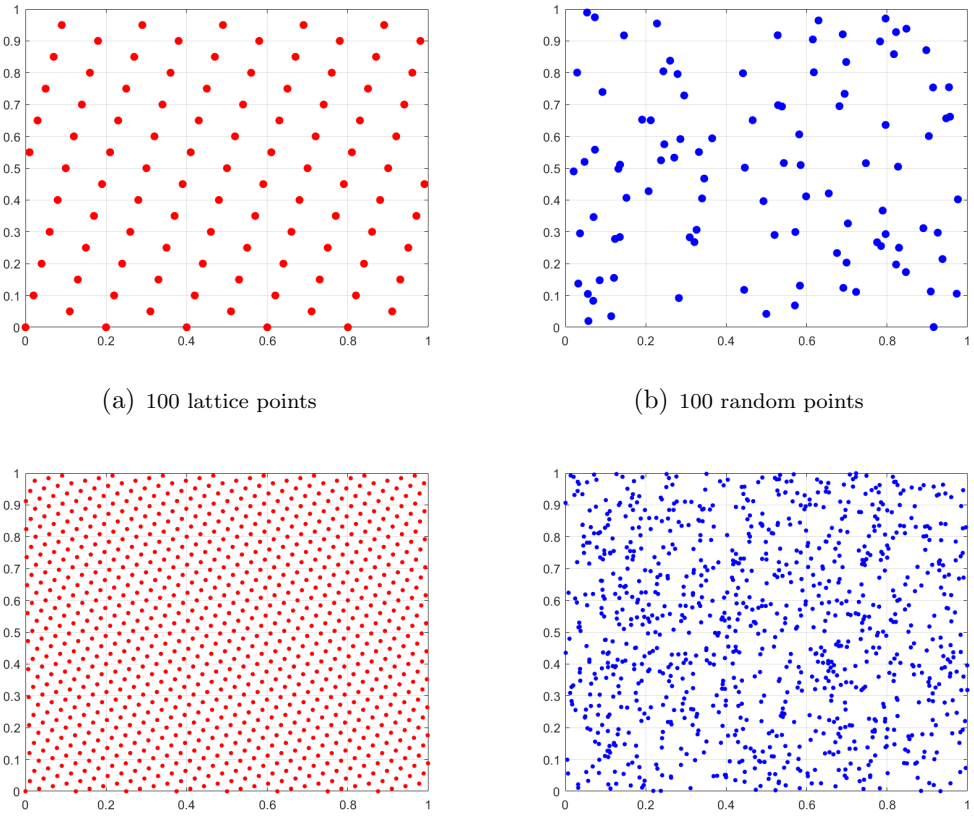
The points generated by Algorithm 10 and uniform sampling are presented in Figure 4.2. We can observe that the points generated by Algorithm 10 cut the domain into several cells. It obtains a smaller covering radius (fill distance) than the random sampling. Thus, it can be used as a robust initialization of BO.

4.7 Experiments

In this section, we focus on the evaluation of the proposed batch method. We evaluate the proposed Batch kernel optimization (BKOP) by comparing it with GP-BUCB [45] and GP-UCB-PE [40] on several synthetic benchmark test problems, hyperparameter tuning of a deep network on CIFAR100 [98] and the robot pushing task in [171].

4.7.1 Comparison with Bull's Non-adaptive Batch Method

[32] presents a non-adaptive batch method with all the query points except one being fixed at the beginning. As mentioned by Bull, this method is not practical. However, [32] does not present an adaptive batch method. We compare our adaptive batch method with Bull's non-adaptive method on Rosebrock and Ackley functions.



(a) 100 lattice points

(b) 100 random points

(c) 1000 lattice points

(d) 1000 random points

Figure 4.2: More Lattice Points and Random Points on $[0, 1]^2$

The mean values of simple regret over 30 independent runs are presented in Figure 4.3, which shows that Bull's non-adaptive method has a very slowly decreasing simple regret. Moreover, we find that Bull's non-adaptive method performs particularly worse when the size of the domain is large and the dimension of the problem is high.

4.7.2 Empirical Evaluation on Synthetic Benchmark Problems

Synthetic benchmark problems: The synthetic test functions and the domains employed are listed in Table 4.4, which includes nonconvex, nonsmooth and multimodal functions.

We fix the weight of the covariance term in the acquisition function of BKOP to one in all the experiments. For all the synthetic test problems, we set the dimension of the domain $d = 6$, and we set the batch size to $L = 5$ and $L = 10$ for all the batch BO algorithms. We use the ARD Matérn 5/2 kernel for all the methods. Instead of finding

Table 4.4: Benchmark functions

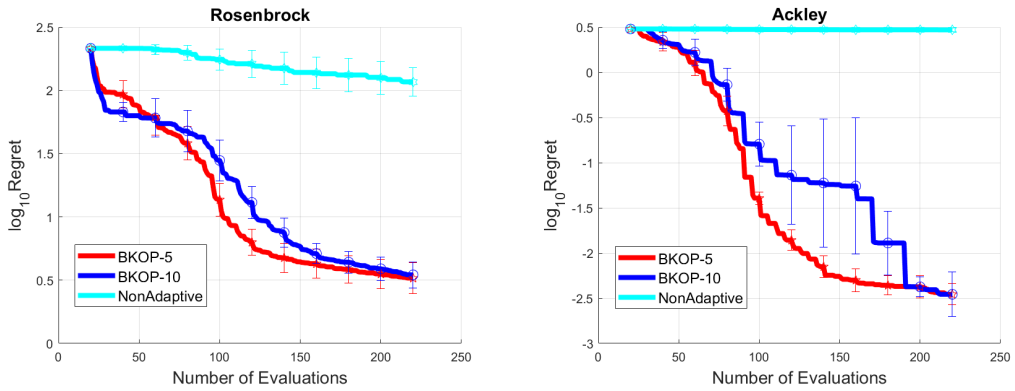
name	function	domain
Rosenbrock	$\sum_{i=1}^{d-1}\left(100(x_{i+1}-x_i^2)^2+(1-x_i)^2\right)$	$[-2,2]^d$
Nesterov	$\frac{1}{4}\lvert x_1-1\rvert+\sum_{i=1}^{d-1}\lvert x_{i+1}-2\lvert x_i\rvert+1\rvert$	$[-2,2]^d$
Different-Powers	$\sum_{i=1}^{d}\lvert x_i\rvert^{2+10\frac{i-1}{d-1}}$	$[-2,2]^d$
Dixon-Price	$(x_1-1)^2+\sum_{i=2}^{d} i(2x_i^2-x_{i-1})^2$	$[-2,2]^d$
Ackley	$-20\exp(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2})-\exp(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i))+20+\exp(1)$	$[-2,2]^d$
Levy	$\sin^2(\pi w_1)+\sum_{i=1}^{d-1}(w_i-1)^2(1+10\sin^2(\pi w_i+1))$ $+(w_d-1)^2(1+\sin^2(2\pi w_d))$ where $w_i=1+(x_i-1)/4,\ i\in\{1,...,d\}$	$[-10,10]^d$

the optimum by discrete approximation, we employ the CMA-ES algorithm [70] to optimize the acquisition function in the continuous domain $\mathcal{X}$ for all the methods, which usually improves the performance compared with discrete approximation. For each test problem, we use 20 rank-1 lattice points resized in the domain $\mathcal{X}$ as the initialization. All the methods use the same initial points.

The mean value and error bar of the simple regret over 30 independent runs concerning different algorithms are presented in Figure 4.5. We can observe that BKOP with batch sizes 5 and 10 performs better than the other methods with the same batch size. Moreover, algorithms with batch size 5 achieve faster-decreasing regret compared with batch size 10. BKOP achieves significantly low regret compared with the other methods on the Different-Powers and Rosenbrock test functions.

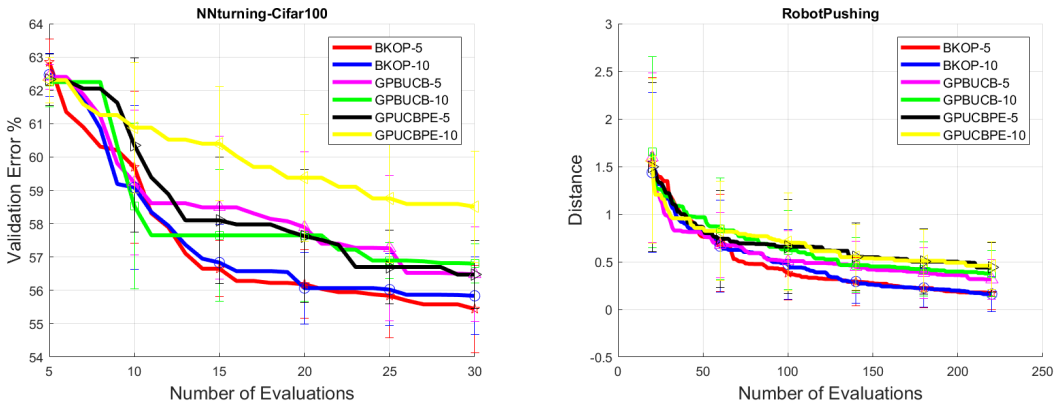
4.7.3 Empirical Evaluation on Hyperparameter tuning of Neural Network

We evaluate BKOP on hyperparameter tuning of the network on the CIFAR100 dataset. The network we employed contains three hidden building blocks, each one consists of one convolution layer, one batch normalization layer and one RELU layer. The depth of a building block is defined as the repeat number of these three layers. Seven hyperparameters are used in total for searching, namely, the depth of the building block ($\{1,2,3\}$), the initialized learning rate for SGD ($[10^{-4},10^{-1}]$), the momentum weight ($[0.1,0.95]$), weight of L2 regularization ($[10^{-10},10^{-2}]$), and three



(a) Rosenbrock function (b) Ackley function

Figure 4.3: The mean value of simple regret over 30 runs on Rosenbrock and Ackley function

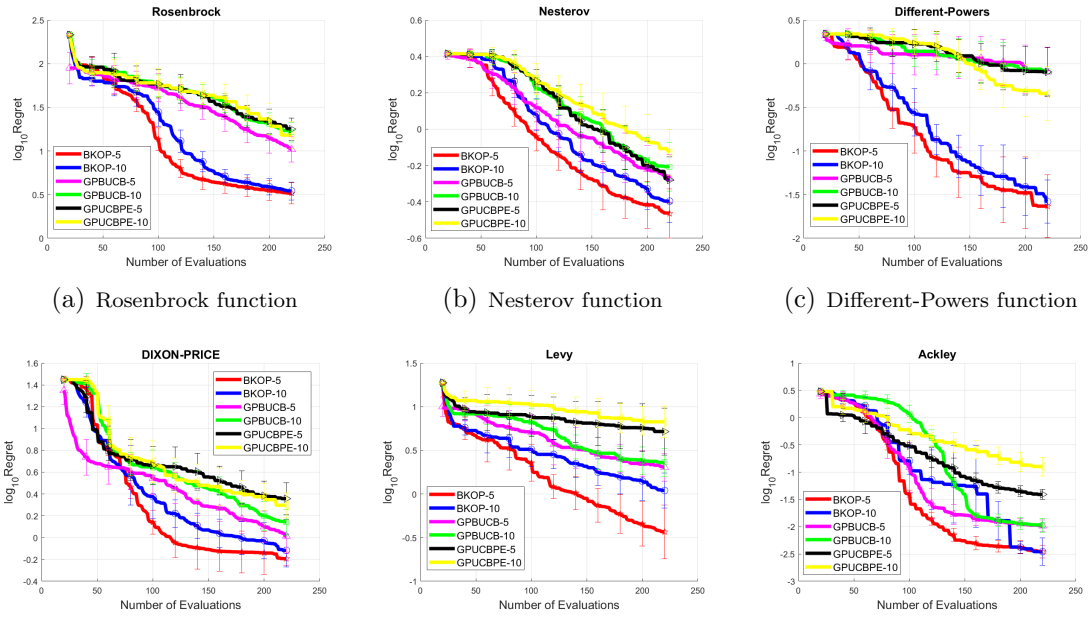


(a) Simple regret on network tuning task on CIFAR100 (b) Simple regret on robot pushing task

Figure 4.4: The mean value of simple regret on network tuning task and robot pushing task.

hyperparameters related to the filter size for each building block, the domain of these three parameters is $\{2 \times 2, 3 \times 3, 4 \times 4\}$. We employ the default training set (i.e., $50,000$ samples) for training, and use the default test set (i.e., $10,000$ samples) to compute the validation error regret of automatic hyperparameter tuning for all the methods.

We employ five rank-1 lattice points resized in the domain as the initialization. All the methods use the same initial points. The mean value of the simple regret of the validation error in percentage over 10 independent runs is presented in Figure 4.4(a). We can observe that BKOP with both batch size 5 and 10 outperforms the others. Moreover, the performance of GP-UCB-PE with batch size 10 is worse than the others.



(a) Rosenbrock function (b) Nesterov function (c) Different-Powers function

(d) Dixon-Price function (e) Levy function (f) Ackley function

Figure 4.5: The mean value of simple regret for different algorithms over 30 runs on different test functions

4.7.4 Empirical Evaluation on Robot Pushing Task

We further evaluate the performance of BKOP on the robot pushing task in [171]. The goal of this task is to select a good action for pushing an object to a target location. The 4-dimensional robot pushing problem consists of the robot location (x, y) and angle θ and the pushing duration τ as the input. And it outputs the distance between the pushed object and the target location as the function value. We employ 20 rank-1 lattice points as initialization. All the methods use the same initialization points. Thirty goal locations are randomly generated for testing. All the methods use the same goal locations. The mean value and error bars over 30 trials are presented in Figure 4.4(b). We can observe that BKOP with both batch size 5 and batch size 10 can achieve lower regret compared with GP-BUCB and GP-UCB-PE. Moreover, BKOP with a batch-size 10 obtains a very competitive regret compared with BKOP with a batch-size 5. This shows that BKOP can be scalable to a large batch. In addition, we can observe that BKOP with a batch-size 10 achieve a lower regret compared with GP-UCB-PE with a small batch-size 5. This shows a great sample efficiency of BKOP as a batch Bayesian optimization algorithm.

4.8 Summary

In this Chapter, I analyzed black-box optimization for functions with a bounded norm in RKHS. For sequential BO, I obtain a similar acquisition function to GP-UCB, but with a constant deviation weight. For batch BO, I proposed the BKOP algorithm, which is competitive with, or better than, other batch confidence-bound methods on a variety of tasks. Theoretically, I derive regret bounds for both the sequential and batch cases regardless of the choice of kernels, which are more general than the previous studies. Furthermore, I derive adversarial regret bounds with respect to the covering radius, which provides an important insight to design robust initialization for BO. To this end, I proposed fast searching methods to construct a good rank-1 lattice. Empirically, the proposed searching methods can obtain a large packing radius (separate distance).

Chapter 5

Subgroup-based Rank-1 Lattice Quasi-Monte Carlo

5.1 Chapter Abstract

Quasi-Monte Carlo (QMC) is an essential tool for integral approximation, Bayesian inference, and sampling for simulation in science, etc. In the QMC area, the rank-1 lattice is important due to its simple operation, and nice properties for point set construction. However, the construction of the generating vector of the rank-1 lattice is usually time-consuming because of an exhaustive computer search. In this chapter, we propose a simple closed-form rank-1 lattice construction method based on group theory. Our method reduces the number of distinct pairwise distance values to generate a more regular lattice. We theoretically prove a lower and an upper bound of the minimum pairwise distance of any non-degenerate rank-1 lattice. Empirically, our methods can generate a near-optimal rank-1 lattice compared with the Korobov exhaustive search regarding the l_1-norm and l_2-norm minimum distance. Moreover, experimental results show that our method achieves superior approximation performance on benchmark integration test problems and kernel approximation problems.

5.2 Background of Lattice

We first give the definition and the properties of lattices in Section 5.2.1. Then we introduce the minimum distance criterion for lattice construction in Section 5.2.2.

5.2.1 The Lattice

A d-dimensional lattice Λ is a set of points that contains no limit points and satisfies [116]

$$\forall \boldsymbol{x}, \boldsymbol{x}' \in \Lambda \Rightarrow \boldsymbol{x} + \boldsymbol{x}' \in \Lambda \text{ and } \boldsymbol{x} - \boldsymbol{x}' \in \Lambda. \tag{5.1}$$

A widely known lattice is the unit lattice $\mathbb{Z}^d$ whose components are all integers. A general lattice is constructed by a generator matrix. Given a generator matrix $\boldsymbol{B} \in \mathbb{R}^{d\times d}$, a d-dimensional lattice Λ can be constructed as

$$\Lambda = \{\boldsymbol{B}\boldsymbol{y} \big|\ \boldsymbol{y} \in \mathbb{Z}^d\}. \tag{5.2}$$

A generator matrix is not unique to a lattice Λ, namely, a lattice Λ can be obtained from a different generator matrices.

A lattice point set for integration is constructed as $\Lambda \cap [0,1)^d$. This step may require an additional search (or check) for all the points inside the unit cube.

A rank-1 lattice is a special case of the general lattice, which has a simple operation for point set construction instead of directly using Eq.(5.2). A rank-1 lattice point set can be constructed as

$$\boldsymbol{x}_i := \left\langle \frac{i\boldsymbol{z}}{n} \right\rangle, i \in \{0, 1, \cdots, n-1\}, \tag{5.3}$$

where $\boldsymbol{z} \in \mathbb{Z}^d$ is the so-called generating vector, and the big $\langle\cdot\rangle$ denotes the operation of taking the fractional part of the input number elementwise. Compared with the general lattice rule, the construction form of the rank-1 lattice already ensures the constructed points to be inside the unit cube without the need for any further checks.

Given a rank-1 lattice set X in the unit cube, we can also construct a randomized point set. Sample a random variable $\boldsymbol{\Delta} \sim Uniform[0,1]^d$, we can construct a point set $\widetilde{X}$ by random shift as [46]

$$\widetilde{X} = \langle X + \boldsymbol{\Delta} \rangle. \tag{5.4}$$

5.2.2 The separating distance of a lattice

Several criteria have been studied in the literature for good lattice construction through computer search. Worst case error is one of the most widely used criteria for functions in a reproducing kernel Hilbert space (RKHS) [46]. However, this criterion requires the prior knowledge of functions and the assumption of the RKHS. Without assumptions of the functions, it is reasonable to construct a good lattice by

designing an evenly spaced point set. Minimizing the covering radius is a good way for evenly spaced point set construction.

As minimizing the covering radius of the lattice is equivalent to maximizing the packing radius [43], we can construct the point set through maximizing the packing radius (separating distance) of the lattice. Define the covering radius and packing radius of a set of points $X = \{x_1, ..., x_N\}$ as Eq.(5.5) and Eq.(5.6), respectively:

$$h_X = \sup_{x \in \mathcal{X}} \min_{x_k \in X} \|x - x_k\|, \tag{5.5}$$

$$\rho_X = \frac{1}{2} \min_{\substack{x_i, x_j \in X, \\ x_i \neq x_j}} \|x_i - x_j\|. \tag{5.6}$$

The l_p-norm-based toroidal distance [64] between two lattice points $\mathbf{x} \in [0,1]^d$ and $\mathbf{x}' \in [0,1]^d$ can be defined as:

$$\|\mathbf{x} - \mathbf{x}'\|_{T_p} := \left(\sum_{i=1}^{d} (\min(|x_i - x_i'|, 1 - |x_i - x_i'|))^p \right)^{\frac{1}{p}} \tag{5.7}$$

Because the difference (subtraction) between two lattice points is still a lattice point, and a rank-1 lattice has a period 1, the packing radius ρ_X of a rank-1 lattice can be calculated as

$$\rho_X = \min_{\mathbf{x} \in X \setminus \mathbf{0}} \frac{1}{2} \|\mathbf{x}\|_{T_2}, \tag{5.8}$$

where $\|\mathbf{x}\|_{T_2}$ denotes the l_2-norm-based toroidal distance between $\mathbf{x}$ and $\mathbf{0}$, symbol $X \setminus \mathbf{0}$ denotes the set X excludes the point $\mathbf{0}$. This formulation calculates the packing radius with a time complexity of $\mathcal{O}(Nd)$ rather than $\mathcal{O}(N^2 d)$ in pairwise comparison. However, the computation of the packing radius is not easily accelerated by fast Fourier transform due to the minimum operation in Eq.(5.8).

5.3 Subgroup-based Rank-1 Lattice

In this section, we derive our construction of a rank-1 lattice based on the subgroup theory. Then we analyze the properties of our method. We provide detailed proofs in the supplement.

5.3.1 Construction of the Generating Vector

From the definition of rank-1 lattice, we know the packing radius of rank-1 lattice with n points can be reformulated as

$$\rho_X = \min_{i \in \{1, \cdots, n-1\}} \frac{1}{2} \|\mathbf{x}_i\|_{T_2}, \tag{5.9}$$

where

$$\boldsymbol{x}_i := \frac{i\boldsymbol{z} \bmod n}{n}, i \in \{1, ..., n-1\}. \tag{5.10}$$

Then, we have

$$\begin{aligned}\rho_X &= \min_{i\in\{1,\cdots,n-1\}} \frac{1}{2} \left\| \min\left(\frac{i\boldsymbol{z} \bmod n}{n}, \frac{n - i\boldsymbol{z} \bmod n}{n} \right) \right\|_2 \\ &= \min_{i\in\{1,\cdots,n-1\}} \frac{1}{2} \left\| \min\left(\frac{i\boldsymbol{z} \bmod n}{n}, \frac{(-i\boldsymbol{z}) \bmod n}{n} \right) \right\|_2,\end{aligned} \tag{5.11}$$

where $\min(\cdot,\cdot)$ denotes the elementwise min operation between two inputs.

Suppose n is a prime number, from number theory, we know that for a primitive root g, the residue of $\{g^0, g^1, \cdots, g^{n-2}\}$ modulo n forms a cyclic group under multiplication, and $g^{n-1} \equiv 1 \bmod n$. Since $(g^{\frac{n-1}{2}})^2 = g^{n-1} \equiv 1 \bmod n$, we know that $g^{\frac{n-1}{2}} \equiv -1 \bmod n$.

Because of the one-to-one correspondence between the residue of $\{g^0, g^1, \cdots, g^{n-2}\}$ modulo n and the set $\{1, 2, \cdots, n-1\}$, we can construct the generating vector as

$$\boldsymbol{z} = [g^{m_1}, g^{m_2}, \cdots, g^{m_d}] \bmod n \tag{5.12}$$

without loss of generality, where $m_1, \cdots, m_d$ are integer components to be designed. Denote $\bar{\boldsymbol{z}} = [g^{\frac{n-1}{2}+m_1}, g^{\frac{n-1}{2}+m_2}, \cdots, g^{\frac{n-1}{2}+m_d}] \bmod n$, maximizing the separating distance ρ_X is equivalent to maximizing

$$J = \min_{k\in\{0,\cdots,n-2\}} \left\| \min(g^k\boldsymbol{z} \bmod n, g^k\bar{\boldsymbol{z}} \bmod n) \right\|_2. \tag{5.13}$$

Suppose $2d$ divides $n-1$, i.e., $2d|(n-1)$, by setting $m_i = g^{\frac{(i-1)(n-1)}{2d}}$ for $i \in \{1, \cdots, d\}$, we know that $H = \{g^{m_1}, g^{m_2}, \cdots, g^{m_d}, g^{\frac{n-1}{2}+m_1}, g^{\frac{n-1}{2}+m_2}, \cdots, g^{\frac{n-1}{2}+m_d}\}$ is equivalent to setting $\{g^0, g^{\frac{n-1}{2d}}, \cdots, g^{\frac{(2d-1)(n-1)}{2d}}\} \bmod n$, and it forms a subgroup of the group $\{g^0, g^1, \cdots, g^{n-2}\} \bmod n$. From Lagrange's theorem in group theory [53], we know that the cosets of the subgroup H partition the entire group $\{g^0, g^1, \cdots, g^{n-2}\}$ into equal-size, non-overlapping sets, and the number of cosets of H is $\frac{n-1}{2d}$. Thus, we know that distance $\min(g^k\boldsymbol{z} \bmod n, g^k\bar{\boldsymbol{z}} \bmod n)$ for $k \in \{0, \cdots, n-2\}$ has $\frac{n-1}{2d}$ different values, and there are the same numbers of items for each value.

Thus, we can construct the generating vector as

$$\boldsymbol{z} = [g^0, g^{\frac{n-1}{2d}}, g^{\frac{2(n-1)}{2d}}, \cdots, g^{\frac{(d-1)(n-1)}{2d}}] \bmod n. \tag{5.14}$$

In this way, the constructed rank-1 lattice is more regular as it has few different distinct pairwise distance values, and for each distance, the same number of items

obtain this value. Usually, the constructed regular lattice is more evenly spaced, and it has a large minimum pairwise distance. We confirm this empirically through extensive experiments in Section 5.5.

We summarize our construction method and the properties of the constructed rank-1 lattice in Theorem 13.

Theorem 13. *Suppose n is a prime number and $2d|(n-1)$. Let g be a primitive root of n. Let $\boldsymbol{z} = [g^0, g^{\frac{n-1}{2d}}, g^{\frac{2(n-1)}{2d}}, \cdots, g^{\frac{(d-1)(n-1)}{2d}}] \bmod n$. Construct a rank-1 lattice $X = \{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_{n-1}\}$ with $\boldsymbol{x}_i = \frac{i\boldsymbol{z} \bmod n}{n}, i \in \{0, ..., n-1\}$. Then, there are $\frac{n-1}{2d}$ distinct pairwise toroidal distance values among X, and each distance value is taken by the same number of pairs in X.*

As shown in Theorem 13, our method can construct regular rank-1 lattice through a very simple closed-form construction, which does not require any exhaustive computer search.

5.3.2 Regular Property of Rank-1 Lattice

We show the regular property of rank-1 lattices in terms of l_p-norm-based toroidal distance.

Theorem 14. *Suppose n is a prime number and $n \geq 2d+1$. Let $\boldsymbol{z} = [z_1, z_2, \cdots, z_d]$ with $1 \leq z_k \leq n-1$. Construct a rank-1 lattice $X = \{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_{n-1}\}$ with $\boldsymbol{x}_i = \frac{i\boldsymbol{z} \bmod n}{n}, i \in \{0, ..., n-1\}$ and $z_i \neq z_j$. Then, the minimum pairwise toroidal distance can be bounded as*

$$\frac{d(d+1)}{2n} \leq \min_{i,j \in \{0,\cdots,n-1\}, i \neq j} \|\mathbf{x}_i - \mathbf{x}_j\|_{T_1} \leq \frac{(n+1)d}{4n} \tag{5.15}$$

$$\frac{\sqrt{6d(d+1)(2d+1)}}{6n} \leq \min_{i,j \in \{0,\cdots,n-1\}, i \neq j} \|\mathbf{x}_i - \mathbf{x}_j\|_{T_2} \leq \sqrt{\frac{(n+1)d}{12n}}, \tag{5.16}$$

where $\|\cdot\|_{T_1}$ and $\|\cdot\|_{T_2}$ denote the l_1-norm-based toroidal distance and the l_2-norm-based toroidal distance, respectively.

Theorem 14 gives the upper and lower bounds of the minimum pairwise distance of any non-degenerate rank-1 lattice. The term 'non-degenerate' means that the elements in the generating vector are not equal, i.e., $z_i \neq z_j$.

We now show that our subgroup-based rank-1 lattice can achieve the optimal minimum pairwise distance when $n = 2d+1$ is a prime number.

Corollary 3. *Suppose $n = 2d + 1$ is a prime number. Let g be a primitive root of n. Let $\boldsymbol{z} = [g^0, g^{\frac{n-1}{2d}}, g^{\frac{2(n-1)}{2d}}, \cdots, g^{\frac{(d-1)(n-1)}{2d}}] \bmod n$. Construct rank-1 lattice $X = \{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_{n-1}\}$ with $\boldsymbol{x}_i = \frac{i\boldsymbol{z} \bmod n}{n}, i \in \{0, ..., n-1\}$. Then, the pairwise toroidal distance of the lattice X attains the upper bound.*

$$\|\mathbf{x}_i - \mathbf{x}_j\|_{T_1} = \frac{(n+1)d}{4n}, \forall i, j \in \{0, \cdots, n-1\}, i \neq j, \tag{5.17}$$

$$\|\mathbf{x}_i - \mathbf{x}_j\|_{T_2} = \sqrt{\frac{(n+1)d}{12n}}, \forall i, j \in \{0, \cdots, n-1\}, i \neq j. \tag{5.18}$$

Corollary 1 shows a case when our subgroup rank-1 lattice obtains the maximum minimum pairwise toroidal distance. It is useful for expensive black-box functions, where the number of function queries is small. Empirically, we find that our subgroup rank-1 lattice can achieve near-optimal pairwise toroidal distance in many other cases.

5.4 QMC for Kernel Approximation

Another application of our subgroup rank-1 lattice is kernel approximation. Kernel approximation has been widely studied. A random feature maps is a promising way for kernel approximation. Rahimi et al. study the shift-invariant kernels by Monte Carlo sampling [140]. Yang et al. suggest employing QMC for kernel approximation [15, 179]. Several previous methods work on the construction of structured feature maps for kernel approximation [39, 105, 117]. Apart from other kernel approximation methods designed for specific kernels, QMC can serve as a plug-in for any integral representation of kernels to improve kernel approximation. We include this section to be self-contained.

From Bochner's Theorem, shift invariant kernels can be expressed as an integral [140]

$$\mathrm{K}(\boldsymbol{x}, \boldsymbol{y}) = \int_{\mathbb{R}^d} e^{-i(\boldsymbol{x}-\boldsymbol{y})^\top \mathbf{w}} p(\mathbf{w}) \mathbf{dw}, \tag{5.19}$$

where $i = \sqrt{-1}$, and $p(\mathbf{w})$ is a probability density. $p(\mathbf{w}) = p(-\mathbf{w}) \geq 0$ ensure the imaginary parts of the integral vanish. Eq.(5.19) can be rewritten as

$$\mathrm{K}(\boldsymbol{x}, \boldsymbol{y}) = \int_{[0,1]^d} e^{-i(\boldsymbol{x}-\boldsymbol{y})^\top \Phi^{-1}(\boldsymbol{\epsilon})} d\boldsymbol{\epsilon}. \tag{5.20}$$

We can approximate the integral Eq.(5.19) by using our subgroup rank-1 lattice according to the QMC approximation in [166, 179]

$$\mathrm{K}(\boldsymbol{x}, \boldsymbol{y}) = \int_{[0,1]^d} e^{-i(\boldsymbol{x}-\boldsymbol{y})^\top \Phi^{-1}(\boldsymbol{\epsilon})} d\boldsymbol{\epsilon} \approx \frac{1}{n} \sum_{i=1}^{n} e^{-i(\boldsymbol{x}-\boldsymbol{y})^\top \Phi^{-1}(\boldsymbol{\epsilon}_i)} = \langle \Psi(\boldsymbol{x}), \Psi(\boldsymbol{y}) \rangle, \tag{5.21}$$

where $\Psi(\boldsymbol{x}) = \frac{1}{\sqrt{n}} \left[e^{-i\boldsymbol{x}^\top \Phi^{-1}(\boldsymbol{\epsilon}_1)}, \cdots, e^{-i\boldsymbol{x}^\top \Phi^{-1}(\boldsymbol{\epsilon}_n)} \right]$ is the feature map of the input $\boldsymbol{x}$.

Table 5.1: Minimum l_1-norm-based toroidal distance of rank-1 lattice constructed by different methods.

d=50		n=101	401	601	701	1201	1301	1601	1801	1901	2801
	SubGroup	**12.624**	**11.419**	**11.371**	**11.354**	**11.029**	**10.988**	10.541	10.501	10.454	**10.748**
	Hua [78]	10.426	10.421	9.8120	10.267	10.074	9.3982	9.5890	9.5175	8.9868	9.2260
	Korobov [97]	**12.624**	**11.419**	**11.371**	**11.354**	**11.029**	**10.988**	**10.665**	**10.561**	**10.701**	**10.748**
		401	601	1201	1601	1801	2801	3001	4001	4201	4801
d=100	SubGroup	**24.097**	**23.760**	22.887	**23.342**	22.711	**23.324**	22.233	**22.437**	22.573	21.190
	Hua [78]	21.050	21.251	21.205	20.675	19.857	20.683	20.700	19.920	19.967	20.574
	Korobov [97]	**24.097**	**23.760**	**23.167**	**23.342**	**22.893**	**23.324**	**22.464**	**22.437**	**22.573**	**22.188**
		401	1201	1601	2801	4001	4801	9601	12401	14401	15601
d=200	SubGroup	**50.125**	**48.712**	47.500	47.075	**47.810**	45.957	45.819	**46.223**	43.982	**45.936**
	Hua [78]	43.062	43.057	43.052	43.055	43.053	43.055	43.053	42.589	42.558	42.312
	Korobov [97]	**50.125**	**48.712**	**47.660**	**47.246**	**47.810**	**46.686**	**46.154**	**46.223**	**45.949**	**45.936**
		3001	4001	7001	9001	13001	16001	19001	21001	24001	28001
d=500	SubGroup	**121.90**	**121.99**	119.60	118.63	**120.23**	**119.97**	116.41	**120.56**	**120.24**	113.96
	Hua [78]	108.33	108.33	108.33	108.33	108.33	108.33	108.33	108.33	108.33	108.33
	Korobov [97]	**121.90**	**121.99**	**120.46**	**120.16**	**120.23**	**119.97**	**119.41**	**120.56**	**120.24**	**118.86**

Table 5.2: Minimum l_2-norm-based toroidal distance of rank-1 lattice constructed by different methods.

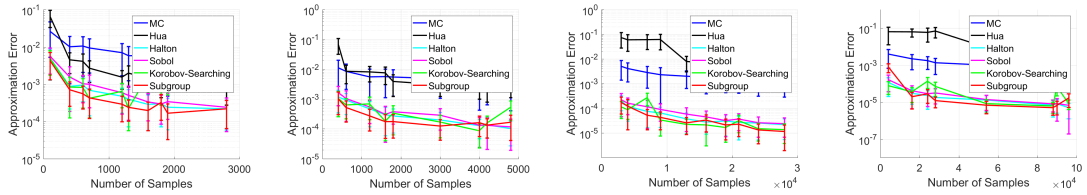
d=50		n=101	401	601	701	1201	1301	1601	1801	1901	2801
	SubGroup	**2.0513**	**1.9075**	**1.9469**	**1.9196**	**1.8754**	1.8019	1.8008	**1.8709**	1.7844	1.7603
	Hua [78]	1.7862	1.7512	1.7293	1.7049	1.7326	1.6295	1.6659	1.6040	1.5629	1.5990
	Korobov [97]	**2.0513**	**1.9075**	**1.9469**	**1.9196**	**1.8754**	**1.8390**	**1.8356**	**1.8709**	**1.8171**	**1.8327**
		401	601	1201	1601	1801	2801	3001	4001	4201	4801
d=100	SubGroup	**2.8342**	**2.8143**	2.7077	**2.7645**	**2.7514**	2.6497	2.6337	2.6410	2.6195	2.5678
	Hua [78]	2.5385	2.5739	2.4965	2.4783	2.4132	2.5019	2.4720	2.4138	2.4537	2.4937
	Korobov [97]	**2.8342**	**2.8143**	**2.7409**	**2.7645**	**2.7514**	**2.6956**	**2.6709**	**2.6562**	**2.6667**	**2.6858**
		401	1201	1601	2801	4001	4801	9601	12401	14401	15601
d=200	SubGroup	**4.0876**	**3.9717**	**3.9791**	3.8425	**3.9276**	3.8035	3.7822	**3.8687**	3.6952	3.8370
	Hua [78]	3.7332	3.7025	3.6902	3.6944	3.7148	3.6936	3.6571	3.5625	3.6259	3.5996
	Korobov [97]	**4.0876**	**3.9717**	**3.9791**	**3.9281**	**3.9276**	**3.9074**	**3.8561**	**3.8687**	**3.8388**	**3.8405**
		3001	4001	7001	9001	13001	16001	19001	21001	24001	28001
d=500	SubGroup	**6.3359**	**6.3769**	6.3141	6.2131	**6.2848**	6.2535	6.0656	**6.2386**	**6.2673**	6.1632
	Hua [78]	5.9216	5.9216	5.9215	5.9215	5.9216	5.9216	5.9215	5.9215	5.8853	5.9038
	Korobov [97]	**6.3359**	**6.3769**	**6.3146**	**6.2960**	**6.2848**	**6.2549**	**6.2611**	**6.2386**	**6.2673**	**6.2422**

5.5 Experiments

In this section, we first evaluate the minimum distance generated by our subgroup rank-1 lattice in section 5.5.1. We then evaluate the subgroup rank-1 lattice on integral approximation tasks and kernel approximation task in section 5.5.2 and 5.5.3, respectively.

5.5.1 Evaluation of the minimum distance

We evaluate the minimum distance of our subgroup rank-1 lattice by comparing with Hua's method [78] and the Korobov [97] searching method. We denote 'SubGroup' as our subgroup rank-1 lattice, 'Hua' as rank-1 lattice constructed by Hua's method [78], and 'Korobov' as rank-1 lattice constructed by exhaustive computer search in Korobov form [97].



(a) 50-d Integral Approximation (b) 100-d Integral Approximation (c) 500-d Integral Approximation (d) 1000-d Integral Approximation

Figure 5.1: Mean approximation error over 50 independent runs.error bars are with in 1× std.

We set the dimension d as in $\{50, 100, 200, 500\}$. For each dimension d, we set the number of points n as the first ten prime numbers such that $2d$ divides $n-1$, i.e., $2d \big| (n-1)$. The minimum l_1-norm-based toroidal distance and the minimum l_2-norm-based toroidal distance for each dimension are reported in Table 5.5.1 and Table 5.2, respectively. The larger the distance, the better.

We can observe that our subgroup rank-1 lattice achieves consistently better (larger) minimum distances than Hua's method in all the cases. Moreover, we see that subgroup rank-1 lattice obtains, in 20 out of 40 cases, the same l_2-norm-based toroidal distance and in 24 out of 40 cases the same l_1-norm-based toroidal distance compared with the exhaustive computer search in Korobov form. The experiments show that our subgroup rank-1 lattice achieves the optimal toroidal distance in exhaustive computer searches in Korobov form in over half of all the cases. Furthermore, the experimental result shows that our subgroup rank-1 lattice obtains a competitive distance compared with the exhaustive Korobov search in the remaining cases. Note that our subgroup rank-1 lattice is a closed-form construction which does not require computer search, making our method more appealing and simple to use.

Time Comparison of Korobov searching and our sub-group rank-1 lattice. The table below shows the time cost (seconds) for lattice construction. The run time for Korobov searching grows fast to hours. Our method can run in less than one second, achieving a $10^4\times$ to $10^5\times$ speed-up. The speed-up increases when n and d becomes larger.

d=500		n=3001	4001	7001	9001	13001	16001	19001	21001	24001	28001
	SubGroup	0.0185	0.0140	0.0289	0.043	0.0386	0.0320	0.0431	0.0548	0.0562	0.0593
	Korobov	34.668	98.876	152.86	310.13	624.56	933.54	1308.9	1588.5	2058.5	2815.9
d=1000		n=4001	16001	24001	28001	54001	70001	76001	88001	90001	96001
	SubGroup	0.0388	0.0618	0.1041	0.1289	0.2158	0.2923	0.3521	0.4099	0.5352	0.5663
	Korobov	112.18	1849.4	4115.9	5754.6	20257	34842	43457	56798	56644	69323

5.5.2 Integral approximation

We evaluate our subgroup rank-1 lattice on the integration test problem

$$f(\boldsymbol{x}) := \exp\left(c\sum_{j=1}^{d} x_j j^{-b}\right) \tag{5.22}$$

$$I(f) := \int_{[0,1]^d} f(\boldsymbol{x})d\boldsymbol{x} = \prod_{j=1}^{d} \frac{\exp(cj^{-b}) - 1}{cj^{-b}}. \tag{5.23}$$

We compare with i.i.d. Monte Carlo, a Hua's rank-1 lattice [78], Korobov searching rank-1 lattice [95], Halton sequence, and Sobol sequence [46]. For both Halton sequence and Sobol sequence, we use the scrambling technique suggested in [46]. For all the QMC methods, we use the random shift technique as in Eq.(5.4).

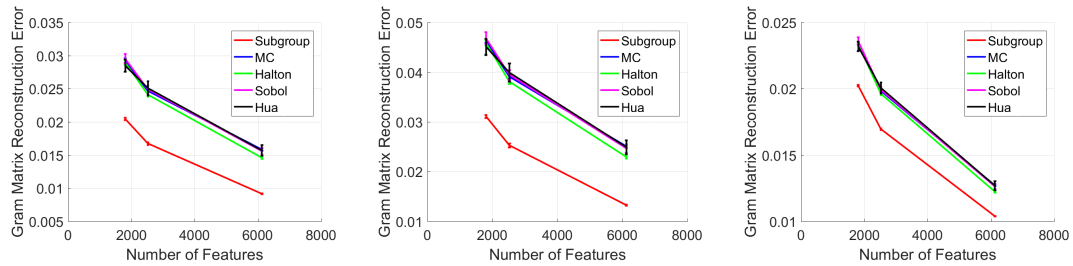
We fix $b = 2$ and $c = 1$ in all the experiments. We set dimension $d = 100$ and $d = 500$, respectively. We set the number of points n as the first ten prime numbers such that $2d$ divides $n-1$, i.e., $2d\big|(n-1)$.

The mean approximation error ($\frac{|Q(f)-I(f)|}{|I(f)|}$) with error bars over 50 independent runs for each dimension d is presented in Figure 5.1. We can see that Hua's method obtains a smaller error than i.i.d Monte Carlo on the 50-d problem, however, it becomes worse than MC on 500-d and 1000-d problems. Moreover, our subgroup rank-1 lattice obtains a consistent smaller error on all the tested problems than Hua and MC. In addition, our subgroup rank-1 lattice achieves a slightly better performance than Halton, Sobol and Korobov searching method.

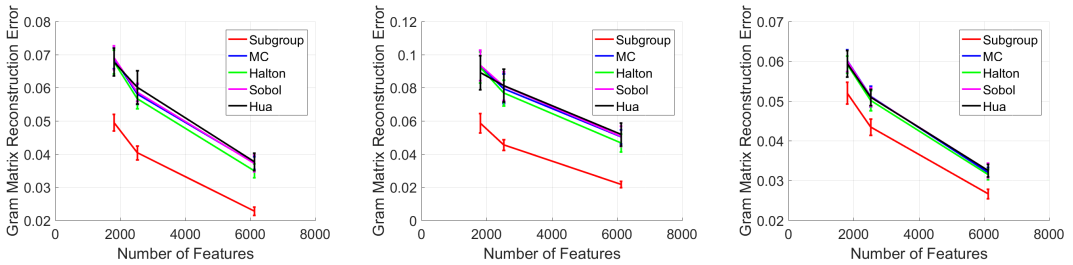
5.5.3 Kernel approximation

We evaluate the performance of subgroup rank-1 lattice on kernel approximation tasks by comparing with other QMC baseline methods. We test the kernel approximation of the Gaussian kernel, the zeroth-order arc-cosine kernel, and the first-order arc-cosine kernel as in [39].

We compare subgroup rank-1 lattice with a Hua's rank-1 lattice [78], Halton sequence, Sobol sequence [46] and standard i.i.d. Monte Carlo sampling. For both the Halton sequence and Sobol sequence, we use the scrambling technique suggested in [46]. For both subgroup rank-1 lattice and Hua's rank-1 lattice, we use the random shift as in Eq.(5.4). We evaluate the methods on the DNA [139] and the SIFT1M [81] dataset over 50 independent runs. Each run contains 2000 random samples to construct the Gram matrix. The bandwidth parameter of Gaussian kernel is set to 15 in all the experiments.



(a) $\frac{\|\widetilde{K}-K\|_F}{\|K\|_F}$ for Gaussian Kernel (b) $\frac{\|\widetilde{K}-K\|_F}{\|K\|_F}$ for First-order Arc Kernel (c) $\frac{\|\widetilde{K}-K\|_F}{\|K\|_F}$ for Zero-order Arc Kernel



(d) $\frac{\|\widetilde{K}-K\|_\infty}{\|K\|_\infty}$ for Gaussian Kernel (e) $\frac{\|\widetilde{K}-K\|_\infty}{\|K\|_\infty}$ for First-order Arc Kerne (f) $\frac{\|\widetilde{K}-K\|_\infty}{\|K\|_\infty}$ for Zero-order Arc Kernel

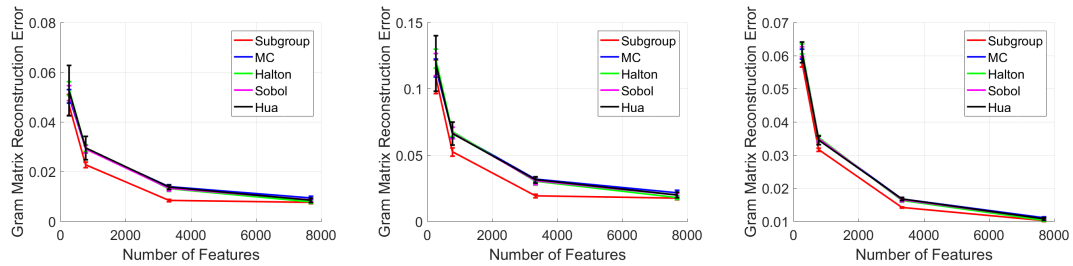
Figure 5.2: Relative Mean and Max Reconstruction Error for Gaussian, Zero-order and First-order Arc-cosine Kernel on DNA dataset. Error bars are within 1× std.

The mean Frobenius norm approximation error ($\|\widetilde{K}-K\|_F/\|K\|_F$) and maximum norm approximation error ($\|\widetilde{K}-K\|_\infty/\|K\|_\infty$) with error bars on DNA [139] dataset are plotted in Figure 5.2. The results on SIFT1M [81] is given in Figure 6 in the supplement. The experimental result shows that subgroup rank-1 lattice consistently obtains a smaller approximation error compared with other baselines.

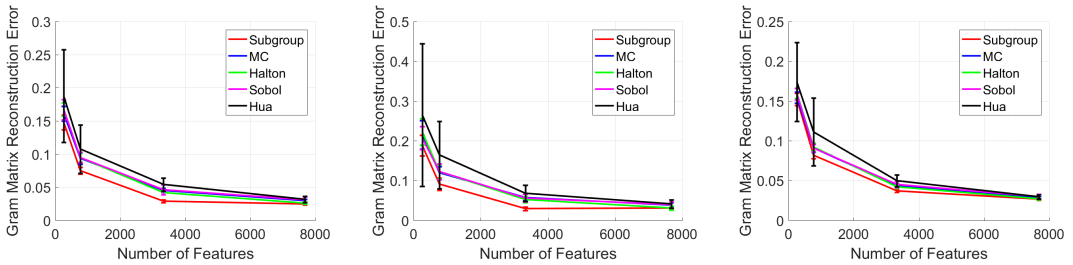
5.5.4 Approximation on Graphical Model

For general Boltzmann machines with continuous state in $[0,1]$, the energy function of $\boldsymbol{x} \in [0,1]^d$ is defined as $E(\boldsymbol{x}) = -(\boldsymbol{x}^\top \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}^\top \boldsymbol{x})/d$. The normalization constant is $Z = \int_{[0,1]^d} \exp(-E(\boldsymbol{x}))d\boldsymbol{x}$. For inference, the marginal likelihood of observation $\boldsymbol{v} \in \mathbb{R}^d$ is $\mathcal{L}(\boldsymbol{v}) = \int_{[0,1]^d} \exp(-f(\boldsymbol{v}))\exp(-E(\boldsymbol{h}))/Z\mathrm{d}\boldsymbol{h}$ with function $f(\boldsymbol{v}) = -(\boldsymbol{v}^\top \boldsymbol{W}_v \boldsymbol{v} + 2\boldsymbol{v}^\top \boldsymbol{W}_h \boldsymbol{h} + \boldsymbol{b}_v^\top \boldsymbol{v})/d$, where $\boldsymbol{h} \in \mathbb{R}^d$ denotes the hidden states.

We evaluate our method on approximation of the normalization constant and inference by comparing with i.i.d. Monte Carlo (MC), slice sampling (SS) and Hamiltonian Monte Carlo (HMC). We generate the elements of $\boldsymbol{W}$, $\boldsymbol{W}_v$, $\boldsymbol{W}_h$, $\boldsymbol{b}$ and $\boldsymbol{b}_v$ by sampling from standard Gaussian $\mathcal{N}(0,1)$. These parameters are fixed and kept the same for all the methods in comparison. For inference, we generate an observation $\boldsymbol{v} \in [0,1]^d$ by uniformly sampling and keep it fixed and same for all the methods. For



(a) $\frac{\|\widetilde{K}-K\|_F}{\|K\|_F}$ for Gaussian Kernel (b) $\frac{\|\widetilde{K}-K\|_F}{\|K\|_F}$ for First-order Arc Kernel (c) $\frac{\|\widetilde{K}-K\|_F}{\|K\|_F}$ for Zero-order Arc Kernel

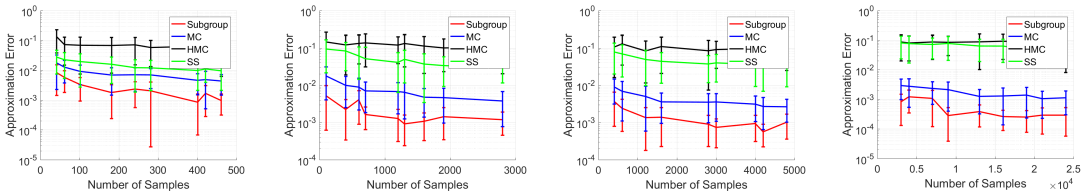


(d) $\frac{\|\widetilde{K}-K\|_\infty}{\|K\|_\infty}$ for Gaussian Kernel (e) $\frac{\|\widetilde{K}-K\|_\infty}{\|K\|_\infty}$ for First-order Arc Kernel (f) $\frac{\|\widetilde{K}-K\|_\infty}{\|K\|_\infty}$ for Zero-order Arc Kernel

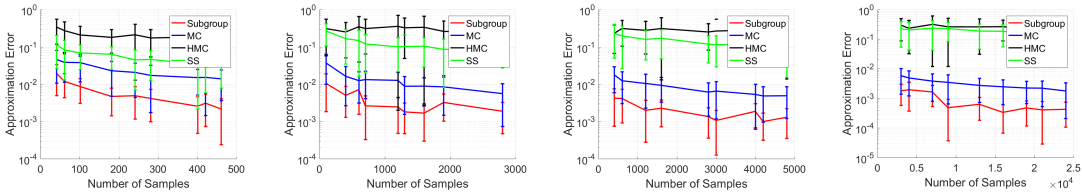
Figure 5.3: Relative Mean and Max Reconstruction Error for Gaussian, Zero-order and First-order Arc-cosine Kernel on SIFT1M dataset.

SS and HMC, we use the *slicesample* function and *hmcSampler* function in MATLAB, respectively. We use the approximation of i.i.d. MC with 10^7 samples as the pseudo ground-truth. The approximation errors $|\widehat{Z}-Z|/Z$ and $|\widehat{\mathcal{L}}-\mathcal{L}|/\mathcal{L}$ are shown in Fig.5.4(a)-5.4(d) and Fig.5.4(e)-5.4(h), respectively. our method consistently outperforms MC, HMC and SS on all cases. Moreover, our method is much cheaper than SS and HMC.

Comparison to sequential Monte Carlo. When the positive density region takes a large fraction of the entire domain, our method is very competitive. When it is only inside a small part of a large domain, our method may not be better than sequential adaptive sampling. In this case, it is interesting to take advantage of both lattice and adaptive sampling. E.g., one can employ our subgroup rank-1 lattice as a rough partition of the domain to find high mass regions, then take sequential adaptive sampling on the promising regions with the lattice points as the start points. Also, it is interesting to consider recursively apply our subgroup rank-1 lattice to refine the partition. Moreover, our subgroup-based rank-1 lattice enables black-box evaluation without the need for gradient information. In contrast, several sequential sampling methods, e.g., HMC, need a gradient of density function for sampling.



(a) 10-d$|\widehat{Z}-Z|/Z$ (b) 50-d $|\widehat{Z}-Z|/Z$ (c) 100-d $|\widehat{Z}-Z|/Z$ (d) 500-d $|\widehat{Z}-Z|/Z$



(e) 10-d $|\widehat{\mathcal{L}}-\mathcal{L}|/\mathcal{L}$ (f) 50-d $|\widehat{\mathcal{L}}-\mathcal{L}|/\mathcal{L}$ (g) 100-d $|\widehat{\mathcal{L}}-\mathcal{L}|/\mathcal{L}$ (h) 500-d $|\widehat{\mathcal{L}}-\mathcal{L}|/\mathcal{L}$

Figure 5.4: Mean approximation error over 50 independent runs. Error bars are with in 1× std

5.6 Subgroup-based QMC on Sphere $\mathbb{S}^{d-1}$

In this section, we propose a closed-form subgroup-based QMC method on the sphere $\mathbb{S}^{d-1}$ instead of unit cube $[0,1]^d$. QMC uniformly on sphere can be used to construct samples for isotropic distribution, which is helpful for variance reduction of the gradient estimators in Evolutionary strategy for reinforcement learning [149].

Lyu [117] constructs structured sampling matrix on $\mathbb{S}^{d-1}$ by minimizing the discrete Riesz energy. In contrast, we construct samples by a closed-form construction without the time-consuming optimization procedure. Our construction can achieve a small mutual coherence.

Without loss of generality, we assume that $d=2m, N=2n$, and n is a prime such that $m|(n-1)$. Let $F\in\mathbb{C}^{n\times n}$ be a $n\times n$ discrete Fourier matrix. $F_{k,j}=e^{\frac{2\pi ikj}{n}}$ is the $(k,j)^{th}$entry of F, where $i=\sqrt{-1}$. Let $\Lambda=\{k_1,k_2,...,k_m\}\subset\{1,...,n-1\}$ be a subset of indexes.

The structured sampling matrix $\mathbf{V}$ in [117] can be defined as equation (10.398).

$$\mathbf{V}=\frac{\mathbf{1}}{\sqrt{\mathbf{m}}}\begin{bmatrix}\mathrm{Re}F_\Lambda & -\mathrm{Im}F_\Lambda\\ \mathrm{Im}F_\Lambda & \mathrm{Re}F_\Lambda\end{bmatrix}\in\mathbb{R}^{\mathbf{d}\times\mathbf{N}} \tag{5.24}$$

where Re and Im denote the real and imaginary part of a complex number, and F_Λ in equation (10.399) is the matrix constructed by m rows of F

$$F_\Lambda=\begin{bmatrix}e^{\frac{2\pi ik_1 1}{n}} & \cdots & e^{\frac{2\pi ik_1 n}{n}}\\ \vdots & \ddots & \vdots\\ e^{\frac{2\pi ik_m 1}{n}} & \cdots & e^{\frac{2\pi ik_m n}{n}}\end{bmatrix}\in\mathbb{C}^{m\times n}. \tag{5.25}$$

With the $\mathbf{V}$ given in equation (10.398), we know that $\|\boldsymbol{v}_i\|_2 = 1$ for $i \in \{1, ..., n\}$. Thus, each column of matrix $\mathbf{V}$ is a point on $\mathbb{S}^{d-1}$.

Let g denote a primitive root modulo n. We construct the index $\Lambda = \{k_1, k_2, ..., k_m\}$ as

$$\Lambda = \{g^0, g^{\frac{n-1}{m}}, g^{\frac{2(n-1)}{m}}, \cdots, g^{\frac{(m-1)(n-1)}{m}}\} \bmod n. \tag{5.26}$$

The set $\{g^0, g^{\frac{n-1}{m}}, g^{\frac{2(n-1)}{m}}, \cdots, g^{\frac{(m-1)(n-1)}{m}}\} \bmod n$ forms a subgroup of the the group $\{g^0, g^1, \cdots, g^{n-2}\} \bmod n$. Based on this, we derive upper bounds of the mutual coherence of the points set $\boldsymbol{V}$. The results are summarized in Theorem 32 and Theorem 16.

Theorem 15. *Suppose $d = 2m, N = 2n$, and n is a prime such that $m|(n-1)$. Construct matrix $\boldsymbol{V}$ as in Eq.(10.398) with index set Λ as Eq.(10.400). Let mutual coherence $\mu(\boldsymbol{V}) := \max_{i \neq j} \frac{|\boldsymbol{v}_i^\top \boldsymbol{v}_j|}{\|\boldsymbol{v}_i\|_2 \|\boldsymbol{v}_j\|_2}$. Then $\mu(\boldsymbol{V}) \leq \frac{\sqrt{n}}{m}$.*

Theorem 16. *Suppose $d = 2m, N = 2n$, and n is a prime such that $m|(n-1)$, and $m \leq n^{\frac{2}{3}}$. Construct matrix $\boldsymbol{V}$ as in Eq.(10.398) with index set Λ as Eq.(10.400). Let mutual coherence $\mu(\boldsymbol{V}) := \max_{i \neq j} \frac{|\boldsymbol{v}_i^\top \boldsymbol{v}_j|}{\|\boldsymbol{v}_i\|_2 \|\boldsymbol{v}_j\|_2}$. Then $\mu(\boldsymbol{V}) \leq C m^{-1/2} n^{1/6} \log^{1/6} m$, where C denotes a positive constant independent of m and n.*

Theorem 32 and Theorem 16 show that our construction can achieve a bounded mutual coherence. A smaller mutual coherence means that the points are more evenly spread on sphere $\mathbb{S}^{d-1}$.

Remark: Our construction does not require a restrictive constraint of the dimension of data. The only assumption of data dimension d is that d is a even number, i.e.,$2|d$, which is commonly satisfied in practice. Moreover, the product $\boldsymbol{V}^\top \boldsymbol{x}$ can be accelerated by fast Fourier transform as in [117].

Evaluation of the mutual coherence:

We evaluate our subgroup-based spherical QMC by comparing with the construction in [117] and i.i.d Gaussian sampling.

We set the dimension d as in $\{50, 100, 200, 500, 1000\}$. For each dimension d, we set the number of points $N = 2n$, with n as the first ten prime numbers such that $\frac{d}{2}$ divides $n-1$, i.e., $\frac{d}{2} | (n-1)$. Both subgroup-based QMC and Lyu's method are deterministic. For Gaussian sampling method, we report the mean $\pm$ standard deviation of mutual coherence over 50 independent runs. The mutual coherence for each dimension are reported in Table 5.3. The smaller the mutual coherence, the better.

We can observe that our subgroup-based spherical QMC achieves a competitive mutual coherence compared with Lyu's method in [117]. Note that our method does not require a time consuming optimization procedure, thus it is appealing for applications that demands a fast construction. Moreover, both our subgroup-based QMC and Lyu's method obtain a significant smaller coherence than i.i.d Gaussian sampling.

Table 5.3: Mutual coherence of points set constructed by different methods. Smaller is better.

d=50		202	302	502	802	1202	1402	1502	2102	2302	2402
	SubGroup	**0.1490**	**0.2289**	**0.1923**	0.2930	**0.2608**	0.3402	0.3358	0.3211	0.4534	**0.3353**
	Lyu [117]	0.2313	0.2377	0.2901	**0.2902**	0.3005	**0.3154**	**0.3155**	**0.3209**	**0.3595**	0.3718
	Gaussian	0.5400± 0.0254	0.5738± 0.0291	0.5904± 0.0257	0.6158± 0.0249	0.6270± 0.0209	0.6254± 0.0184	0.6328± 0.0219	0.6447± 0.0184	0.6520± 0.0204	0.6517± 0.0216
d=100		202	302	502	802	1202	1402	1502	2102	2302	2402
	SubGroup	**0.1105**	**0.1529**	0.1923	**0.1764**	0.2397	0.2749	0.2513	0.2679	0.4534	0.3353
	Lyu [117]	0.1234	0.1581	**0.1586**	0.1870	**0.2041**	**0.2191**	**0.1976**	**0.2047**	**0.2244**	**0.2218**
	Gaussian	0.4033± 0.0272	0.4210± 0.0274	0.4422± 0.0225	0.4577± 0.0230	0.4616± 0.0170	0.4734± 0.0174	0.4716± 0.0234	0.4878± 0.0167	0.4866± 0.0172	0.4947± 0.0192
d=200		202	802	1202	1402	2402	2602	3202	3602	3802	5602
	SubGroup	**0.0100**	0.1251	0.1835	0.1966	0.2365	0.1553	0.1910	0.1914	0.2529	0.2457
	Lyu [117]	**0.0100**	**0.1108**	**0.1223**	**0.1262**	**0.1417**	**0.1444**	**0.1505**	**0.1648**	**0.1624**	**0.1679**
	Gaussian	0.2887± 0.0163	0.3295± 0.0155	0.3362± 0.0148	0.3447± 0.0182	0.3564± 0.0140	0.3578± 0.0142	0.3645± 0.0143	0.3648± 0.0142	0.3689± 0.0140	0.3768± 0.0151
d=500		502	1502	4502	6002	6502	8002	9502	11002	14002	17002
	SubGroup	**0.0040**	0.0723	0.1051	0.1209	0.1107	0.1168	0.1199	0.1425	0.1587	0.1273
	Lyu [117]	**0.0040**	**0.0650**	**0.0946**	**0.0934**	**0.0930**	**0.1004**	**0.0980**	**0.1022**	**0.1077**	**0.1110**
	Gaussian	0.2040± 0.0111	0.2218± 0.0099	0.2388± 0.0092	0.2425± 0.0081	0.2448± 0.0113	0.2498± 0.0110	0.2528± 0.0100	0.2527± 0.0084	0.2579± 0.0113	0.2607± 0.0092
d=1000		6002	8002	11002	14002	17002	18002	21002	26002	32002	38002
	SubGroup	0.0754	0.0778	0.0819	0.0921	0.0935	0.0764	0.1065	0.0931	0.0908	0.1125
	Lyu [117]	**0.0594**	**0.0637**	**0.0662**	**0.0680**	**0.0684**	**0.0744**	**0.0774**	**0.0815**	**0.0781**	**0.0814**
	Gaussian	0.1736± 0.0067	0.1764± 0.0059	0.1797± 0.0060	0.1828± 0.0062	0.1846± 0.0052	0.1840± 0.0057	0.1869± 0.0052	0.1888± 0.0055	0.1909± 0.0067	0.1920± 0.0056

5.7 QMC for Generative models

Our subgroup rank-1 lattice can be used for generative models. Buchholz et al. [31] suggest using QMC for variational inference to maximize the evidence lower bound (ELBO). We present a new method by directly learning the inverse of the cumulative distribution function (CDF).

In variational autoencoder, the objective is the evidence lower bound (ELBO) [94] defined as

$$\mathcal{L}(x, \phi, \theta) = \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x|z)\right] - \mathrm{KL}\left[q_\phi(z|x)||p_\theta(z)\right]. \tag{5.27}$$

The ELBO consists of two terms, i.e., the reconstruction term $\mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x|z)\right]$ and the regularization term $\mathrm{KL}\left[q_\phi(z|x)||p_\theta(z)\right]$. The reconstruction term is learning

to fit, while the regularization term controls the distance between distribution $q_\phi(z|x)$ to the prior distribution $p_\theta(z)$.

The reconstruction term $\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$ can be reformulated as

$$\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] = \int_{\mathcal{Z}} q_\phi(z|x) \log p_\theta(x|z) \mathrm{d}\boldsymbol{z} \tag{5.28}$$

$$= \int_{[0,1]^d} \log p_\theta\left(x|\Phi^{-1}(\boldsymbol{\epsilon})\right) \mathrm{d}\boldsymbol{\epsilon}. \tag{5.29}$$

where $\Phi^{-1}(\cdot)$ denotes the inverse cumulative distribution function with respect to the density $q_\phi(z|x)$.

Eq.(5.29) provides an alternative training scheme, we directly learn the inverse of CDF $F(\boldsymbol{\epsilon}; x) = \Phi^{-1}(\boldsymbol{\epsilon})$ given x instead of the density $q_\phi(z|x)$. We parameterize $F(\boldsymbol{\epsilon}, x)$ as a neural network with input $\boldsymbol{\epsilon}$ and data x. The inverse of CDF function $F(\boldsymbol{\epsilon}, x)$ can be seen as an encoder of x for inference. It is worth noting that learning the inverse of CDF can bring more flexibility without the assumption of the distribution, e.g., Gaussian.

To ensure the distribution q close to the prior distribution $p(z)$, we can use other regularization terms instead of the KL-divergence for any implicit distribution q, e.g., the maximum mean discrepancy. Besides this, we can also use a discriminator-based adversarial loss similar to adversarial autoencoders [121]

$$\widetilde{L}(x, F, D) = \mathbb{E}_{p_\theta(\boldsymbol{z})}[\log(D(\boldsymbol{z}))] + \mathbb{E}_{p(\boldsymbol{\epsilon})}[\log(1 - D(F(\boldsymbol{\epsilon}, x)))], \tag{5.30}$$

where $p(\boldsymbol{\epsilon})$ denotes a uniform distribution on unit cube $[0, 1]^d$, D is the discriminator, F denotes the inverse of CDF mapping.

When the domain $\mathcal{Z}$ coincides with a target domain $\mathcal{Y}$, we can use an empirical data distribution Y as the prior. This leads to a training scheme similar to cycle GAN [189]. In contrast to cycle GAN, the encoder F depends on both data x in source domain and $\boldsymbol{\epsilon}$ in unit cube. The expectation term $\mathbb{E}_{p(\boldsymbol{\epsilon})}[\cdot]$ can be approximated by QMC methods.

5.8 Generative Inference for CycleGAN

We evaluate our subgroup rank-1 lattice on training generative model. As shown in section 5.7, we can learn the inverse CDF functions $F(\boldsymbol{\epsilon}, x)$ as a generator from domain $\mathcal{X}$ to domain $\mathcal{Y}$ in cycle GAN. We set $F(\boldsymbol{\epsilon}, x) = G_1(x) + G_2(\boldsymbol{\epsilon})$, where G_1 and G_2 denotes the neural networks. Network G_1 maps input image x to a target mean,

while network G_2 maps $\boldsymbol{\epsilon} \in [0,1]^d$ as the residue. Similarly, $\widetilde{F}(\widetilde{\boldsymbol{\epsilon}}, y) = \widetilde{G}_1(y) + \widetilde{G}_2(\widetilde{\boldsymbol{\epsilon}})$ denotes an generator from domain $\mathcal{Y}$ to domain $\mathcal{X}$.

We implement the model based on the open-sourced Pytorch code of [189]. All G_1, G_2, $\widetilde{G}_1$ and $\widetilde{G}_2$ employ the default ResNet architecture with 9 blocks in [189]. The input size of both $\boldsymbol{\epsilon}$ and $\widetilde{\boldsymbol{\epsilon}}$ are $d = 256 \times 256$. We keep all the hyperparameters same for all the methods as the default value in [189].

We compare our subgroup rank-1 lattice with Monte Carlo sampling for training the generative model. For subgroup rank-1 lattice, we set the number of points $n = 12d+1 = 786433$. We do not store all the points, instead we sample $i \in \{0, \cdots, n-1\}$ uniformly and construct $\boldsymbol{\epsilon}$ and $\widetilde{\boldsymbol{\epsilon}}$ based on Eq.(5.3) during the training process. For Monte Carlo sampling, $\boldsymbol{\epsilon}$ and $\widetilde{\boldsymbol{\epsilon}}$ are sampled from $Uniform[0,1]^d$.

We train generative models on the Vangogh2photo data set and maps data set employed in [189]. We present experimental results of the generated images from models trained with subgroup-based rank-1 lattice sampling, Monte-Carlo sampling, and standard version of CycleGAN. The experimental results on Vangogh2photo dataset and maps dataset are shown in Figure 5.5 and Figure 5.6, respectively. From Figure 5.5, we can observe that the images generated by the model trained with Monte-Carlo sampling have some blurred patches. This phenomenon may be because the additional flexibility of randomness makes the training more difficult to converge to a good model. In contrast, the model trained with subgroup-based rank-1 lattice sampling generates more clearer images. It may be because the rank-1 lattice sampling has finite possible choices, i.e., $n = 786433$ possible points in the experiments, which is much smaller than the case of Monte-Carlo uniform sampling. The rank-1 lattice sampling is more deterministic than Monte Carlo sampling, which alleviates the training difficulty to fit a good model. Since in our subgroup-based rank-1 lattice it is very simple to construct new samples, it can serve as a good alternative to Monte Carlo sampling for generative model training.

5.9 Summary

In this Chapter, we propose a closed-form method for rank-1 lattice construction, which is simple and efficient without exhaustive computer search. Theoretically, we prove that our subgroup rank-1 lattice has few different pairwise distance values, which is more regular to be evenly spaced. Moreover, we prove a lower and an upper bound for the minimum toroidal distance of a non-degenerate rank-1 lattice.

Figure 5.5: Illustration of the generated images from models trained with subgroup rank-1 lattice sampling, Monte-Carlo sampling, and Standard version of CycleGAN.

Empirically, our subgroup rank-1 lattice obtains near-optimal minimum toroidal distance compared with Korobov exhaustive search. Moreover, subgroup rank-1 lattice achieves smaller integration approximation error. In addition, we propose a closed-form method to generate QMC points set on sphere $\mathbb{S}^{d-1}$. We proved upper bounds of the mutual coherence of the generated points. Further, we show an example of CycleGAN training in the supplement. Our subgroup rank-1 lattice sampling and QMC on sphere can serve as an alternative for training generative models.

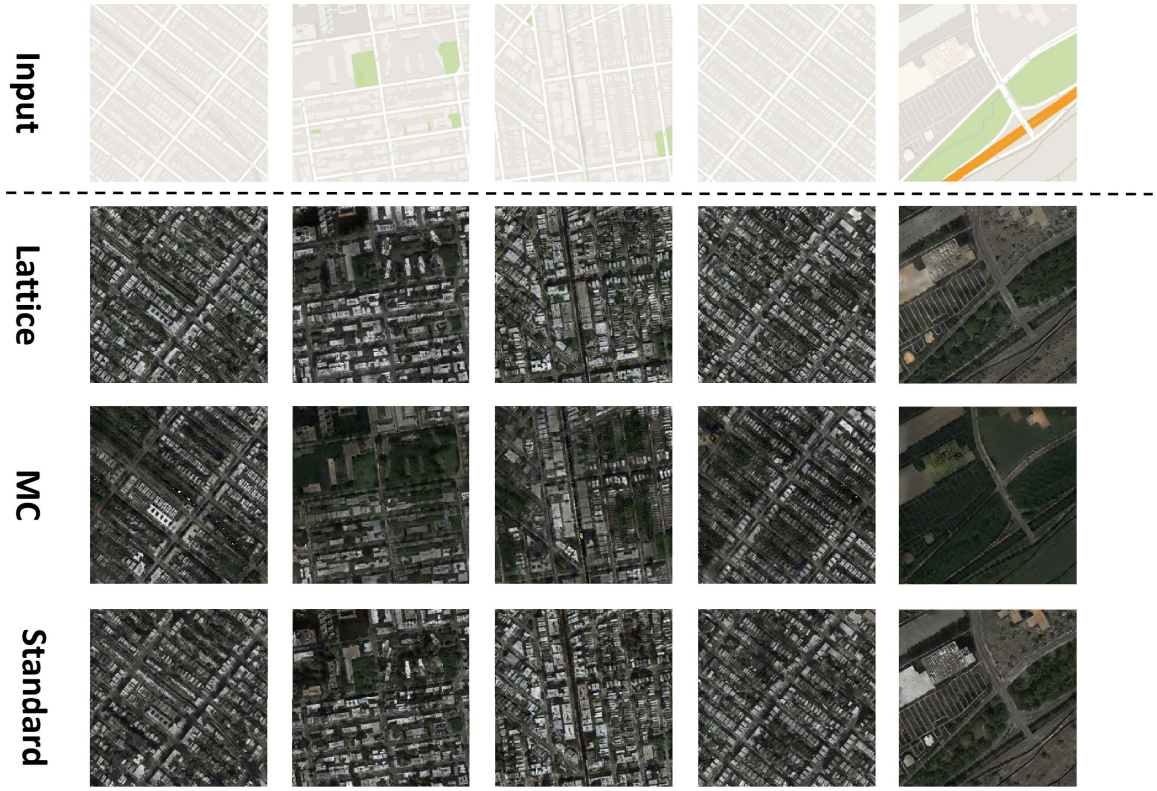


Figure 5.6: Illustration of the generated images from models trained with subgroup rank-1 lattice sampling, Monte-Carlo sampling, and Standard version of CycleGAN.

Chapter 6

Spherical Structured Feature Maps for Kernel Approximation

6.1 Chapter Abstract

In this chapter, we propose Spherical Structured Feature (SSF) maps to approximate shift and rotation invariant kernels as well as b^{th}-order arc-cosine kernels [34]. We construct SSF maps based on the point set on $d-1$ dimensional sphere $\mathbb{S}^{d-1}$. We prove that the inner product of SSF maps are unbiased estimates for above kernels if asymptotically uniformly distributed point set on $\mathbb{S}^{d-1}$ is given. According to [29], optimizing the discrete Riesz s-energy can generate asymptotically uniformly distributed point set on $\mathbb{S}^{d-1}$. Thus, we propose an efficient coordinate decent method to find a local optimum of the discrete Riesz s-energy for SSF maps construction. Theoretically, SSF maps construction achieves linear space complexity and loglinear time complexity. Empirically, SSF maps achieve superior performance compared with other methods.

6.2 Background of Kernel Approximation

We provide a brief review of random feature maps and the discrete Riesz s-energy in this section as preliminaries.

6.2.1 Random Feature Maps

Random feature maps can be viewed as equal weight approximation of multidimensional integrals. One earlier work [142] approximates the shift invariant kernels based on the Bochner's Theorem.

Theorem. *Bochner's Theorem ([148]) : A continuous shift invariant scaled kernel function* $\mathrm{K}(\mathbf{x},\mathbf{z}) = \mathbf{K}(\mathbf{x}-\mathbf{z}) : \mathbb{R}^{\mathbf{d}} \to \mathbb{C}$ *is positive definite if and only if it is the Fourier Transform of a unique finite probability measure* p *on* $\mathbb{R}^d$.

$$\mathrm{K}(\mathbf{x},\mathbf{z}) = \int_{\mathbb{R}^{\mathbf{d}}} \mathbf{e}^{-\mathbf{i}(\mathbf{x}-\mathbf{z})^{\mathbf{T}}\mathbf{w}} \mathbf{p}(\mathbf{w}) \mathbf{dw} \tag{6.1}$$

For a real valued kernel $\mathrm{K}(\mathbf{x},\mathbf{z})$, $p(\mathbf{w}) = p(-\mathbf{w}) \geq 0$ can ensure the imaginary parts of the integral vanish. According to the Bochner's theorem, there is a one-to-one correspondence between the kernel functions $\mathrm{K}(\mathbf{x},\mathbf{z})$ and probability densities $p(\mathbf{w})$ defined on $\mathbb{R}^d$.

Shift and rotation invariant kernels are shift invariant kernels with the rotation invariant property, i.e. $K(\mathbf{x},\mathbf{z}) = K(R\mathbf{x}, R\mathbf{z})$, given any rotation $R \in SO(d)$, where $SO(d)$ denotes rotation groups. The Gaussian kernel $K(\mathbf{x},\mathbf{z}) = e^{-\|\mathbf{x}-\mathbf{z}\|_2^2/2\sigma^2}$ is a member of this family. From Bochner's theorem, the corresponding probability density is also Gaussian. For a general Gaussian RBF kernel $K(\mathbf{x},\mathbf{z}) = e^{-(\mathbf{x}-\mathbf{z})^T\Sigma(\mathbf{x}-\mathbf{z})/2}$, it can be transformed into rotation invariant form by using $\mathbf{y} = \Sigma^{1/2}\mathbf{x}$ in the original domain.

$\mathbf{b^{th}}$-order arc-cosine kernels are rotation invariant kernels. As discussed in [34], b^{th}-order arc-cosine kernels have the following form:

$$K_b(\mathbf{x},\mathbf{z}) = \tfrac{1}{\pi} \|\mathbf{x}\|_2^b \|\mathbf{z}\|_2^b J_b(\theta) \tag{6.2}$$

where $\theta = \cos^{-1}\left(\frac{\mathbf{x}^T\mathbf{z}}{\|\mathbf{x}\|_2\|\mathbf{z}\|_2}\right)$

b^{th}-order arc-cosine kernels have trivial dependence on the norm of $\mathbf{x}$ and $\mathbf{z}$. The dependence on the angle is defined by function $J_b(\theta)$. b^{th}-order arc-cosine kernels are rotation invariant kernels but not shift invariant kernels in general. For example, the zero-order (6.3) and first-order (6.4) arc-cosine kernel are not shift invariant kernels.

$$K_0(\mathbf{x},\mathbf{z}) = 1 - \tfrac{\theta}{\pi} \tag{6.3}$$

$$K_1(\mathbf{x},\mathbf{z}) = \tfrac{1}{\pi} \|\mathbf{x}\|_2 \|\mathbf{z}\|_2 (\sin\theta + (\pi-\theta)\cos\theta) \tag{6.4}$$

The b^{th}-order arc-cosine kernel $\mathrm{K}_b(\mathbf{x},\mathbf{z})$ can be reformulated via the integral representation:

$$\mathrm{K}_b(\mathbf{x},\mathbf{z}) = 2\int_{\mathbb{R}^d} s(\mathbf{w}^T\mathbf{x}) s(\mathbf{w}^T\mathbf{z}) (\mathbf{w}^T\mathbf{x})^b (\mathbf{w}^T\mathbf{z})^b p(\mathbf{w}) d\mathbf{w} \tag{6.5}$$

where $s(\cdot)$ is a step function (i.e. $s(x) = 1$ if $x > 0$ and 0 otherwise) and the density p is standard Gaussian.

Feature maps: Both Monte Carlo and Quasi-Monte Carlo approximation [47] are equal weight approximation to integrals. Based on equal weight approximation, the feature maps can be constructed as:

$$K(\mathbf{x},\mathbf{z}) \approx \frac{1}{N}\sum_{i=1}^{N} f\left(\mathbf{w}_i^T\mathbf{x}\right) f\left(\mathbf{w}_i^T\mathbf{x}\right) = \Psi(\mathbf{x})^T\Psi(\mathbf{z}) \tag{6.6}$$

where $\mathbf{w}_i, i \in 1, ..., N$ are samples constructed by Monte Carlo or Quasi-Monte Carlo methods. $f(\cdot)$ is a nonlinear function depending on the kernel. $\Psi(\cdot)$ is the explicit finite dimensional feature map. For Gaussian kernel with bandwidth σ, the associated nonlinear function is a complex exponential function $f(x) = e^{ix/\sigma}$. For a zero-order arc-cosine kernel in (6.3) and first-order arc-cosine kernel in (6.4), the associated nonlinear functions are step function $f(x) = s(x)$ and ReLU activation function $f(x) = max(0, x)$ respectively.

6.2.2 Discrete Riesz s-energy

The discrete Riesz s-energy is related to the equal weight numerical integration and uniformly distributed point set.

Equal weight numerical integration over a d-dimensional sphere $\mathbb{S}^d := \{\mathbf{x} \in \mathbb{R}^{d+1} \mid \|\mathbf{x}\|_2 = 1\}$ uses equal weight summation of finite point evaluations of the integrands to approximate the integrals:

$$\int_{\mathbb{S}^d} f(\mathbf{v}) d\sigma(\mathbf{v}) \approx \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{v}_i) \tag{6.7}$$

where σ denotes the normalized surface area measure on $\mathbb{S}^d$.

According to [29], the point set $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_N] \in \mathbb{S}^{d\times N}$ is asymptotically uniformly distributed if equation (10.539) holds true.

$$\lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{v}_i) = \int_{\mathbb{S}^d} f(\mathbf{v}) d\sigma(\mathbf{v}) \tag{6.8}$$

The discrete Riesz s-energy [29, 63] is defined as equation (10.542):

$$E_s(\mathbf{V}) := \begin{cases} \sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} \frac{1}{\|\mathbf{v}_i - \mathbf{v}_j\|_2^s}, & s \neq 0 \\ \sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} \log \frac{1}{\|\mathbf{v}_i - \mathbf{v}_j\|_2}, & s = 0 \end{cases} \tag{6.9}$$

Theorem. *([29]): For $s > -2$, the optimum N-point configuration of the Riesz s-energy on $\mathbb{S}^d$ is asymptotically uniformly distributed w.r.t the normalized surface area measure σ on $\mathbb{S}^d$.*

According to [28, 29], the discrete Riesz s-energy can serve as a criterion to construct the point set $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_N] \in \mathbb{S}^{d\times N}$ for QMC designs. Particularly, [28] have proved that maximizing the discrete Riesz s-energy with $s \in (-2, 0)$ can generate QMC designs for functions in Sobolev space. They also prove that QMC designs have higher convergence rate of worst-case error than fully randomly chosen points for functions in Sobolev space.

6.3 Spherical Structured Feature Maps

In this section, we propose SSF maps to approximate shift and rotation invariant kernels as well as b^{th}-order arc-cosine kernels by employing their rotation invariant property.

6.3.1 Feature Maps for Shift and Rotation Invariant Kernels

Shift and rotation invariant kernels are highly symmetric and structured because they satisfy both shift invariant property and rotation invariant property. Rotation invariant property means that $K(\mathbf{x}, \mathbf{z}) = K(R\mathbf{x}, R\mathbf{z})$, given any rotation $R \in SO(d)$, where $SO(d)$ denotes rotation groups. To benefit from rotation invariant property, it is reasonable to construct the feature maps by using spherical equal weight approximation in equation (10.537) and (10.539).

The feature maps for real valued shift and rotation invariant kernels $K(\mathbf{x}, \mathbf{z})$ can be constructed as equation (6.10):

$$\begin{aligned}\Psi(\mathbf{x}) = \tfrac{1}{\sqrt{NM}}[&\cos\left(\Phi^{-}(t_1)\mathbf{x}^T\mathbf{v}_1\right), \sin\left(\Phi^{-}(t_1)\mathbf{x}^T\mathbf{v}_1\right), \\ &..., \cos\left(\Phi^{-}(t_M)\mathbf{x}^T\mathbf{v}_N\right), \sin\left(\Phi^{-}(t_M)\mathbf{x}^T\mathbf{v}_N\right)]^T\end{aligned} \tag{6.10}$$

where $t_j = \frac{j}{M+1}$, $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_N] \in \mathbb{S}^{d-1\times N}$ denotes the point set asymptotically uniformly distributed on $\mathbb{S}^{d-1}$and $\Phi^{-}(x)$ denotes the inverse cumulative distribution function w.r.t the nonnegative radial scale.

Theorem 17. $\Psi(\mathbf{x})^T\Psi(\mathbf{z})$ *is an unbiased estimate of a real valued shift and rotation invariant kennel* $\mathrm{K}(\mathbf{x}, \mathbf{z})$.

Proof: From Bochner's Theorem, a shift invariant kernel $K(\mathbf{x}, \mathbf{z})$ can be written as equation (6.1). Let $r = \|\mathbf{w}\|_2$and $p(r)$ be the density function of r. Because of the rotation invariant property of $K(\mathbf{x}, \mathbf{z})$, we achieve equation (6.11).

$$\begin{aligned}\mathrm{K}(\mathbf{x}, \mathbf{z}) &= \int_{\mathbb{R}_+}\int_{\mathbb{S}^{d-1}} e^{-ir(\mathbf{x}-\mathbf{z})^T\mathbf{v}}p(r)dr d\sigma(\mathbf{v}) \\ &= \int_{[0,1]}\int_{\mathbb{S}^{d-1}} e^{-i\,\Phi^{-}(t)(\mathbf{x}-\mathbf{z})^T\mathbf{v}}d\sigma(\mathbf{v})dt\end{aligned} \tag{6.11}$$

where $\mathbb{R}_+$ denotes the nonnegative real values.

For real valued kernel $K(\mathbf{x}, \mathbf{z})$, the imaginary parts of the integral vanish. We can achieve equation (6.12).

$$\mathrm{K}(\mathbf{x}, \mathbf{z}) = \int_{[0,1]} \int_{\mathbb{S}^{d-1}} \cos\left(\Phi^-(t)(\mathbf{x}-\mathbf{z})^T\mathbf{v}\right) d\sigma(\mathbf{v}) dt \tag{6.12}$$

According to the property of asymptotically uniformly distributed point set $\mathbf{V}$ in equation (10.539) and the one-dimensional QMC rule, we obtain equation (6.13).

$$\begin{aligned}
&\lim_{M,N\to\infty} \Psi(\mathbf{x})^T \Psi(\mathbf{z}) = \\
&\lim_{M,N\to\infty} \tfrac{1}{MN} \sum_{i=1}^{N} \sum_{j=1}^{M} (\cos\left(\Phi^-(t_j)\mathbf{x}^T\mathbf{v}_i\right) \cos\left(\Phi^-(t_j)\mathbf{z}^T\mathbf{v}_i\right) \\
&\qquad + \sin\left(\Phi^-(t_j)\mathbf{x}^T\mathbf{v}_i\right) \sin\left(\Phi^-(t_j)\mathbf{z}^T\mathbf{v}_i\right)) \\
&= \lim_{M,N\to\infty} \tfrac{1}{MN} \sum_{j=1}^{M} \sum_{i=1}^{N} \cos\left(\Phi^-(t_j)(\mathbf{x}-\mathbf{z})^T\mathbf{v}_i\right) \\
&= \int_{[0,1]} \int_{\mathbb{S}^{d-1}} \cos\left(\Phi^-(t)(\mathbf{x}-\mathbf{z})^T\mathbf{v}\right) d\sigma(\mathbf{v}) dt \\
&= K(\mathbf{x}, \mathbf{z})
\end{aligned} \tag{6.13}$$

□

Proposition 1. *: Let $\mathbf{U} = [\mathbf{V}, -\mathbf{V}]$, using point set $\mathbf{U}$ to approximate a real valued shift and rotation invariant kernel $K(\mathbf{x}, \mathbf{z})$ by using equation (6.10) is equal to using point set $\mathbf{V}$ to approximate $K(\mathbf{x}, \mathbf{z})$:*

$$\Psi(\mathbf{x}; \mathbf{U})^T \Psi(\mathbf{z}; \mathbf{U}) = \Psi(\mathbf{x}; \mathbf{V})^T \Psi(\mathbf{z}; \mathbf{V}) \tag{6.14}$$

Proof: Note that cosine function is an even function. Thus, we obtain equation (6.15).

$$\cos\left(\Phi^-(t_j)(\mathbf{x}-\mathbf{z})^T\mathbf{v}_i\right) = \cos\left(-\Phi^-(t_j)(\mathbf{x}-\mathbf{z})^T\mathbf{v}_i\right) \tag{6.15}$$

Thus, we achieve equation (6.16).

$$\begin{aligned}
&\Psi(\mathbf{x}; \mathbf{U})^T \Psi(\mathbf{z}; \mathbf{U}) \\
&= \tfrac{1}{2NM} \sum_{i=1}^{N} \sum_{j=1}^{M} \cos\left(\Phi^-(t_j)(\mathbf{x}-\mathbf{z})^T\mathbf{v}_i\right) \\
&+ \tfrac{1}{2NM} \sum_{i=1}^{N} \sum_{j=1}^{M} \cos\left(-\Phi^-(t_j)(\mathbf{x}-\mathbf{z})^T\mathbf{v}_i\right) \\
&= \tfrac{1}{2NM} \sum_{i=1}^{N} \sum_{j=1}^{M} 2\cos\left(\Phi^-(t_j)(\mathbf{x}-\mathbf{z})^T\mathbf{v}_i\right) \\
&= \Psi(\mathbf{x}; \mathbf{V})^T \Psi(\mathbf{z}; \mathbf{V})
\end{aligned} \tag{6.16}$$

□

Proposition 1 shows that for a shift and rotation invariant kernel, computing N points can achieve the same approximation effect compared with using $2N$ points.

6.3.2 Feature Maps for b$^{\text{th}}$-order Arc-cosine Kernels

In this subsection, we discuss the feature maps for b^{th}-order arc-cosine kernels. We discuss them separately because they are rotation invariant kernels but not shift invariant kernels in general. Moreover, they are closely related to deep neural networks [34], which demonstrate super performance in many areas.

Lemma 2. *The b^{th}-order arc-cosine kernels can be calculated as equation (6.17).*

$$\begin{aligned} \mathrm{K}_b(\mathbf{x},\mathbf{z}) = C_b \int_{\mathbb{S}^{d-1}} \chi\left(\mathbf{v}^T\mathbf{x}\right) \chi\left(\mathbf{v}^T\mathbf{z}\right) \\ +\chi(-\mathbf{v}^T\mathbf{x})\chi(-\mathbf{v}^T\mathbf{z}) d\sigma(\mathbf{v}) \end{aligned} \tag{6.17}$$

where $\chi(x) = \max(0, sign(x)|x|^b)$, $C_b = \int_{\mathbb{R}_+} r^{2b} p(r) dr$. C_b is a constant that is independent of $\mathbf{x}$ and $\mathbf{z}$. $p(r)$ is the density function of the chi-distribution with d degrees freedom. For example, the constants associated with the zero, first and second-order arc-cosine kernels are $C_0 = 1$, $C_1 = d$ and $C_2 = d(d+2)$ respectively.

Proof: From equation (6.5), we can achieve equation (6.18).

$$\begin{aligned} \mathrm{K}_b(\mathbf{x},\mathbf{z}) &= 2\int_{\mathbb{R}^d} s(\mathbf{w}^T\mathbf{x}) s(\mathbf{w}^T\mathbf{z}) (\mathbf{w}^T\mathbf{x})^b (\mathbf{w}^T\mathbf{z})^b p(\mathbf{w}) d\mathbf{w} \\ &= 2\int_{\mathbb{R}^d} \chi\left(\mathbf{w}^T\mathbf{x}\right) \chi\left(\mathbf{w}^T\mathbf{z}\right) p(\mathbf{w}) d\mathbf{w} \end{aligned} \tag{6.18}$$

Let $r = \|\mathbf{w}\|_2$. Since p is standard Gaussian, by taking rotation invariant property, we obtain equation (6.19).

$$\begin{aligned} \mathrm{K}_b(\mathbf{x},\mathbf{z}) &= \mathbf{2}\int_{\mathbb{R}^{\mathbf{d}}} \chi\left(\mathbf{w}^{\mathbf{T}}\mathbf{x}\right) \chi\left(\mathbf{w}^{\mathbf{T}}\mathbf{z}\right) \mathbf{p}(\mathbf{w}) \mathbf{dw} \\ &= 2\int_{\mathbb{S}^{d-1}} \int_{\mathbb{R}_+} \chi\left(r^b\mathbf{v}^T\mathbf{x}\right) \chi\left(r^b\mathbf{v}^T\mathbf{z}\right) p(r) d\sigma(\mathbf{v}) dr \\ &= 2\int_{\mathbb{S}^{d-1}} \int_{\mathbb{R}_+} r^{2b} \chi\left(\mathbf{v}^T\mathbf{x}\right) \chi\left(\mathbf{v}^T\mathbf{z}\right) p(r) d\sigma(\mathbf{v}) dr \\ &= 2\int_{\mathbb{R}_+} r^{2b} p(r) dr \int_{\mathbb{S}^{d-1}} \chi\left(\mathbf{v}^T\mathbf{x}\right) \chi\left(\mathbf{v}^T\mathbf{z}\right) d\sigma(\mathbf{v}) \\ &= 2C_b \int_{\mathbb{S}^{d-1}} \chi\left(\mathbf{v}^T\mathbf{x}\right) \chi\left(\mathbf{v}^T\mathbf{z}\right) d\sigma(\mathbf{v}) \end{aligned} \tag{6.19}$$

Since $\mathrm{K}_b(\mathbf{x},\mathbf{z})$ is rotation invariant, we have $\mathrm{K}_b(\mathbf{x},\mathbf{z}) = \mathrm{K}_b(-\mathbf{x},-\mathbf{z})$. Together with equation (6.19), we achieve equation (6.20).

$$\begin{aligned} \mathrm{K}_b(\mathbf{x},\mathbf{z}) = C_b \int_{\mathbb{S}^{d-1}} \chi\left(\mathbf{v}^T\mathbf{x}\right) \chi\left(\mathbf{v}^T\mathbf{z}\right) \\ +\chi(-\mathbf{v}^T\mathbf{x})\chi(-\mathbf{v}^T\mathbf{z}) d\sigma(\mathbf{v}) \end{aligned} \tag{6.20}$$

□

The feature maps for a b^{th}-order arc-cosine kernel $\mathrm{K}_b(\mathbf{x},\mathbf{z})$ can be constructed as equation (6.21).

$$\begin{aligned} \Psi(\mathbf{x}) = \sqrt{\tfrac{C_b}{N}} [\chi\left(\mathbf{v}_1^T\mathbf{x}\right), \chi\left(-\mathbf{v}_1^T\mathbf{x}\right), ..., \\ \chi\left(\mathbf{v}_N^T\mathbf{x}\right), \chi\left(-\mathbf{v}_N^T\mathbf{x}\right)]^T \in \mathbb{R}^{2N} \end{aligned} \tag{6.21}$$

Theorem 18. *$\Psi(\mathbf{x})^T \Psi(\mathbf{z})$ is an unbiased estimate of a b^{th}-order arc-cosine kernel $\mathrm{K}_b(\mathbf{x},\mathbf{z})$.*

Proof: According to the Lemma 3.1 and the property of the asymptotically uniformly distributed point set $\mathbf{V}$, we obtain equation (6.22).

$$\begin{aligned}
&\lim_{N\to\infty} \Psi(\mathbf{x})^T \Psi(\mathbf{z}) \\
&= \lim_{N\to\infty} \frac{C_b}{N} \sum_{i=1}^{N} \chi\left(\mathbf{v}_i^T\mathbf{x}\right)\chi\left(\mathbf{v}_i^T\mathbf{z}\right) + \chi(-\mathbf{v}_i^T\mathbf{x})\chi(-\mathbf{v}_i^T\mathbf{z}) \\
&= C_b \int_{S^{d-1}} \chi\left(\mathbf{v}^T\mathbf{x}\right)\chi\left(\mathbf{v}^T\mathbf{z}\right) + \chi(-\mathbf{v}^T\mathbf{x})\chi(-\mathbf{v}^T\mathbf{z}) d\sigma(\mathbf{v}) \\
&= \mathrm{K}_b(\mathbf{x},\mathbf{z})
\end{aligned} \tag{6.22}$$

□

From equation (6.17) and (6.22), we observe that the approximation is actually operated on the $(d-1)$-dimensional domain instead of d-dimensional domain [34]. Generally, the approximation error of Quasi Monte Carlo methods with N points depends on the dimension of integration. A lower dimension leads to smaller approximation error, thus the feature maps in equation (6.21) can achieve lower approximation error.

The feature maps in equation (6.21) are closely related to the bidirectional activation neural network. Specifically, the feature maps for the first-order arc-cosine kernel are related to the bidirectional ReLU activation function [8] which has the distance preservation property compared with ReLU.

From equation (6.14) and (6.21), we know that the feature maps actually rely on the point set $\mathbf{U} = [\mathbf{V}, -\mathbf{V}]$. The design of the point set $\mathbf{U}$ will be discussed in section 6.4.

6.4 Design of Matrix U

We have discussed the construction of SSF maps in last section. However, one unsolved problem is how to obtain the matrix $\mathbf{U} = [\mathbf{V}, -\mathbf{V}]$. We employ the discrete Riesz s-energy as the objective function to obtain matrix $\mathbf{U}$ because it can generate asymptotically uniformly distributed points on $\mathbb{S}^{d-1}$ [29]. Moreover, to achieve computation and storage efficiency for feature maps construction , we add a structured constraint to the matrix $\mathbf{U}$. In this section, we show the structure of matrix $\mathbf{U}$ first and then the optimization of discrete Riesz s-energy.

It is worth noting that matrix $\mathbf{U}$ can be used not only for kernel approximation, but also for approximation of general integrals over hypersphere. Moreover, by using FFT, matrix $\mathbf{U}$ can accelerate the integral approximation which involves projection operations. In addition, it only needs to store the indexes with linear storage cost (i.e. $O(d)$) instead of to explicitly store the matrix with cost $O(Nd)$.

6.4.1 Structure of Matrix U

Since $\mathbf{U}$ can be constructed by $\mathbf{V}$, i.e. $\mathbf{U} = [\mathbf{V}, -\mathbf{V}]$, we only need to define structured matrix $\mathbf{V}$. To achieve loglinear time complexity of SSF maps construction, we construct $\mathbf{V}$ by extracting rows from a discrete Fourier matrix. The complexity analysis of SSF maps construction based on matrix $\mathbf{V}$ is given in section 6.5.

Mathematically, the construction of matrix $\mathbf{V}$ is shown as follows. Without loss of generality, we assume that $d = 2m, N = 2n$, $m < n$. Let $F \in \mathbb{C}^{n\times n}$ be a $n \times n$ discrete Fourier matrix. $F_{k,j} = e^{\frac{2\pi ikj}{n}}$ is the $(k,j)^{th}$entry of F, where $i = \sqrt{-1}$. Let $\Lambda = [k_1, k_2, ..., k_m] \subset \{1, ..., n-1\}$ be a subset of indexes.

The structured matrix $\mathbf{V}$ can be defined as equation (10.398).

$$\mathbf{V} = \frac{\mathbf{1}}{\sqrt{\mathbf{m}}} \begin{bmatrix} \mathrm{Re}F_\Lambda & -\mathrm{Im}F_\Lambda \\ \mathrm{Im}F_\Lambda & \mathrm{Re}F_\Lambda \end{bmatrix} \in \mathbb{R}^{\mathbf{d}\times\mathbf{N}} \tag{6.23}$$

where F_Λ in equation (10.399) is the matrix constructed by m rows of F.

$$F_\Lambda = \begin{bmatrix} e^{\frac{2\pi ik_1 1}{n}} & \cdots & e^{\frac{2\pi ik_1 n}{n}} \\ \vdots & \ddots & \vdots \\ e^{\frac{2\pi ik_m 1}{n}} & \cdots & e^{\frac{2\pi ik_m n}{n}} \end{bmatrix} \in \mathbb{C}^{m\times n} \tag{6.24}$$

With the $\mathbf{V}$ given in equation (10.398), it is easy to verify that $\|\mathbf{v}_i\|_2 = 1$ for $i \in \{1, ..., n\}$. Thus, each column of matrix $\mathbf{V}$ is a point on $\mathbb{S}^{d-1}$.

6.4.2 Minimize the Discrete Riesz s-energy

With structured matrix $\mathbf{V}$ defined in equation (10.398), our goal is to select a subset of indexes Λ that optimizes the discrete Riesz s-energy. Specifically, we will discuss how to minimize the Riesz 0-energy in equation (6.25). The other Riesz s-energy can be optimized in a similar way.

$$E(\mathbf{U}) = \sum_{i=1}^{2N} \sum_{j=1, j\neq i}^{2N} \log \frac{1}{\|\mathbf{u}_i - \mathbf{u}_j\|} \tag{6.25}$$

where $\mathbf{U} = [\mathbf{V}, -\mathbf{V}] = [\mathbf{u}_1, ..., \mathbf{u}_{2N}]$.

In the following, we will discuss how to minimize equation (6.25) by using a coordinate decent method.

Theorem 19. *Let* $\mathbf{U} = [\mathbf{V}, -\mathbf{V}]$ *with* $\mathbf{V}$ *defined in (10.398), the discrete Riesz 0-energy of* $\mathbf{U}$ *can be calculated as equation (6.26).*

$$\begin{aligned} E(\mathbf{U}) = C - 2n \sum_{p=1}^{n-1} \log \left(1 - (\mathrm{Im}\frac{1}{m} \sum_{s=1}^{m} e^{\frac{2\pi ik_s p}{n}})^2\right) \\ -2n \sum_{p=1}^{n-1} \log \left(1 - (\mathrm{Re}\frac{1}{m} \sum_{s=1}^{m} e^{\frac{2\pi ik_s p}{n}})^2\right) \end{aligned} \tag{6.26}$$

where C is a constant independent of the choice of Λ.

Proof: Since $\mathbf{U} = [\mathbf{V}, -\mathbf{V}] \in \mathbb{S}^{(\mathbf{d-1})\times\mathbf{2N}}$, we obtain equation (6.27).

$$\begin{aligned} E(\mathbf{U}) &= -\sum_{\mathbf{i=1}}^{\mathbf{2N}} \sum_{\mathbf{j=1,j\neq i}}^{\mathbf{2N}} \log \|\mathbf{u_i} - \mathbf{u_j}\| \\ &= -2\sum_{i=1}^{N} \log \|2\mathbf{v}_i\| \\ &-2\sum_{i=1}^{N} \sum_{j=1,j\neq i}^{N} \left(\log \|\mathbf{v}_i - \mathbf{v}_j\| + \log \|\mathbf{v}_i + \mathbf{v}_j\|\right) \\ &= C - 2\sum_{i=1}^{N} \sum_{j=1,j\neq i}^{N} \log \left(\|\mathbf{v}_i - \mathbf{v}_j\| \, \|\mathbf{v}_i + \mathbf{v}_j\|\right) \\ &= C - 2\sum_{i=1}^{N} \sum_{j=1,j\neq i}^{N} \log \left(\sqrt{2 - 2\mathbf{v}_i^T\mathbf{v}_j}\sqrt{2 + 2\mathbf{v}_i^T\mathbf{v}_j}\right) \end{aligned} \tag{6.27}$$

Recall that $N = 2n$. By separating the summation term into two parts (each part has $n \times n$ term), we achieve equation (6.28).

$$\begin{aligned} E(\mathbf{U}) &= \mathbf{C} - \mathbf{2}\sum_{\mathbf{i=1}}^{\mathbf{2n}} \sum_{\mathbf{j=1,j\neq i}}^{\mathbf{2n}} \log \left(\mathbf{2}\sqrt{\mathbf{1} - (\mathbf{v_i^T v_j})^{\mathbf{2}}}\right) \\ &= C - 4\sum_{i=1}^{n} \sum_{j=n+1}^{2n} \log \left(2\sqrt{1 - (\mathbf{v}_i^T\mathbf{v}_j)^2}\right) \\ &-4\sum_{i=1}^{n} \sum_{j=1,j\neq i}^{n} \log \left(2\sqrt{1 - (\mathbf{v}_i^T\mathbf{v}_j)^2}\right) \end{aligned} \tag{6.28}$$

Let $\mathbf{V}_{\cdot,\mathbf{1:n}} = [\mathbf{v_1}, ..., \mathbf{v_n}]$ and $\mathbf{V}_{\cdot,\mathbf{n+1:2n}} = [\mathbf{v_{n+1}}, ..., \mathbf{v_{2n}}]$ be the matrix consisting of the first n and last n columns of $\mathbf{V}$ respectively. We can obtain equation (6.29).

$$\mathbf{V}_{\cdot,\mathbf{1:n}}^{\mathbf{T}}\mathbf{V}_{\cdot,\mathbf{n+1:2n}} = \tfrac{\mathbf{1}}{\mathbf{m}}\mathrm{Re}\mathbf{F}_{\mathbf{\Lambda}}{}^{\mathbf{T}}(-\mathrm{Im}\mathbf{F}_{\mathbf{\Lambda}}) + \tfrac{\mathbf{1}}{\mathbf{m}}(\mathrm{Im}\mathbf{F}_{\mathbf{\Lambda}})^{\mathbf{T}}\mathrm{Re}\mathbf{F}_{\mathbf{\Lambda}} \tag{6.29}$$

Note that all diagonal elements of $\mathbf{V}_{\cdot,\mathbf{1:n}}^{\mathbf{T}}\mathbf{V}_{\cdot,\mathbf{n+1:2n}}$ are zero. By further separating the first summation term of equation (6.28) into two parts, we obtain equation (6.30).

$$\begin{aligned} E(\mathbf{U}) &= \mathbf{C} - \mathbf{4}\sum_{\mathbf{i=1}}^{\mathbf{n}} \sum_{\mathbf{j=n+i}}^{\mathbf{2n}} \log \left(\mathbf{2}\sqrt{\mathbf{1-0}}\right) \\ &-4\sum_{i=1}^{n} \sum_{j=n+1,j\neq n+i}^{2n} \log \left(2\sqrt{1 - (\mathbf{v}_i^T\mathbf{v}_j)^2}\right) \\ &-4\sum_{i=1}^{n} \sum_{j=1,j\neq i}^{n} \log \left(2\sqrt{1 - (\mathbf{v}_i^T\mathbf{v}_j)^2}\right) \\ &= C - 4\sum_{i=1}^{n} \sum_{j=n+1,j\neq n+i}^{2n} \log \left(2\sqrt{1 - (\mathbf{v}_i^T\mathbf{v}_j)^2}\right) \\ &-4\sum_{i=1}^{n} \sum_{j=1,j\neq i}^{n} \log \left(2\sqrt{1 - (\mathbf{v}_i^T\mathbf{v}_j)^2}\right) \end{aligned} \tag{6.30}$$

To be concise, let $\mathbf{Z} = [\mathbf{z_1}, , ..., \mathbf{z_n}] = \frac{\mathbf{1}}{\sqrt{\mathbf{m}}}\mathbf{F_\Lambda}$.

For $1 \leq j \leq n, j \neq i$, we achieve equation (6.31).

$$(\mathbf{v_i^T v_j})^\mathbf{2} = (\mathrm{Re}\mathbf{z_i^* z_j})^\mathbf{2} = \left(\frac{1}{m}\mathrm{Re}\sum_{s=1}^{m} e^{2\pi i k_s p/n}\right)^2 \tag{6.31}$$

For $n+1 \leq j \leq 2n, , j \neq n+i$, we attain equation (6.32).

$$(\mathbf{v_i^T v_j})^\mathbf{2} = (\mathrm{Im}\mathbf{z_i^* z_{j-n}})^\mathbf{2} = \left(\frac{1}{m}\mathrm{Im}\sum_{s=1}^{m} e^{2\pi i k_s p/n}\right)^2 \tag{6.32}$$

In equation (6.31) and (6.32), $p \equiv i - j \pmod{n}$, where mod denotes the modulus operation on integers.

Note that $\mathbf{z_i^* z_j}$ has at most $n-1$ distinct values when $i \neq j \pmod{n}$. Together with equation (6.30), we achieve equation (6.33).

$$\begin{aligned}
&E(\mathbf{U}) = \mathbf{C} - \mathbf{4}\sum_{\mathbf{i=1}}^{\mathbf{n}}\sum_{\mathbf{j=n+1, j\neq n+i}}^{\mathbf{2n}} \log\left(\mathbf{2}\sqrt{\mathbf{1} - (\mathbf{v_i^T v_j})^\mathbf{2}}\right) \\
&-4\sum_{i=1}^{n}\sum_{j=1, j\neq i}^{n} \log\left(2\sqrt{1 - (\mathbf{v_i^T v_j})^2}\right) \\
&= C - 4\sum_{i=1}^{n}\sum_{j=n+1, j\neq n+i}^{2n} \log\left(2\sqrt{1 - (\mathrm{Im}\mathbf{z_i^* z_{j-n}})^2}\right) \\
&-4\sum_{i=1}^{n}\sum_{j=1, j\neq i}^{n} \log\left(2\sqrt{1 - (\mathrm{Re}\mathbf{z_i^* z_j})^2}\right) \\
&= C - 4n\sum_{p=1}^{n-1} \log\left(2\sqrt{1 - (\mathrm{Im}\frac{1}{m}\sum_{s=1}^{m} e^{2\pi i k_s p/n})^2}\right) \\
&-4n\sum_{p=1}^{n-1} \log\left(2\sqrt{1 - (\mathrm{Re}\frac{1}{m}\sum_{s=1}^{m} e^{2\pi i k_s p/n})^2}\right) \\
&= C - 2n\sum_{p=1}^{n-1} \log\left(1 - (\mathrm{Im}\frac{1}{m}\sum_{s=1}^{m} e^{2\pi i k_s p/n})^2\right) \\
&-2n\sum_{p=1}^{n-1} \log\left(1 - (\mathrm{Re}\frac{1}{m}\sum_{s=1}^{m} e^{2\pi i k_s p/n})^2\right)
\end{aligned} \tag{6.33}$$

□

From Theorem 4.1, we know that minimizing $E(\mathbf{U})$ is equivalent to maximizing $J(\Lambda)$ which is defined in equation (6.34).

$$\begin{aligned}
J(\Lambda) &= \sum_{p=1}^{n-1} \log\left(1 - (\mathrm{Im}\frac{1}{m}\sum_{s=1}^{m} e^{2\pi i k_s p/n})^2\right) \\
&+ \sum_{p=1}^{n-1} \log\left(1 - (\mathrm{Re}\frac{1}{m}\sum_{s=1}^{m} e^{2\pi i k_s p/n})^2\right)
\end{aligned} \tag{6.34}$$

Algorithm 12 Coordinate Index Selection

Initialization: random sample $\Lambda = [k_1, k_2, ..., k_m]$ from $\{1, 2, ...n-1\}$ without replacement. Set $\widetilde{\mathbf{h}} = \mathbf{1}^T F_\Lambda$

repeat

Set $J = J(\Lambda)$

for $q = 1$ **to** m **do**

Set $\mathbf{g} = [e^{2\pi i k_q/n}, e^{2\pi i k_q 2/n}..., e^{2\pi i k_q(n-1)/n}]$

Set $\mathbf{h} = \widetilde{\mathbf{h}} - \mathbf{g}$

Find k_q^* by $k_q^* = \underset{k_q \in \{1,...,n-1\}}{\arg\max} J(k_q)$ in (6.35)

Update $\mathbf{g} = [e^{2\pi i k_q^*/n}, e^{2\pi i k_q^* 2/n}..., e^{2\pi i k_q^*(n-1)/n}]$

Set $\widetilde{\mathbf{h}} = \mathbf{h} + \mathbf{g}$

end for

until J does not change

By keeping all the indexes in $\Lambda = [k_1, k_2, ..., k_m]$ fixed except the q^{th} element, we can obtain equation (6.35).

$$\begin{aligned} J(k_q) = &\sum_{p=1}^{n-1} \log\left(1 - \left(\mathrm{Im}\left(h_p + e^{2\pi i k_q p/n}\right)/m\right)^2\right) \\ &+ \sum_{p=1}^{n-1} \log\left(1 - \left(\mathrm{Re}\left(h_p + e^{2\pi i k_q p/n}\right)/m\right)^2\right) \end{aligned} \tag{6.35}$$

where $k_q \in \{1, 2, ...n-1\}$, $h_p = \sum_{s=1, s\neq q}^{m} e^{2\pi i k_s p/n}$.

With equation (6.35), we can maximize $J(\Lambda)$ by maximizing $J(k_q)$ with other indexes fixed each time. Let $\mathbf{h} = [h_1, ..., h_{n-1}]$, $\mathbf{g} = [e^{2\pi i k_q/n}, e^{2\pi i k_q 2/n}..., e^{2\pi i k_q(n-1)/n}]$. $\mathbf{1} = [1, ..., 1]^T \in \mathbb{R}^m$ is the vector of all ones. A coordinate ascent method to maximize $J(\Lambda)$ is given in Algorithm 12.

Obviously, it is a discrete optimization problem. Algorithm 12 can find a local optimum. The time complexity of the Algorithm 12 is $O(Tmn^2)$, where T denotes the number of outer iteration. Empirically, the outer iteration T is less than ten.

6.5 Fast Feature Maps Construction

In this section, we will discuss how to construct SSF maps in loglinear time complexity and linear space complexity by using the structure property of $\mathbf{V}$.

Theorem 20. *Assume that* $d = 2m, N = 2n$, $m < n$. *Let* $\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} \in \mathbb{R}^{\mathbf{2m}}$ *and* $\mathbf{z} = \mathbf{x_1} + \mathbf{i x_2} \in \mathbb{C}^{\mathbf{m}}$. *Given* $\Lambda = [k_1, k_2, ..., k_m] \subset \{1, ..., n-1\}$, *let* $\mathbf{y} \in \mathbb{C}^{\mathbf{n}}$ *with*

$\mathbf{y_\Lambda} = \mathbf{z}$. *Other elements outside the index set* Λ *are equal to zero. Given* $\mathbf{V}$ *defined in equation (10.398), equation (6.36) holds.*

$$\mathbf{V}^T\mathbf{x} = \frac{1}{\sqrt{m}}[\mathrm{Re}(F^*\mathbf{y}), \mathrm{Im}(F^*\mathbf{y})]^T \tag{6.36}$$

Proof: Let $\Omega \in \mathbb{R}^{n\times n}$ be a diagonal matrix with all diagonal elements inside the index set Λ equal to one , the others equal to zero.

$$\begin{aligned}
\mathbf{V}^T\mathbf{x} &= \frac{1}{\sqrt{m}}\begin{bmatrix} \mathrm{Re}F_\Lambda & -\mathrm{Im}F_\Lambda \\ \mathrm{Im}F_\Lambda & \mathrm{Re}F_\Lambda \end{bmatrix}^T \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} \\
&= \frac{1}{\sqrt{m}}\begin{bmatrix} (\mathrm{Re}{F_\Lambda}^T)\mathbf{x}_1 + (\mathrm{Im}{F_\Lambda}^T)\mathbf{x}_2 \\ (-\mathrm{Im}{F_\Lambda}^T)\mathbf{x}_1 + (\mathrm{Re}{F_\Lambda}^T)\mathbf{x}_2 \end{bmatrix} \\
&= \frac{1}{\sqrt{m}}\begin{bmatrix} \mathrm{Re}(F_\Lambda^*\mathbf{z}) \\ \mathrm{Im}(F_\Lambda^*\mathbf{z}) \end{bmatrix} \\
&= \frac{1}{\sqrt{m}}\begin{bmatrix} \mathrm{Re}(F^*\Omega\mathbf{y}) \\ \mathrm{Im}(F^*\Omega\mathbf{y}) \end{bmatrix} \\
&= \frac{1}{\sqrt{m}}\begin{bmatrix} \mathrm{Re}(F^*\mathbf{y}) \\ \mathrm{Im}(F^*\mathbf{y}) \end{bmatrix}
\end{aligned} \tag{6.37}$$

□

Thus, the projection operation $\mathbf{V}^T\mathbf{x}$ (previously mentioned in equation (6.10) and (6.21)) can be calculated by Fast Fourier Transform algorithm (FFT) in $O(n\log n)$ time complexity. Because scaling and taking nonlinear transform can be finished in $O(n)$, the total time complexity to construct SSF maps is $O(n\log n)$.

All steps to construct SSF maps are summarized as follows:

(a) Compute $\widetilde{\mathbf{x}}$ by $\widetilde{\mathbf{x}} = \mathbf{D}\mathbf{x}$, where $\mathbf{D} \in \{-\mathbf{1}, +\mathbf{1}\}^{\mathbf{d}\times\mathbf{d}}$ is a diagonal matrix where diagonal elements are uniformly sampled from $\{-1, +1\}$.

(b) Construct $\mathbf{y}$ such that $\mathbf{y_\Lambda} = \widetilde{\mathbf{x}}_\mathbf{1} + \mathbf{i}\widetilde{\mathbf{x}}_\mathbf{2}$, other elements outside the index set Λ are equal to zero.

(c) Compute $\mathbf{V}^T\widetilde{\mathbf{x}}$ by equation (6.36) via FFT.

(d) Construct feature maps $\Psi(\mathbf{x})$ via equation (6.10) or (6.21).

For each (m, n) pair , the index set Λ only need to be computed once. It takes $O(m)$ space to store Λ. For shift and rotation invariant kernels, it takes $O(M)$ space to store $\Phi^-(t_j), j \in 1, ..., M$ and takes $O(d)$ $(d = 2m)$ space to store Λ and $\mathbf{D}$. For b^{th}-order arc-cosine kernels, it only needs to store one parameter C_b and takes $O(d)$ space to store Λ and $\mathbf{D}$. By setting $M \leq d$, the total space complexity to store the projection matrix is $O(d)$.

6.6 Empirical Studies

We compare SSF maps with feature maps obtained by fully Gaussian [34,142], the Circulant [39] matrices, QMC with Halton set and QMC with Sobol set [14]. For Halton set and Sobol set, the implementation in MATLAB are employed in the experiments. The scrambling and shifting techniques are used for Haltonset and Sobolset. In all the experiments, we fix $M = 1$ (the number of one-dimensional QMC points) for SSF maps.

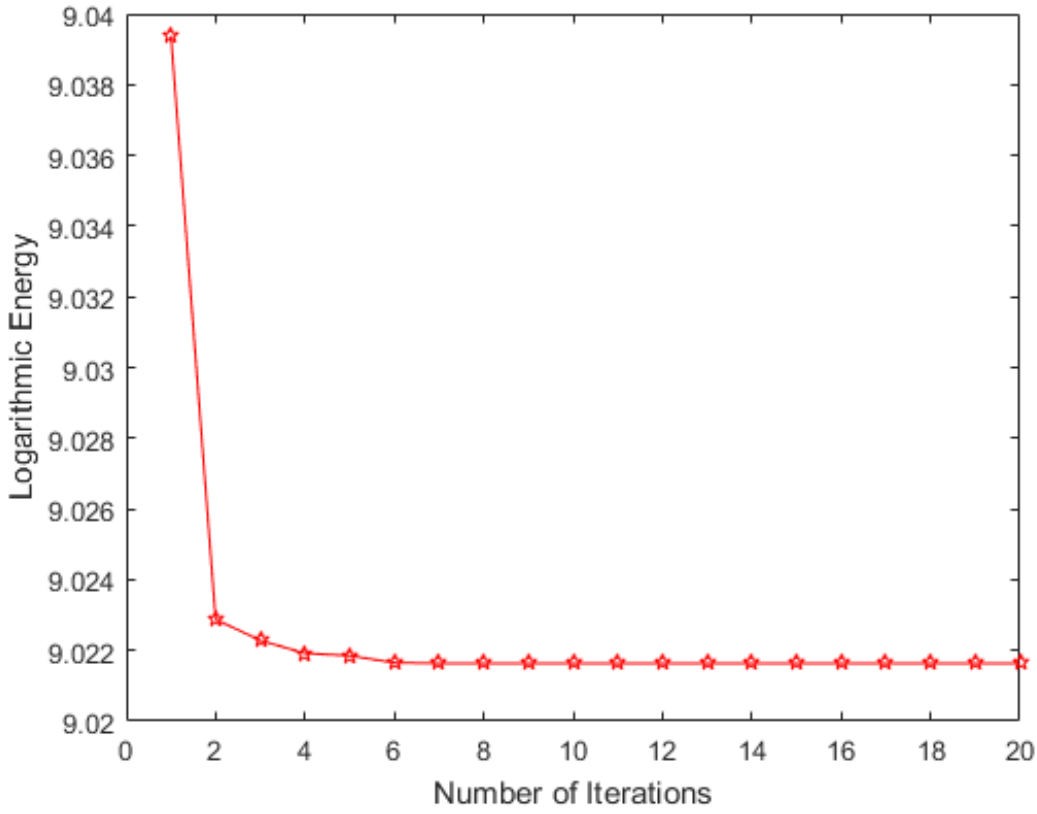


Figure 6.1: Convergence of the Logarithmic Energy

6.6.1 Convergence and Speedup

First, the convergence of the logarithmic energy ($-J(\Lambda)$ in equation (6.34)) with $(m, n) = (160, 1600)$ is shown in Figure 6.1. From Figure 6.1, we find that it takes less than ten iterations (i.e. $T < 10$) for Algorithm 12 to find a local optimum.

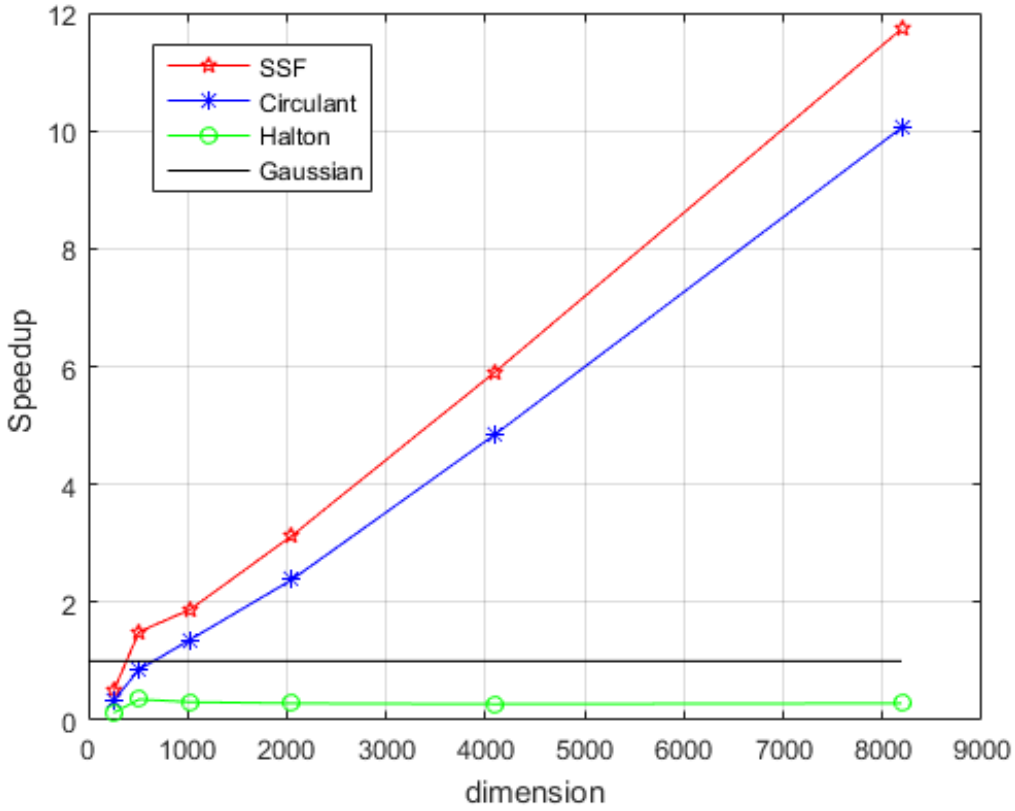
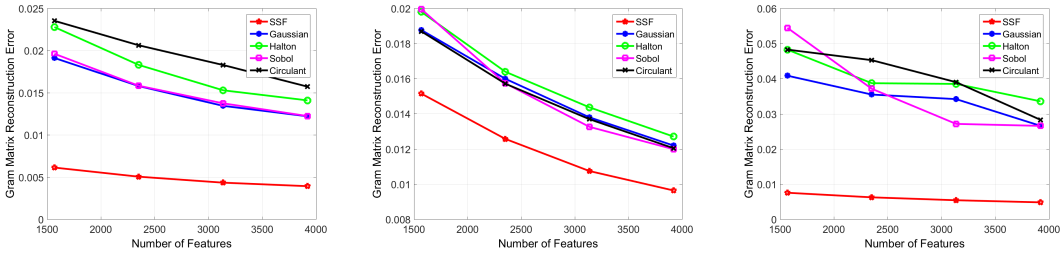


Figure 6.2: Speedup of the Feature Maps Construction

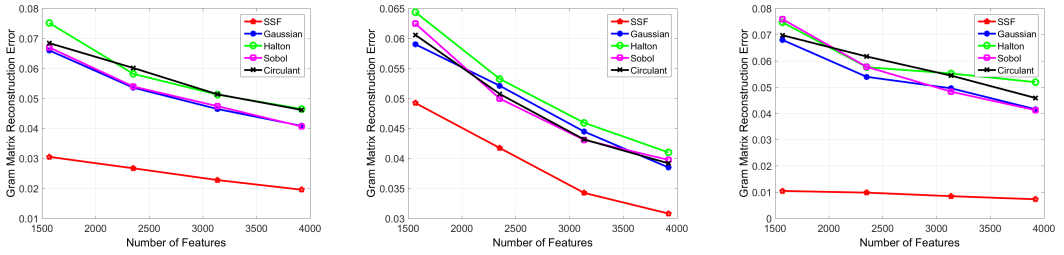
Second, the speedup results of all methods are shown in Figure 6.2. We set $N = 2d$ for all the methods. The speedup of fully Gaussian projection is the baseline. We can observe that the speedup of QMC with Halton set is constant as the dimension d increases and is slower than the baseline. The speedup of both SSF maps and the Circulant increase fast as dimension increases, which is consistent with theoretical analysis. The speedup of Sobol set is not shown because the inbuilt Sobolset routine of MATLAB does not support dimension larger than 1,111.

6.6.2 Approximation Accuracy

We evaluate reconstruction error of Gaussian kernel, zero-order arc-cosine kernel and first-order arc-cosine kernel on CIFAR10 [99], MNIST [107], usps and dna dataset. MNIST is a handwritten digit image dataset, which contains 70,000 samples with 784-dimensional features(pixel). For CIFAR10 with 60,000 samples, the 320-dimensional gist feature [60] are employed in the experiments. Both the relative Frobenius er-



(a) $\frac{\|\widetilde{K}-K\|_F}{\|K\|_F}$ for Gaussian Kernel (b) $\frac{\|\widetilde{K}-K\|_F}{\|K\|_F}$ for Zero-order Arc Kernel (c) $\frac{\|\widetilde{K}-K\|_F}{\|K\|_F}$ for First-order Arc Kernel



(d) $\frac{\|\widetilde{K}-K\|_\infty}{\|K\|_\infty}$ for Gaussian Kernel (e) $\frac{\|\widetilde{K}-K\|_\infty}{\|K\|_\infty}$ for Zero-order Arc Kernel (f) $\frac{\|\widetilde{K}-K\|_\infty}{\|K\|_\infty}$ for First-order Arc Kernel

Figure 6.3: Relative Mean and Max Reconstruction Error for Gaussian, Zero-order and First-order Arc-cosine Kernel on MNIST

ror (i.e. $\frac{\|\widetilde{K}-K\|_F}{\|K\|_F}$) and the relative element-wise maximum error (i.e. $\frac{\|\widetilde{K}-K\|_\infty}{\|K\|_\infty}$) are evaluated, where K and $\widetilde{K}$ denote the exact and approximated Gram matrices respectively. The Frobenius norm and the elementwise maximum norm are defined as $\|X\|_F = \sqrt{\sum_i \sum_j |X_{ij}|^2}$ and $\|X\|_\infty = \max_{i,j} |X_{ij}|$ respectively.

The reconstruction error in the experiments is the mean value over 10 independent runs. The dimensions of the feature maps are set to $\{2 \times d, 3 \times d, 4 \times d, 5 \times d\}$, where d is the dimension of the data. For MNIST and CIFAR10 dataset, each run randomly select 2,000 samples to construct the Gram matrix. The mean value of the reconstruction errors with different norms on MNIST are shown in Figure 6.3. Results on the other datasets are similar to that of Figure 6.3. One can refer to the supplementary material for results on other datasets.

Figure 6.3 shows that the feature maps obtained with fully Gaussian matrix, the Circulant matrix, QMC with Halton set and QMC with Sobol set have similar reconstruction error. SSF maps have the smallest approximation error among five methods. Especially for the first-order arc-cosine kernel, it achieves nearly one-fifth relative mean error and one-seventh relative max error of other methods. Moreover, even if $M = 1$, SSF maps can achieve about one-third relative mean error and half of the relative max error of other methods for Gaussian Kernel approximation.

6.7 Summary

In this Chapter, we propose Spherical Structured Feature (SSF) maps to approximate shift and rotation invariant kernels as well as b^{th}-order arc-cosine kernels. SSF maps can achieve computation and storage efficiency as well as better approximation accuracy.

Chapter 7

Neural Optimization Kernel: Towards Robust Deep Learning

7.1 Chapter Abstract

Deep neural networks (NN) have achieved great success in many applications. However, why do deep neural networks obtain good generalization at an over-parameterization regime is still unclear. To better understand deep NN, in this chapter, we establish the connection between deep NN and a novel kernel family, i.e., Neural Optimization Kernel (NOK), from an approximation perspective. The architecture of structured approximation of NOK performs monotonic descent updates of implicit regularization problems. We can implicitly choose the regularization problems by employing different activation functions, e.g., ReLU, max pooling, and soft-thresholding. We further establish a new generalization bound of our deep structured approximated NOK architecture. Our unsupervised structured approximated NOK block can serve as a simple plug-in of popular backbones for a good generalization against input noise.

7.2 Neural Optimization Kernel

Denote $\mathcal{L}_2$ as the Gaussian square-integrable functional space, i.e., $\mathcal{L}_2 := \{f \big| \mathbb{E}_{\boldsymbol{w}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I}_d)}[f(\boldsymbol{w})^2] < \infty\}$, and denote $\overline{\mathcal{L}}_2$ as the spherically square-integrable function space, i.e., $\overline{\mathcal{L}}_2 := \{f \big| \mathbb{E}_{\boldsymbol{w}\sim Uni[\sqrt{d}\mathbb{S}^{d-1}]}[f(\boldsymbol{w})^2] < \infty\}$.

For functional $f \in \mathcal{F}$, where $\mathcal{F} = \mathcal{L}_2$ or $\mathcal{F} = \overline{\mathcal{L}}_2$, define function $k(\cdot,\cdot) : \mathcal{X} \times \mathcal{X} \to \mathcal{R}$ as

$$k(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w}, \boldsymbol{x}) f(\boldsymbol{w}, \boldsymbol{y})]. \tag{7.1}$$

Then, we know $k(\cdot,\cdot)$ is a bounded kernel, which is shown in Proposition 1. All detailed proofs are given in Appendix.

Table 7.1: Regularizers and Proximal Operators

	l_0-norm [51]	l_1-norm [50]	MCP [186]
$\phi_\lambda(z)$	$\lambda\|z\|_0$	$\lambda\|z\|_1$	$\lambda\int_0^{\|z\|} max(0, 1 - x/(\gamma\lambda))\, dx$
$h(z)$	$h(z)=\begin{cases} z, & \|z\| \geq \sqrt{2\lambda} \\ 0, & \|z\| < \sqrt{2\lambda} \end{cases}$.	$h(z)=\text{sign}(z)\max(0, \|z\|-\lambda)$	$h(z)=\begin{cases} z, & \|z\| > \gamma\lambda \\ \frac{sign(z)(\|z\|-\lambda)}{1-1/\gamma}, & \lambda < \|z\| \leq \gamma\lambda \\ 0, & \|z\| \leq \lambda \end{cases}$.
	Capped l_1-norm [187]	SCAD [55]	MCP0 [136]
$\phi_\lambda(z)$	$\lambda\min(\|z\|, \gamma)$	$\lambda\int_0^{\|z\|} min(1, \frac{max(0, \gamma\lambda - z)}{(\gamma-1)\lambda})\, dx\,, (\gamma > 2)$	$\phi_\lambda(z)=\frac{1}{2}(\lambda - max(\sqrt{\lambda} - \|z\|, 0)^2)$
$h(z)$	$h(z)=\begin{cases} x_1, & q(x_1) \leq q(x_2) \\ x_2, & q(x_1) > q(x_2) \end{cases}$, *where* $\begin{matrix} x_1 = sign(z)\max(\|z\|, \gamma) \\ x_2 = sign(z)\min(\gamma, \max(0, \|z\| - \lambda)) \\ q(x) = 0.5(x - z)^2 + \lambda\min(\|x\|, \gamma) \end{matrix}$	$h(z)=\begin{cases} z, & \|z\| > \gamma\lambda \\ \frac{(\gamma-1)z - sign(z)\gamma\lambda}{\gamma-2}, & 2\lambda < \|z\| \leq \gamma\lambda \\ sign(z)\max(\|z\| - \lambda, 0), & \|z\| \leq 2\lambda \end{cases}$	$h(z)=\begin{cases} z, & \|z\| > \sqrt{\lambda} \\ \beta\sqrt{\lambda} \quad (\|\beta\| \leq 1), & \|z\| = \sqrt{\lambda} \\ 0, & \|z\| < \sqrt{\lambda} \end{cases}$.

Proposition 1. *For* $\forall f \in \mathcal{F}$ *(* $\mathcal{F} = \mathcal{L}_2$ *or* $\mathcal{F} = \overline{\mathcal{L}}_2$*), define function* $k(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w}, \boldsymbol{x})f(\boldsymbol{w}, \boldsymbol{y})] : \mathcal{X} \times \mathcal{X} \to \mathcal{R}$*, then* $k(\boldsymbol{x}, \boldsymbol{y})$ *is a bounded kernel, i.e.,* $k(\boldsymbol{x}, \boldsymbol{y}) = k(\boldsymbol{y}, \boldsymbol{x}) < \infty$ *and* $k(\boldsymbol{x}, \boldsymbol{y})$ *is positive definite.*

For functional $f \in \mathcal{F}$, where $\mathcal{F} = \mathcal{L}_2$ or $\mathcal{F} = \overline{\mathcal{L}}_2$, define operator $\mathcal{A}(\cdot) : \mathcal{F} \to \mathcal{R}^d$ as $\mathcal{A}(f) := \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]$. Define operator $\mathcal{A}^* : \mathcal{R}^d \to \mathcal{F}$ as $\mathcal{A}^*(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x}$, $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ or $\boldsymbol{w} \sim Uni[\sqrt{d}\mathbb{S}^{d-1}]$. We know $\mathcal{A} \circ \mathcal{A}^*(\cdot) = \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\boldsymbol{w}^\top] = \boldsymbol{I}_d : \mathcal{R}^d \to \mathcal{R}^d$. Details are provided in Appendix. Define operator $\Phi_\lambda(\cdot) : \mathcal{F} \to \mathcal{R}$ as $\Phi_\lambda(f) := \mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f(\boldsymbol{w}))]$, where $\phi_\lambda(\cdot)$ is a function with parameter λ and bounded from below, and $\mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f(\boldsymbol{w}))]$ exists for some $f \in \mathcal{F}$. Several examples of ϕ_λ and the corresponding proximal operators are shown in Table 7.1. It is worth noting that $\phi_\lambda(\cdot)$ can be either convex or non-convex.

Our Neural Optimization Kernel (NOK) is defined upon the solution of optimization problems. Before giving our Neural Optimization Kernel (NOK) definition, we first introduce a family of functional optimization problems. The Φ_λ-regularized optimization problem is defined as

$$\min_{f \in \mathcal{F}} \frac{1}{2}\|\boldsymbol{x} - \mathcal{A}(f)\|_2^2 + \Phi_\lambda(f) = \frac{1}{2}\|\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2 + \mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f(\boldsymbol{w}))], \tag{7.2}$$

where $\mathcal{F} = \mathcal{L}_2$ or $\mathcal{F} = \overline{\mathcal{L}}_2$. $f(\cdot) := f(\cdot, \boldsymbol{x})$ is a function specified by $\boldsymbol{x}$. We simplify the notation $f(\boldsymbol{w}, \boldsymbol{x})$ as $f(\boldsymbol{w})$ when the dependence of $\boldsymbol{x}$ is clear from the context.

For $\phi_\lambda(\cdot)$ with efficient proximal operators $h(\cdot)$ defined as $h(z) = \arg\min_x \frac{1}{2}(x - z)^2 + \phi_\lambda(x)$, we can optimize the problem (7.2) by iterative updating with Eq.(7.3):

$$f_{t+1}(\cdot) = h\big(\mathcal{A}^*(\boldsymbol{x}) + f_t(\cdot) - \mathcal{A}^* \circ \mathcal{A}(f_t(\cdot))\big). \tag{7.3}$$

The initialization is $f_0(\cdot) = 0$.

Remark: In the update rule (7.3), the term $-\mathcal{A}^* \circ \mathcal{A}(f_t(\cdot))$ can be viewed as a two-layer transformed residual modular of $f_t(\cdot)$. Then adding a skip connection $f_t(\cdot)$ and a biased term $\mathcal{A}^*(\boldsymbol{x})$. As shown in [4, 5], a ResNet-type architecture (residual

modular with skip connections) is crucial for obtaining a small error with sample and time efficiency.

For both convex and non-convex function ϕ_λ, our update rule in Eq.(7.3) leads to a monotonic descent.

Theorem 21. *(Monotonic Descent) For a function $\phi_\lambda(\cdot)$, denote $h(\cdot)$ as the proximal operator of $\phi_\lambda(\cdot)$. Suppose $|h(x)| \leq c|x|$ (or $|h(x)| \leq c$), $0 < c < \infty$ (e.g., hard thresholding function). Given a bouned $\boldsymbol{x} \in \mathcal{R}^d$, set function $f_{t+1}(\cdot) = h\big(\mathcal{A}^*(\boldsymbol{x}) + f_t(\cdot) - \mathcal{A}^* \circ \mathcal{A}(f_t(\cdot))\big)$ and $f_0 \in \mathcal{F}$ (e.g., $f_0 = 0$). Denote $Q(f) = \frac{1}{2}\|\boldsymbol{x} - \mathcal{A}(f)\|_2^2 + \Phi_\lambda(f)$. For $\forall t \geq 0$, we have*

$$Q(f_{t+1}) \leq Q(f_t) - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}) - \boldsymbol{w}^\top \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}))])^2] \leq Q(f_t). \tag{7.4}$$

Remark: Assumption$|h(x)| \leq c|x|$ (or $|h(x)| \leq c$) is used to ensure that each $f_t \in \mathcal{F}$. Neural networks with a activation function $h(\cdot)$, e.g., sigmoid, tanh, and ReLU, as long as $h(\cdot)$ satisfies the above assumption, it corresponds to a (implicit) $\phi(\cdot)$-regularized problem. Theorem 21 shows that a T-layer network performs T-steps monotonic descent updates of the $\phi(\cdot)$-regularized objective $Q(\cdot)$.

For a convex ϕ_λ, we can achieve a $O(\frac{1}{T})$ convergence rate, which is formally shown in Theorem 22.

Theorem 22. *For a convex function $\phi_\lambda(\cdot)$, denote $h(\cdot)$ as the proximal operator of $\phi_\lambda(\cdot)$. Suppose $|h(x)| \leq c|x|$ (or $|h(x)| \leq c$), $0 < c < \infty$. Given a bouned $\boldsymbol{x} \in \mathcal{R}^d$, set function $f_{t+1}(\cdot) = h\big(\mathcal{A}^*(\boldsymbol{x}) + f_t(\cdot) - \mathcal{A}^* \circ \mathcal{A}(f_t(\cdot))\big)$ and $f_0 \in \mathcal{F}$ (e.g., $f_0 = 0$). Denote $Q(f) = \frac{1}{2}\|\boldsymbol{x} - \mathcal{A}(f)\|_2^2 + \Phi_\lambda(f)$ and $f_* \in \mathcal{F}$ as an optimal of $Q(\cdot)$. For $\forall T \geq 1$, we have*

$$\begin{aligned} T\big(Q(f_T) - Q(f_*)\big) \leq & \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_0(\boldsymbol{w}) - f_*(\boldsymbol{w}))^2] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_T(\boldsymbol{w}) - f_*(\boldsymbol{w}))^2] \\ & - \frac{1}{2}\sum_{t=0}^{T-1} \|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w}) - f_*(\boldsymbol{w})\big)]\|_2^2 \\ & - \frac{1}{2}\sum_{t=0}^{T-1} (t+1)\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}))^2]. \end{aligned} \tag{7.5}$$

Remark: A ReLU $h(z) = \max(z, 0)$ corresponds a $\phi_\lambda(z) = \begin{cases} 0, & z \geq 0 \\ +\infty, & z < 0 \end{cases}$ (a lower semi-continuous convex function) , which results in a convex regularization problem. A T-layer NN obtains $O(1/T)$ convergence rate , which is faster than non-convex cases. This explains the success of ReLU on training deep NN from a NN

architecture optimization perspective. When ReLU and $f_0 = 0$ is used, the resultant first-layer kernel is the arc-cosine kernel in [35]. More interestingly, when ReLU is used (related to indicator function $\phi_\lambda(\cdot)$), the learned functional representation $p(\boldsymbol{w})f_t(\boldsymbol{w})$ is an unnormalized non-negative measure, where $p(\boldsymbol{w})$ denotes the density of Gaussian or Uniform sphere surface distribution. We can achieve a probability measure representation by normalizing $p(\boldsymbol{w})f_t(\boldsymbol{w})$ with $Z = \int p(\boldsymbol{w})f_t(\boldsymbol{w})\, d\boldsymbol{w}$.

Our **Neural Optimization Kernel (NOK)** is defined upon the optimized functional f_T (T-layer) as

$$k_{T,\infty}(\boldsymbol{x}, \boldsymbol{y}) := \mathbb{E}_{\boldsymbol{w}}[f_T(\boldsymbol{w}, \boldsymbol{x})f_T(\boldsymbol{w}, \boldsymbol{y})]. \tag{7.6}$$

With $f_0 \in \mathcal{F}$, we know $f_T \in \mathcal{F}$. From the Proposition 1, we know $k_{T,\infty}$ is a bounded kernel.

7.3 Structured Approximation

The orthogonal sampling [181] and spherically structured sampling [117, 119] have been successfully used for Gaussian and spherical integral approximation. In the QMC area, randomization of structured points set is standard and widely used to achieve an unbiased estimator (the same marginal distribution $p(\boldsymbol{w})$). In the hypercube domain $[0, 1]^d$, a uniformly distributed vector shift is employed. In the hypersphere domain $\mathbb{S}^{d-1}$, a uniformly random rotation is used. For the purpose of acceleration, [117] employs a diagonal random rotation matrix to approximate the full matrix rotation, which results in a $O(d)$ rotation time complexity instead of $O(d^3)$ complexity in computing SVD of random Gaussian matrix (full rotation). When the goal is to reduce approximation error, we can use the standard full matrix random orthogonal rotation of the structured points [117] as an unbiased estimator of integral on $Uni[\mathbb{S}^{d-1}]$. Moreover, we propose a new diagonal rotation method that maintains the $O(n\log n)$ time complexity and $O(d)$ space complexity by FFT as [117], which may of independent interest for integral approximation.

For all-layer trainable networks, we propose a data-dependent structured approximation as

$$\boldsymbol{W} = \sqrt{d}\boldsymbol{R}^\top \boldsymbol{B} \in \mathcal{R}^{d\times N}, \tag{7.7}$$

where $\boldsymbol{R}^\top \boldsymbol{R} = \boldsymbol{R}\boldsymbol{R}^\top = \boldsymbol{I}_d$ is a trainable orthogonal matrix parameter, N denotes the number of samples, and structured matrix $\boldsymbol{B}$ can either be a concatenate of random orthogonal matrices [181], or be the structured matrix in [117, 119].

Define operator $\widehat{\mathcal{A}} := \frac{1}{N}\boldsymbol{W} \ : \mathcal{R}^N \to \mathcal{R}^d$ and $\widehat{\mathcal{A}}^* := \boldsymbol{W}^\top \ : \mathcal{R}^d \to \mathcal{R}^N$. Operator $\widehat{\mathcal{A}}$ is an approximation of $\mathcal{A}$ by taking expectation over finite samples. Remarkably, by using our structured approximation, we have $\widehat{\mathcal{A}} \circ \widehat{\mathcal{A}}^* = \frac{1}{N}\boldsymbol{W}\boldsymbol{W}^\top = \boldsymbol{I}_d$.

Remark: The orthogonal property of the operator $\mathcal{A} \circ \mathcal{A}^* = \boldsymbol{I}_d$ is vitally important to achieve $O(\frac{1}{T})$ convergence rate with our update rule. It leads to a ResNet-type network architecture, which enables a stable gradient flow for training. When approximation with finite samples, standard Monte Carlo sampling does not maintain the orthogonal property, which degenerates the convergence. In contrast, our structured approximation preserves the second order moment $\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top] = \boldsymbol{I}_d$. Namely, our approximation maintains the orthogonal property, i.e., $\widehat{\mathcal{A}} \circ \widehat{\mathcal{A}}^* = \boldsymbol{I}_d$. With the orthogonal property, we can obtain the same convergence rate (w.r.t. the approximation objective) with our update rule. Moreover, for a k-sparse constrained problem, we prove the strictly monotonic descent property of our structured approximation when using $\boldsymbol{B}$ in [117, 119].

7.3.1 Convergence Rate for Finite Dimensional Approximation Problem

The finite approximation of problem (7.2) is given as

$$\widehat{Q}(\boldsymbol{y}) := \frac{1}{2}\|\boldsymbol{x} - \widehat{\mathcal{A}}(\boldsymbol{y})\|_2^2 + \frac{1}{N}\phi_\lambda(\boldsymbol{y}) = \frac{1}{2}\|\boldsymbol{x} - \frac{1}{N}\boldsymbol{W}\boldsymbol{y}\|_2^2 + \frac{1}{N}\phi_\lambda(\boldsymbol{y}), \tag{7.8}$$

where $\boldsymbol{y} \in \mathcal{R}^N$ and $\phi_\lambda(\boldsymbol{y}) := \sum_{i=1}^N \phi_\lambda(y_i)$.

The finite dimension update rule is given as :

$$\boldsymbol{y}_{t+1} = h\big(\boldsymbol{W}^\top\boldsymbol{x} + (\boldsymbol{I} - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{W})\boldsymbol{y}_t\big). \tag{7.9}$$

Thanks to the structured $\boldsymbol{W} = \sqrt{d}\boldsymbol{R}^\top\boldsymbol{B}$, we show the monotonic descent property, convergence rate for convex ϕ_λ, and a strictly monotonic descent for a k-sparse constrained problem.

For both convex and non-convex ϕ_λ, our update rule in Eq.(7.9) leads to a monotonic descent.

Theorem 23. *(Monotonic Descent) For a function $\phi_\lambda(\cdot)$, denote $h(\cdot)$ as the proximal operator of $\phi_\lambda(\cdot)$. Given a bounded $\boldsymbol{x} \in \mathcal{R}^d$, set $\boldsymbol{y}_{t+1} = h\big(\boldsymbol{W}^\top\boldsymbol{x} + (\boldsymbol{I} - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{W})\boldsymbol{y}_t\big)$ with $\frac{1}{N}\boldsymbol{W}\boldsymbol{W}^\top = \boldsymbol{I}_d$. Denote $\widehat{Q}(\boldsymbol{y}) := \frac{1}{2}\|\boldsymbol{x} - \widehat{\mathcal{A}}(\boldsymbol{y})\|_2^2 + \frac{1}{N}\phi_\lambda(\boldsymbol{y})$. For $t \geq 0$, we have*

$$\widehat{Q}(\boldsymbol{y}_{t+1}) \leq \widehat{Q}(\boldsymbol{y}_t) - \frac{1}{2N}\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \tag{7.10}$$

$$= \widehat{Q}(\boldsymbol{y}_t) - \frac{1}{2N}\|(\boldsymbol{I}_d - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{W})(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \leq \widehat{Q}(\boldsymbol{y}_t). \tag{7.11}$$

Remark: For the finite dimensional case, the monotonic descent property is preserved. For popular activation function, e.g., sigmoid, tanh and ReLU, it corresponds to a finite dimensional (implicit) $\phi(\cdot)$-regularized problem. A T-layer NN performs T-steps monotonic descent of the $\phi(\cdot)$-regularized problem $\widehat{Q}(\cdot)$ desipte of the non-convexity of the activation function $h(\cdot)$.

For convex ϕ_λ, we can achieve a $O(\frac{1}{T})$ convergence rate, which is formally shown in Theorem 24.

Theorem 24. *For a convex function $\phi_\lambda(\cdot)$, denote $h(\cdot)$ as the proximal operator of $\phi_\lambda(\cdot)$. Given a bouned $\boldsymbol{x} \in \mathcal{R}^d$, set $\boldsymbol{y}_{t+1} = h\big(\boldsymbol{W}^\top \boldsymbol{x} + (\boldsymbol{I} - \frac{1}{N}\boldsymbol{W}^\top \boldsymbol{W})\boldsymbol{y}_t\big)$ with $\frac{1}{N}\boldsymbol{W}\boldsymbol{W}^\top = \boldsymbol{I}_d$. Denote $\widehat{Q}(\boldsymbol{y}) := \frac{1}{2}\|\boldsymbol{x} - \widehat{\mathcal{A}}(\boldsymbol{y})\|_2^2 + \frac{1}{N}\phi_\lambda(\boldsymbol{y})$ and $\boldsymbol{y}^*$ as an optimal of $\widehat{Q}(\cdot)$. For $T \geq 1$, we have*

$$T\big(\widehat{Q}(\boldsymbol{y}_T) - \widehat{Q}(\boldsymbol{y}^*)\big) \leq \frac{1}{2N}\|\boldsymbol{y}_0 - \boldsymbol{y}^*\|_2^2 - \frac{1}{2N}\|\boldsymbol{y}_T - \boldsymbol{y}^*\|_2^2 - \frac{1}{2}\sum_{t=0}^{T-1}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_t - \boldsymbol{y}^*)\|_2^2 - \frac{1}{2}\sum_{t=0}^{T-1}\frac{t+1}{N}\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2. \tag{7.12}$$

Theorem 25. *(Strictly Monotonic Descent of k-sparse problem) Let $L(\boldsymbol{y}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{y}\|_2^2$, s.t. $\|\boldsymbol{y}\|_0 \leq k$ with $\boldsymbol{D} = \frac{\sqrt{d}}{\sqrt{N}}\boldsymbol{R}^\top \boldsymbol{B}$, where $\boldsymbol{B}$ is constructed as in [119] with $N = 2n, d = 2m$. Set $\boldsymbol{y}_{t+1} = h(\boldsymbol{a}_{t+1})$ with sparity k and $\boldsymbol{a}_{t+1} = \boldsymbol{D}^\top \boldsymbol{x} + (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D})\boldsymbol{y}_t$. For $\forall t \geq 1$, we have*

$$\begin{aligned} & L(\boldsymbol{y}_{t+1}) \\ & \leq L(\boldsymbol{y}_t) + \frac{1}{2}\|\boldsymbol{y}_{t+1} - \boldsymbol{a}_{t+1}\|_2^2 - \frac{1}{2}\|\boldsymbol{y}_t - \boldsymbol{a}_{t+1}\|_2^2 - \frac{n - (2k-1)\sqrt{n} - m}{2n}\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 \\ & \leq L(\boldsymbol{y}_t), \end{aligned} \tag{7.13}$$

where $h(\cdot)$ is defined as

$$h(z_j) = \begin{cases} z_j & \text{if } |z_j| \text{ is one of the } k\text{-highest values of } |\boldsymbol{z}| \in \mathcal{R}^N \\ 0 & \text{otherwise} \end{cases}. \tag{7.14}$$

Remark: When sparsity $k < \frac{n-m+\sqrt{n}}{2\sqrt{n}}$, we have $L(\boldsymbol{y}_{t+1}) < L(\boldsymbol{y}_t)$ unless $\boldsymbol{y}_{t+1} = \boldsymbol{y}_t$. Our update with the structured $\boldsymbol{D}$ makes a strictly monotonic descent progress each step.

7.3.2 Learning Parameter $\boldsymbol{R}$

Supervised Learning: For the each t^{th} layer, we can maintain an orthogonal matrix $\boldsymbol{R}_t$. The orthogonal matrix $\boldsymbol{R}_t$ can be parameterized by exponential mapping or

Cayley mapping [72] of a skew-symmetric matrix. We can employ the Cayley mapping to enable gradient update w.r.t a loss function $\ell(\cdot)$ in an end-to-end training. Specifically, the orthogonal matrix $\boldsymbol{R}_t$ can be obtained by the Cayley mapping of a skew-symmetric matrix as

$$\boldsymbol{R}_t = (\boldsymbol{I} + \boldsymbol{M}_t)(\boldsymbol{I} - \boldsymbol{M}_t)^{-1}, \tag{7.15}$$

where $\boldsymbol{M}_t$ is a skew-symmetric matrix, i.e., $\boldsymbol{M}_t = -\boldsymbol{M}_t^\top \in \mathbb{R}^{d\times d}$. For a skew-symmetric matrix $\boldsymbol{M}_t$, only the upper triangular matrix (without main diagonal) are free parameters. Thus, total the number of free parameters of T-Layer is $Td(d-1)/2$.

Unsupervised Learning: The parameter $\boldsymbol{R}$ can also be learned in an unsupervised manner. Specifically, for a finite dataset $\boldsymbol{X}$, the finite dimensional approximation problem with the structured $\boldsymbol{W} = \sqrt{d}\boldsymbol{R}^\top\boldsymbol{B}$ is given as

$$\begin{aligned} &\min_{\boldsymbol{Y},\boldsymbol{R}} \frac{1}{2}\|\boldsymbol{X} - \frac{\sqrt{d}}{N}\boldsymbol{R}^\top\boldsymbol{B}\boldsymbol{Y}\|_F^2 + \frac{1}{N}\phi_\lambda(\boldsymbol{Y}) \\ &\text{subject to } \ \boldsymbol{R}^\top\boldsymbol{R} = \boldsymbol{R}\boldsymbol{R}^\top = \boldsymbol{I}_d, \end{aligned} \tag{7.16}$$

where $\phi_\lambda(\cdot)$ is a separable non-convex or convex regularization function with parameter λ, i.e., $\phi_\lambda(\boldsymbol{Y}) = \sum_i \phi_\lambda(\boldsymbol{y}^{(i)})$.

The problem (7.16) can be solved by the alternative descent method. For a fixed $\boldsymbol{R}$, we perform a iterative update of $\boldsymbol{Y}$ a few steps to decrease the objective. For the fixed $\boldsymbol{Y}$, parameter $\boldsymbol{R}$ has a closed-form solution.

Fix $\boldsymbol{R}$, Optimize $\boldsymbol{Y}$: The problem (7.16) can be rewritten as :

$$\frac{1}{2}\|\boldsymbol{X} - \frac{\sqrt{d}}{N}\boldsymbol{R}^\top\boldsymbol{B}\boldsymbol{Y}\|_F^2 + \frac{1}{N}\phi_\lambda(\boldsymbol{Y}) = \sum_i \widehat{Q}(\boldsymbol{y}^{(i)}). \tag{7.17}$$

Thus, with fixed $\boldsymbol{R}$, we can update each $\boldsymbol{y}^{(i)}$ by Eq.(7.9) in parallel. We can perform T_1 steps update with initialization as the output of previous alternative phase, i.e., $\boldsymbol{Y}_0^j = \boldsymbol{Y}_{T_1}^{j-1}$ (and initialization $\boldsymbol{Y}_0^0 = 0$ and $\boldsymbol{R}_0 = \boldsymbol{I}_d$).

Fix $\boldsymbol{Y}$, Optimize $\boldsymbol{R}$: This is the nearest orthogonal matrix problem, which has a closed-form solution as shown in [151]. Let $\frac{\sqrt{d}}{N}\boldsymbol{B}\boldsymbol{Y}\boldsymbol{X}^\top = \boldsymbol{U}\Gamma\boldsymbol{V}^\top$ obtained by singular value decomposition (SVD), where $\boldsymbol{U}, \boldsymbol{V}$ are orthgonal matrix. Then, Eq.(7.16) is minimized by $\boldsymbol{R} = \boldsymbol{U}\boldsymbol{V}^\top$.

Remark: A T_2-step alternative descent computation graph of $\boldsymbol{R}$ and $\boldsymbol{Y}$ can be viewed as a T_1T_2-layer NN block, which can be used as a plug-in of popular backbones for robust deep learning.

7.3.3 Kernel Approximation

Define $k_{T,N}(\boldsymbol{x},\boldsymbol{x}') = \frac{1}{N} < \boldsymbol{y}_T(\boldsymbol{W},\boldsymbol{x}), \boldsymbol{y}_T(\boldsymbol{W},\boldsymbol{x}') >$, where $\boldsymbol{y}_T(\boldsymbol{W},\boldsymbol{x}) : \mathcal{R}^d \to \mathcal{R}^N$ is a finite approximation of $f_T(\cdot,\boldsymbol{x}) \in \mathcal{H}_{k_T}$. We know $k_{T,N}(\boldsymbol{x},\boldsymbol{x}')$ is bounded kernel, and it is an approximation of kernel $k_{T,\infty} = \mathbb{E}_w[f_T(\boldsymbol{w},\boldsymbol{x})f_T(\boldsymbol{w},\boldsymbol{x}')]$.

Remark: Let $\boldsymbol{B}$ be a points set that marginally uniformly distributed on the surface of sphere $\mathbb{S}^{d-1}$ (e.g, Block-wise random orthogonal rotation of structured samples [117]). Employing our structured approximation $\boldsymbol{W} = \boldsymbol{R}^\top \boldsymbol{B}$, we know $\forall \boldsymbol{R} \in SO(d)$ and $\forall f \in \overline{\mathcal{L}}_2$, $\lim_{N\to\infty} \frac{\sum_{i=1}^N f(\boldsymbol{w}_i)}{N} = \mathbb{E}_{\boldsymbol{w}\sim Uni[\mathbb{S}^{d-1}]}[f(\boldsymbol{w})]$. It means that although the orthogonal rotation parameter $\boldsymbol{R}$ is learned, we still maintain an unbiased estimator of $\mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w})]$.

First-Layer Kernel: Set $\boldsymbol{y} = \boldsymbol{0}$ and $f_0 = 0$, we know $y_i(\boldsymbol{x}) = h(\boldsymbol{w}_i^\top \boldsymbol{x})$ and $f_1(\boldsymbol{w},\boldsymbol{x}) = h(\boldsymbol{w}^\top \boldsymbol{x})$. Suppose $|h(x)| \le c|x|$ (or $|h(x)| \le c$), $0 < c < \infty$, it follows that

$$\begin{aligned}\lim_{N\to\infty} k_{1,N}(\boldsymbol{x},\boldsymbol{x}') &= \lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^N h(\boldsymbol{w}_i^\top \boldsymbol{x})h(\boldsymbol{w}_i^\top \boldsymbol{x}') = \mathbb{E}_{\boldsymbol{w}}[h(\boldsymbol{w}^\top \boldsymbol{R}\boldsymbol{x})h(\boldsymbol{w}^\top \boldsymbol{R}\boldsymbol{x}')] \\ &= \mathbb{E}_{\boldsymbol{w}}[h(\boldsymbol{w}^\top \boldsymbol{x})h(\boldsymbol{w}^\top \boldsymbol{x}')] = k_{1,\infty}(\boldsymbol{x},\boldsymbol{x}'). \end{aligned} \tag{7.18}$$

In Eq.(7.18), we use the fact that a rotation does not change the uniform surface measure on $\mathbb{S}^{d-1}$. The first layer kernel $k_{1,N}$ uniformly converge to $k_{1,\infty}$ over a bounded domain $\mathcal{X} \times \mathcal{X}$.

Higher-Layer Kernel: For both the shared $\boldsymbol{R}$ case and the unsupervised updating $\boldsymbol{R}$ case, the monotonic descent property and convergence rate is well preserved for any bounded $\boldsymbol{x} \in \mathcal{X}$. With the same assumption of $h(\cdot)$ and $\boldsymbol{y} = \boldsymbol{0}$, as $N \to \infty$, we know $\boldsymbol{y}_t \to \widehat{f}_t \in \overline{\mathcal{L}}_2$, where $\widehat{f}_t$ is a countable-infinite dimensional functional. And inequality (7.10) and inequality (7.12) uniformly converges to inequality (7.19) and inequality (7.21) over a bounded domain $\mathcal{X}$, respectively.

$$Q(\widehat{f}_{t+1}) \le Q(\widehat{f}_t) - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[\big(\widehat{f}_{t+1}(\boldsymbol{w}) - \widehat{f}_t(\boldsymbol{w})\big)^2] + \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(\widehat{f}_{t+1}(\boldsymbol{w}) - \widehat{f}_t(\boldsymbol{w})\big)]\|_2^2 \tag{7.19}$$

$$= Q(\widehat{f}_t) - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(\widehat{f}_{t+1}(\boldsymbol{w}) - \widehat{f}_t(\boldsymbol{w}) - \boldsymbol{w}^\top \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}(\widehat{f}_{t+1}(\boldsymbol{w}) - \widehat{f}_t(\boldsymbol{w}))])^2] \le Q(\widehat{f}_t), \tag{7.20}$$

$$\begin{aligned} T\big(Q(\widehat{f}_T) - Q(f_*)\big) \le{}& \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_0(\boldsymbol{w}) - f_*(\boldsymbol{w}))^2] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(\widehat{f}_T(\boldsymbol{w}) - f_*(\boldsymbol{w}))^2] \\ & - \frac{1}{2}\sum_{t=0}^{T-1}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(\widehat{f}_t(\boldsymbol{w}) - f_*(\boldsymbol{w})\big)]\|_2^2 - \frac{1}{2}\sum_{t=0}^{T-1}(t+1)\mathbb{E}_{\boldsymbol{w}}[(\widehat{f}_{t+1}(\boldsymbol{w}) - \widehat{f}_t(\boldsymbol{w}))^2]. \end{aligned} \tag{7.21}$$

It is worth noting that $\lim_{N\to\infty} k_{T,N}$ converge to a $\widehat{k}_{T,\infty}$ that is determined by the initialization $\boldsymbol{R}_0$ and dataset $\boldsymbol{X}$. Specifically, for both the unsupervised learning case and the shared parameter case, the approximated kernel converge to a fixed kernel as the width tends to infinity. As $N \to \infty$, training a finite structured NN with GD tends to perform a functional gradient descent with a fixed kernel. For a strongly convex regularized regression problem, functional gradient descent leads to global convergence.

For the case of updating T-layer parameter $\boldsymbol{R}_t, t \in \{1, \cdots, T\}$ in a supervised manner, the sequence $\{\boldsymbol{R}_t\}$ determines the kernel. When the data distribution is isotropic, e.g., $\mathbb{S}^{d-1}$, the monotonic descent property is preserved for the expectation $\mathbb{E}_X[Q(\widehat{f}_t, X)]$ (at least one step descent). Actually, when parameters of each layer are learned in a supervised manner, the model is adjusted to fit the supervised signal. When the prior regularization $\mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(\widehat{f}(\boldsymbol{w}))]$ is consistent with learning the supervised signal, the monotonic descent property is well preserved. When the prior regularization contradicts the supervised signal, the monotonic descent property for prior is weakened.

7.4 Functional Optimization

We can minimize a regularized expected risk defined as

$$J(f) := \underbrace{\mathbb{E}_{X,Y}[\ell(g(X)), Y] + \frac{\lambda}{2}\|g\|^2_{\mathcal{H}_k}}_{J_1} + \beta \underbrace{\mathbb{E}_X\big[\frac{1}{2}\|X - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f(\boldsymbol{w}, X)]\|_2^2 + \mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f(\boldsymbol{w}, X))]\big]}_{J_2}, \tag{7.22}$$

where the functional space $\mathcal{H}_k \ni g$ is is determined by the kernel $k(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w}, \boldsymbol{x}) f(\boldsymbol{w}, \boldsymbol{y})]$. Our NOK enables us to optimize the objective J by optimizing J_1 and J_2 separately. Namely, the functional space $\mathcal{H}_{k_T} \ni g$ is determined by the kernel associated with the T-step update f_T. With our NOK, J_2 with convex regularization $\phi_\lambda(\cdot)$ can be optimized with a convergence rate $O(\frac{1}{T})$ by the T-layer network architecture. When $\phi_\lambda(\cdot)$ is an indicator function, the optimal J_2 actually is the l_2-norm optimal transport between $p(X)$ and a probability measure induced by $\mathcal{A}(f_X^*, X) = \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f_X^*(\boldsymbol{w}, X)]$.

For a convex function $\ell(\cdot)$, $J_1(g)$ is strongly convex w.r.t the functional $g \in \mathcal{H}_k$. Functional gradient descent can converge to a minimizer of J_1. For regression problems, $\ell(z, y) = \frac{1}{2}(z - y)^2$, the functional gradient of J_1 is

$$\partial J_1(g) = \mathbb{E}_{X,Y}[\partial_{z=g(X)} \ell(g(X), Y) k(\cdot, X)] + \lambda g = (\Sigma + \lambda I) g - \mathbb{E}_{X,Y}[Y k(\cdot, X)], \tag{7.23}$$

where $\Sigma := \mathbb{E}_{X\sim p_X}[k(\cdot, X) \otimes_{\mathcal{H}} k(X, \cdot)]$ denotes the covariance operator.

We can perform the average stochastic gradient descent using a stochastic unbiased estimator of Eq.(7.23). For strongly convex problem, we can achieve $O(\frac{1}{T})$ convergence rate (Theorem A in [132]).

7.5 Rademacher Complexity and Generalization Bound

We show the Rademacher complexity bound and the generalization bound of our structured approximated NOK (SNOK).

Neural Network Structure: For structured approximated NOK networks (SNOK), the 1-T layers are given as

$$\boldsymbol{y}_{t+1} = h(\boldsymbol{D}^\top \boldsymbol{R}_t \boldsymbol{x} + (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D})\boldsymbol{y}_t), \tag{7.24}$$

where $\boldsymbol{R}_t$ are free parameters such that $\boldsymbol{R}_t^\top \boldsymbol{R}_t = \boldsymbol{R}_t^\top \boldsymbol{R}_t = \boldsymbol{I}_d$. And $\boldsymbol{D}$ is a scaled structured spherical samples such that $\boldsymbol{D}\boldsymbol{D}^\top = \boldsymbol{I}_d$ [117], and $\boldsymbol{y}_0 = \boldsymbol{0}$.

The last layer ($(T{+}1)^{th}$ layer) is given by $z = \boldsymbol{w}^\top \boldsymbol{y}_{T+1}$. Consider a L-Lipschitz continuous loss function $\ell(z, y) : \mathcal{Z} \times \mathcal{Y} \to [0, 1]$ with Lipschitz constant L w.r.t the input z.

Rademacher Complexity [21]: Rademacher complexity of a function class $\mathcal{G}$ is defined as

$$\mathfrak{R}_N(\mathcal{G}) := \frac{1}{N}\mathbb{E}\left[\sup_{g\in\mathcal{G}} \sum_{i=1}^{N} \epsilon_i g(\boldsymbol{x}_i)\right], \tag{7.25}$$

where $\epsilon_i, i \in \{1, \cdots, N\}$ are i.i.d. samples drawn uniformly from $\{+1, -1\}$ with probality $\mathrm{P}[\epsilon_i = +1] = \mathrm{P}[\epsilon_i = -1] = 1/2$. And $\boldsymbol{x}_i, i \in \{1, \cdots, N\}$ are i.i.d. samples from $\mathcal{X}$.

Theorem 26. *(Rademacher Complexity Bound) Consider a Lipschitz continuous loss function $\ell(z, y) : \mathcal{Z} \times \mathcal{Y} \to [0, 1]$ with Lipschitz constant L w.r.t the input z. Let $\widetilde{\ell}(z, y) := \ell(z, y) - \ell(0, y)$. Let $\widehat{G}$ be the function class of our $(T{+}1)$-layer SNOK mapping from $\mathcal{X}$ to $\mathcal{Z}$. Suppose the activation function $|h(\boldsymbol{y})| \le |\boldsymbol{y}|$ (element-wise), and the l_2-norm of last layer weight is bounded, i.e., $\|\boldsymbol{w}\|_2 \le \mathcal{B}_w$. Let $(\boldsymbol{x}_i, y_i)_{i=1}^N$ be i.i.d. samples drawn from $\mathcal{X} \times \mathcal{Y}$. Denote $\boldsymbol{Y}_{T+1}$ as the T^{th} layer output. Denote the mutual coherence of $\boldsymbol{Y}_{T+1}$ as μ^*, i.e., $\mu^* = \mu(\boldsymbol{Y}_{T+1}) \le 1$. Then, we have*

$$\mathfrak{R}_N(\widetilde{\ell} \circ \widehat{G}) = \frac{1}{N}\mathbb{E}\left[\sup_{g\in\widehat{\mathcal{G}}} \sum_{i=1}^{N} \epsilon_i \widetilde{\ell}(g(\boldsymbol{x}_i), y_i)\right] \le \frac{L\mathcal{B}_w\sqrt{\big((N-1)\mu^* + 1\big)}}{N} \sqrt{\sum_{i=0}^{T-1} \beta^i} \|\boldsymbol{X}\|_F, \tag{7.26}$$

where $\beta = \|\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}\|_2^2 \leq 1$, $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N]$. $\|\cdot\|_F$ *and* $\|\cdot\|_2$ *denote the matrix Frobenius norm and matrix spectral norm,respectively.*

Remark: A small mutual coherence $\mu(\boldsymbol{Y}_{T+1})$ leads to a small Rademacher complexity bound. When the width of NN $N_D > d$, we have $\beta = 1$. In this case, the Rademacher complexity bound has a complexity $O(\sqrt{T})$ w.r.t. the depth of NN (SNOK).

Theorem 27. *(Generalization Bound) Consider a Lipschitz continuous loss function* $\ell(z, y) : \mathcal{Z} \times \mathcal{Y} \to [0, 1]$ *with Lipschitz constant* L *w.r.t the input* z. *Let* $\widetilde{\ell}(z, y) := \ell(z, y) - \ell(0, y)$. *Let* $\widehat{G}$ *be the function class of our* $(T+1)$*-layer SNOK mapping from* $\mathcal{X}$ *to* $\mathcal{Z}$. *Suppose the activation function* $|h(\boldsymbol{y})| \leq |\boldsymbol{y}|$ *(element-wise), and the* l_2*-norm of last layer weight is bounded, i.e.,* $\|\boldsymbol{w}\|_2 \leq \mathcal{B}_w$. *Let* $(\boldsymbol{x}_i, y_i)_{i=1}^N$ *be i.i.d. samples drawn from* $\mathcal{X} \times \mathcal{Y}$. *Denote* $\boldsymbol{Y}_{T+1}$ *as the* T^{th} *layer output with input* $\boldsymbol{X}$. *Denote the mutual coherence of* $\boldsymbol{Y}_{T+1}$ *as* μ^*, *i.e.,* $\mu^* = \mu(\boldsymbol{Y}_{T+1}) \leq 1$. *Then, for* $\forall N$ *and* $\forall \delta, 0 < \delta < 1$, *with a probability at least* $1 - \delta$, $\forall g \in \widehat{G}$, *we have*

$$\mathbb{E}\big[\ell(g(X), Y)\big] \leq \frac{1}{N}\sum_{i=1}^{N} \ell(g(\boldsymbol{x}_i), y_i) + \frac{L\mathcal{B}_w\sqrt{\big((N-1)\mu^*+1\big)}}{N}\sqrt{\sum_{i=0}^{T-1}\beta^i}\|\boldsymbol{X}\|_F + \sqrt{\frac{8\ln(2/\delta)}{N}} \tag{7.27}$$

where $\beta = \|\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}\|_2^2 \leq 1$, $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N]$. $\|\cdot\|_F$ *and* $\|\cdot\|_2$ *denote the matrix Frobenius norm and matrix spectral norm,respectively.*

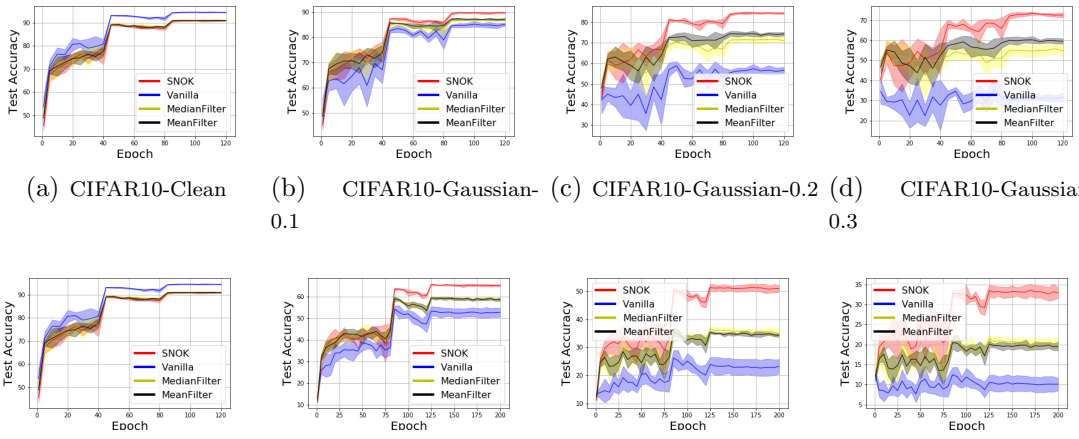
Remark: The mutual coherence $\mu(\boldsymbol{Y}_{T+1})$ (or $\|\boldsymbol{Y}_{T+1}^\top \boldsymbol{Y}_{T+1} - \boldsymbol{I}\|_F^2$, etc.) can serve as a good regularization to reduce Rademacher complexity and generalization bound. When the width of SNOK (N_D) is large enough, specifically, when $N_D > N$, it is possible to obtain $\mu(\boldsymbol{Y}_{T+1}) = 0$ (orthogonal representation), which significantly reduces the generalization bound. Namely, overparameterized deep NNs can increase the expressive power to reduce empirical risk and reduce the generalization bound at the same time.

7.6 Experiments

We evaluate the performance of our unsupervised SNOK blocks on classification tasks under input noise perturbation (Gaussian noise or Laplace noise), and under FGSM adversarial attack [62].

7.6.1 Empirical Evaluation on Classification under Gaussian Noise Perturbation

We first evaluate our unsupervised SNOK blocks as a plug-in on classification tasks under Gaussian noise perturbation. In all the experiments, the Gaussian noise is added after input normalization. The standard deviation of input noise is set to $\{0, 0.1, 0.2, 0.3\}$, respectively. We employ both DenseNet-100 [79] and ResNet-34 [71] as backbone. We test the performance of four methods in comparison: (1) *Vanilla Backbone*, (2) *Backbone + Mean Filter*, (3) *Backbone + Median Filter*, (4) *Backbone + SNOK*. For both *Mean Filter* and *Median Filter* cases, we set the filter neighborhood size as 3×3 same as in [177]. For our SNOK case, we plug two SNOK blocks before and after the first learnable *Conv2D* layer. In all the experiments, CIFAR10 and CIFAR100 datasets [100] are employed for evaluation. All the methods are evaluated over five independent runs with seeds $\{1, 2, 3, 4, 5\}$. During training, we stored the model every five epochs, and reported all evaluation results over all the stored models. It covers the whole training trajectory, which is more informative.

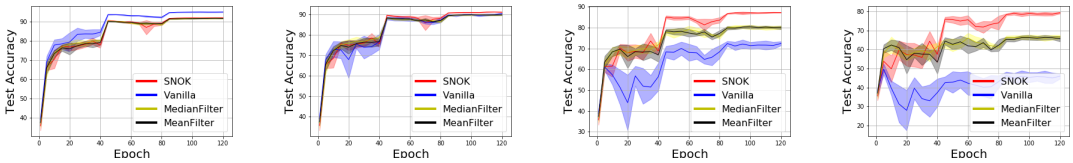


(a) CIFAR10-Clean (b) CIFAR10-Gaussian-0.1 (c) CIFAR10-Gaussian-0.2 (d) CIFAR10-Gaussian-0.3

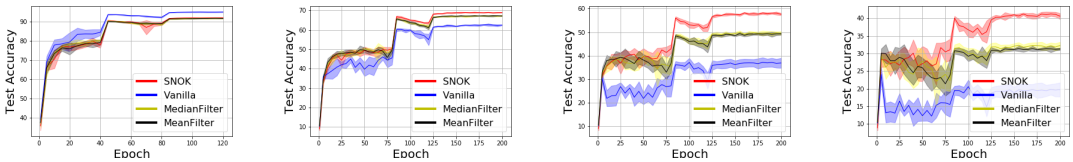
(e) CIFAR100-Clean (f) CIFAR100-Gaussian-0.1 (g) CIFAR100-Gaussian-0.2 (h) CIFAR100-Gaussian-0.3

Figure 7.1: Mean test accuracy ± std over 5 independent runs under Gaussian noise with DenseNet backbone

The results of classification under Gaussian noise perturbation with DenseNet backbone and ResNet backbone are shown in Fig. 7.1 and Fig 7.2, respectively. We can observe that *Backbone + Mean Filter*, *Backbone + Median Filter*, and our *Backbone + SNOK* achieve a sightly lower classification accuracy on the case without noise perturbation. Moreover, on the case without input noise, our SNOK plug-in obtains a similar classification performance compared with other plug-in blocks. Furthermore,



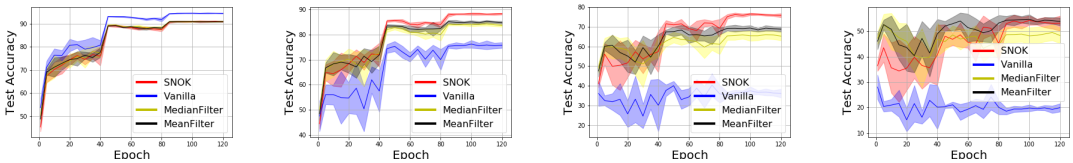
(a) CIFAR10-Clean (b) CIFAR10-Gaussian-0.1 (c) CIFAR10-Gaussian-0.2 (d) CIFAR10-Gaussian-0.3



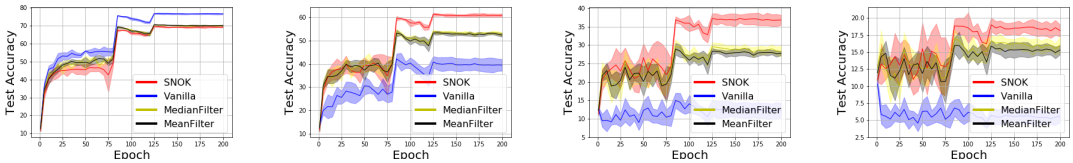
(e) CIFAR100-Clean (f) CIFAR100-Gaussian-0.1 (g) CIFAR100-Gaussian-0.2 (h) CIFAR100-Gaussian-0.3

Figure 7.2: Mean test accuracy ± std over 5 independent runs under Gaussian noise with ResNet backbone

our *Backbone + SNOK* obtains a increasingly higher test accuracy compared with other baselines as the standard deviation of the noise grows. This shows a superior robustness of our SNOK blocks against Gaussian noise perturbation. Notably, the *Vanilla Backbone* achieves a degenerate performance when the standard deviation of the noise is large. In contrast, our *Backbone + SNOK* achieves almost two times test accuracy compared with the *Vanilla Backbone* on the large noise case. This shows a significant improvement, which demonstrates a promising application value of our SNOK blocks.

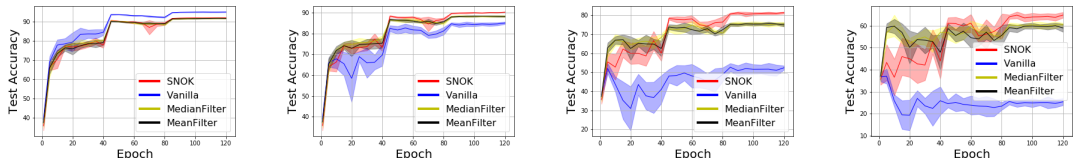


(a) CIFAR10-Clean (b) CIFAR10-Laplace-0.1 (c) CIFAR10-Laplace-0.2 (d) CIFAR10-Laplace-0.3

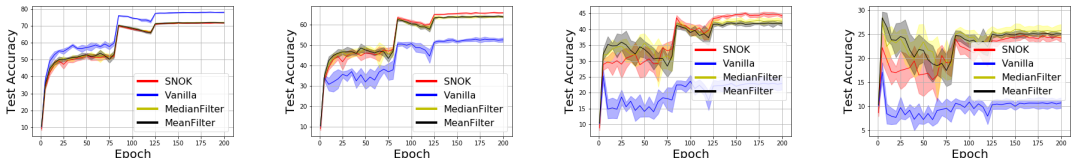


(e) CIFAR100-Clean (f) CIFAR100-Laplace-0.1 (g) CIFAR100-Laplace-0.2 (h) CIFAR100-Laplace-0.3

Figure 7.3: Mean test accuracy ± std over 5 independent runs under Laplace noise with DenseNet backbone



(a) CIFAR10-Clean (b) CIFAR10-Laplace-0.1 (c) CIFAR10-Laplace-0.2 (d) CIFAR10-Laplace-0.3



(e) CIFAR100-Clean (f) CIFAR100-Laplace-0.1 (g) CIFAR100-Laplace-0.2 (h) CIFAR100-Laplace-0.3

Figure 7.4: Mean test accuracy ± std over 5 independent runs under Laplace noise with ResNet backbone

7.6.2 Empirical Evaluation on Classification under Laplace Noise Perturbation

We further evaluate our unsupervised SNOK blocks as a plug-in on classification tasks under Laplace noise perturbation. In all the experiments, the Laplace noise is added after input normalization. The standard deviation of the Laplace noise is set to $\{0, 0.1, 0.2, 0.3\}$, respectively. We employ both DenseNet-100 [79] and ResNet-34 [71] as backbone. We test the performance of four methods in comparison same as in Section 7.6.1. For both *Mean Filter* and *Median Filter* cases, we set the filter neighborhood size as 3×3 same as in [177]. For our SNOK case, we plug two SNOK blocks before and after the first learnable *Conv2D* layer. In all the experiments, CIFAR10 and CIFAR100 datasets [100] are employed for evaluation. All the methods are evaluated over five independent runs with seeds $\{1, 2, 3, 4, 5\}$. During training, we stored the model every five epochs, and reported all evaluation results over all the stored models.

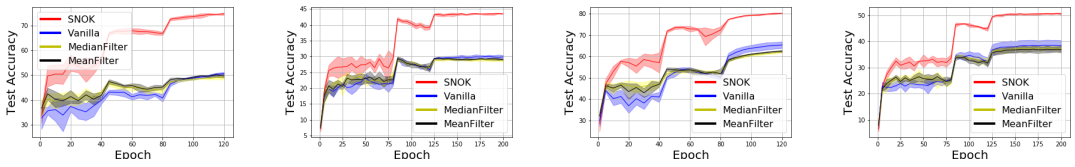
The results of classification under Laplace noise perturbation with DenseNet backbone and ResNet backbone are shown in Fig. 7.3 and Fig 7.4, respectively. We can observe that all the robust baseline methods obtain a sightly lower test accuracy compared with vanilla backbone on the clean case. Moreover, we find that all robust baseline methods achieve a significant higher test accuracy compared with vanilla backbone on the case with noise perturbation. In addition, the vanilla backbone totally degenerate when the standard deviation of the input noise is 0.3. Furthermore, we find that DenseNet backbone with our SNOK plug-in performs better than ResNet

backbone with SNOK. This may show that ResNet is more sensitive to input Laplace noise.

7.6.3 Empirical Evaluation on Classification with Adversarial Perturbation

We further evaluate our unsupervised SNOK blocks as a plug-in on classification tasks under adversarial perturbation. We employ both DenseNet-100 [79] and ResNet-34 [71] as backbone. We test the performance of four methods in comparison same as in Section 7.6.1. In all the experiments, CIFAR10 and CIFAR100 datasets [100] are employed for evaluation. All the methods are evaluated over five independent runs with seeds $\{1, 2, 3, 4, 5\}$. During training, we stored the model every five epochs, and reported all evaluation results over all the stored models. We employ the FGSM method to generate the adversarial perturbation.

The experimental results of different models under the FGSM attack are shown in Fig. 7.5. Our SNOK plug-in achieves a significantly higher test accuracy than baselines. Moreover, we can observe that mean filter and median filter plug-in do not gain an improvement of test accuracy compared with vanilla backbone under FGSM attack. This shows that these two filter block fail to gain additional robustness against the FGSM attack, although they obtains robustness against Gaussian noise and Laplace noise. In contrast, our SNOK plug-in not only bring a robustness against Gaussian noise/Laplace noise, but also gain a robustness under FGSM attack. This show a superior robustness of our SNOK block in a consistent way.



(a) DenseNet-CIFAR10 (b) DenseNet-CIFAR100 (c) ResNet34-CIFAR10 (d) ResNet34-CIFAR100

Figure 7.5: Mean test accuracy ± std over 5 independent runs on CIFAR10/CIFAR100 dataset under FGSM adversarial attack for DenseNet and ResNet backbone.

7.7 Summary

We first proposed a novel kernel family NOK. We also analyzed its approximation, the connection to deep NNs, the NN architecture optimization properties, and the gener-

alization bounds. Our studies show that overparameterized NN (SNOK) with a wide range of popular activation functions, e.g., ReLU, max pooling, soft-thresholding, can increase the expressive power to reduce the empirical risk, and reduce the generalization bound at the same time. In future, we will investigate the convergence behavior of training the supervised SNOK with SGD.

Chapter 8

Curriculum Loss for Robust Deep Learning

8.1 Chapter Abstract

Deep neural networks (DNNs) have great expressive power, which can even memorize samples with wrong labels. It is vitally important to reiterate robustness and generalization in DNNs against label corruption. To this end, this chapter studies the 0-1 loss, which has a monotonic relationship with empirical adversary (reweighted) risk [77]. Although the 0-1 loss has some robust properties, it is difficult to optimize. To efficiently optimize the 0-1 loss while keeping its robust properties, we propose a very simple and efficient loss, i.e. curriculum loss (CL). Our CL is a tighter upper bound of the 0-1 loss compared with conventional summation based surrogate losses, which is an tighter approximation of expected risk. Moreover, CL can adaptively select samples for model training. As a result, our loss can be deemed as a novel perspective of curriculum sample selection strategy, which bridges a connection between curriculum learning and robust learning. Experimental results on benchmark datasets validate the robustness of the proposed loss.

8.2 Curriculum Loss

In this section, we present the framework of our proposed Curriculum Loss (CL). We begin with discussion about robustness of the 0-1 loss in Section 8.2.1. We then show that our CL is a tighter upper bound of the 0-1 loss compared with conventional summation based surrogate losses in Section 8.2.2. A tighter bound of the 0-1 loss means that it is less sensitive to the noisy outliers, and it better preserves the robustness of the 0-1 loss with a small rate of label corruption. For a large rate of label corruption,

we extend our CL to a Noise Pruned Curriculum Loss (NPCL) to address this issue in Section 8.2.3. A simple multi-class extension and a novel soft multi-hinge loss are included in the Appendix. All the detailed proofs can be found in the Appendix as well.

8.2.1 Robustness of 0-1 loss against label corruption

We rephrase Theorem 1 in [77] from a different perspective, which motivates us to employ the 0-1 loss for training against label corruption.

Theorem 28. ***(Monotonic Relationship)*** *(Hu et al. [77]) Let $p(x,y)$ and $q(x,y)$ be the training and test density,respectively. Define $r(x,y) = q(x,y)/p(x,y)$ and $r_i = r(x_i,y_i)$. Let $l(\widehat{y},y) = \mathbf{1}\big(sign(\widehat{y}) \neq y\big)$ and $l(\widehat{y},y) = \mathbf{1}\big(argmax_k(\widehat{y}_k) \neq y\big)$ be 0-1 loss for binary classification and multi-class classification, respectively. Let $f(\cdot)$ be convex with $f(1) = 0$. Define risk $\mathcal{R}(\theta)$, empirical risk $\widehat{\mathcal{R}}(\theta)$, adversarial risk $\mathcal{R}_{adv}(\theta)$ and empirical adversarial risk $\widehat{\mathcal{R}}_{adv}(\theta)$ as*

$$\mathcal{R}(\theta) = \mathbb{E}_{p(x,y)}\left[l(g_\theta(x),y)\right] \tag{8.1}$$

$$\widehat{\mathcal{R}}(\theta) = \frac{1}{n}\sum\nolimits_{i=1}^{n} l(g_\theta(x_i),y_i) \tag{8.2}$$

$$\mathcal{R}_{adv}(\theta) = \sup_{r\in\mathcal{U}_f} \mathbb{E}_{p(x,y)}\left[r(x,y)l(g_\theta(x),y)\right] \tag{8.3}$$

$$\widehat{\mathcal{R}}_{adv}(\theta) = \sup_{\mathbf{r}\in\widehat{\mathcal{U}}_f} \frac{1}{n}\sum\nolimits_{i=1}^{n} r_i l(g_\theta(x_i),y_i), \tag{8.4}$$

where $\mathcal{U}_f = \left\{r(x,y)\,\middle|\,\mathbb{E}_{p(x,y)}\left[f\left(r(x,y)\right)\right] \leq \delta, \mathbb{E}_{p(x,y)}\left[r(x,y)\right] = 1, r(x,y) \geq 0, \forall (x,y) \in \mathcal{X}\times\mathcal{Y}\right\}$ and $\widehat{\mathcal{U}}_f = \left\{\mathbf{r}\,\middle|\,\frac{1}{n}\sum_{i=1}^{n} f(r_i) \leq \delta, \frac{1}{n}\sum_{i=1}^{n} r_i = 1, \mathbf{r} \geq 0\right\}$. Then we have that

$$\textbf{\textit{If}}\ \mathcal{R}_{adv}(\theta_1) < 1,\ \ \textbf{\textit{then}}\ \mathcal{R}(\theta_1) < \mathcal{R}(\theta_2) \iff \mathcal{R}_{adv}(\theta_1) < \mathcal{R}_{adv}(\theta_2). \tag{8.5}$$

$$\textbf{\textit{If}}\ \mathcal{R}_{adv}(\theta_1) = 1,\ \ \textbf{\textit{then}}\ \mathcal{R}(\theta_1) \leq \mathcal{R}(\theta_2) \iff \mathcal{R}_{adv}(\theta_2) = 1. \tag{8.6}$$

The same monotonic relationship holds between their empirical approximation: $\widehat{\mathcal{R}}(\theta)$ and $\widehat{\mathcal{R}}_{adv}$.

Theorem 28 [77] shows that the monotonic relationship between the (empirical) risk and the (empirical) adversarial risk (worst-case risk) when 0-1 loss function is used. It means that minimizing (empirical) risk is equivalent to minimize the (empirical) adversarial risk (worst-case risk) for 0-1 loss. When we train a model based on the corrupted training distribution $p(x,y)$, we want our model to perform well on the clean distribution $q(x,y)$. Since we do not know the clean distribution q, we

want our model to perform well for the worst-case estimate of the clean distribution, with the assumption that the f-divergence between the corrupted distribution p and the clean distribution q is bounded by δ. Note that the underlying clean distribution is fixed but unknown, given the corrupted training distribution, the smallest δ that bounds the divergence between the corrupted distribution and clean distribution measures the intrinsic difficulty of the corruption, and it is also fixed and unknown. The corresponding worst-case distribution w.r.t the smallest δ is an estimate of the true clean distribution, and this worst-case risk upper bounds the risk of the true clean distribution. In addition, this bound is tighter than the other worst-case risks w.r.t larger δ. It is natural to use this upper bound as the objective for robust learning. When we use 0-1 loss (that is commonly employed for evaluation), because of the equivalence of the risk and the worst-case risk, we can directly minimize risk under training distribution p instead of directly minimizing the worst-case risk (i.e., the upper bound). Moreover, this enables us to minimize the upper bound without knowing the true δ beforehand. When the true δ is small, i.e., the corruption of the training data is not heavy, the upper bound is not too pessimistic. Usually, minimizing the upper bound can decrease the true risk under clean distribution. Particularly, when the clean distribution coincides with the worst-case estimate w.r.t the smallest δ, minimizing the risk under the corrupted training distribution leads to the same minimizer as minimizing the risk under the clean distribution.

8.2.2 Tighter upper bounds of the 0-1 Loss

Unlike commonly used loss functions in machine learning, the non-differentiability and zero gradients of the 0-1 loss make it difficult to optimize. We thus propose a tighter upper bound surrogate loss. We use the classification margin to define the 0-1 loss. For binary classification, classification margin is $u = \widehat{y}y$, where $\widehat{y}$ and $y \in \{+1, -1\}$ denotes the prediction and ground truth, respectively. (A simple multi-class extension is discussed in the Appendix.) Let $u_i \in \mathbb{R}$ be the classification margin of the i^{th} sample for $i \in \{1, ..., n\}$. Denote $\mathbf{u} = [u_1, ..., u_n]$. The 0-1 loss objective can be defined as follows:

$$J(\mathbf{u}) = \sum_{i=1}^{n} \mathbf{1}\big(u_i < 0\big). \tag{8.7}$$

Given a base upper bound function $l(u) \geq \mathbf{1}\big(u < 0\big), u \in \mathbb{R}$, the conventional surrogate of the 0-1 loss can be defined as

$$\widehat{J}(\mathbf{u}) = \sum_{i=1}^{n} l(u_i). \tag{8.8}$$

Our curriculum loss $Q(\mathbf{u})$ can be defined as Eq.(8.9). $Q(\mathbf{u})$ is a tighter upper bound of 0-1 loss $J(\mathbf{u})$ compared with the conventional surrogate loss $\widehat{J}(\mathbf{u})$, which is summarized in Theorem 29:

Theorem 29. ***(Tighter Bound)*** *Suppose that base loss function* $l(u) \geq \mathbf{1}\big(u < 0\big), u \in \mathbb{R}$ *is an upper bound of the 0-1 loss function. Let* $u_i \in \mathbb{R}$ *be the classification margin of the* i^{th} *sample for* $i \in \{1, ..., n\}$. *Denote* $\max(\cdot, \cdot)$ *as the maximum between two inputs. Let* $\mathbf{u} = [u_1, ..., u_n]$. *Define* $Q\left(\mathbf{u}\right)$ *as follows:*

$$Q\left(\mathbf{u}\right) = \min_{\mathbf{v}\in\{0,1\}^n} \max\big(\sum\nolimits_{i=1}^{n} v_i l(u_i), n - \sum\nolimits_{i=1}^{n} v_i + \sum\nolimits_{i=1}^{n} \mathbf{1}\big(u_i < 0\big)\big). \tag{8.9}$$

Then $J(\mathbf{u}) \leq Q\left(\mathbf{u}\right) \leq \widehat{\mathrm{J}}\left(\mathbf{u}\right)$ *holds true.*

Remark: For any fixed $\mathbf{u}$, we can obtain an optimum solution $\mathbf{v}^*$ of the partial optimization. The index indicator $\mathbf{v}^*$ can naturally select samples as a curriculum paradigm for training models. The partial optimization w.r.t index indicator $\mathbf{v}$ can be solved by a very simple and efficient algorithm (Algorithm 13) in $\mathcal{O}(n \log n)$. Thus, the loss is very efficient to compute. Moreover, since $Q\left(\mathbf{u}\right)$ is tighter than conventional surrogate loss $\widehat{J}(\mathbf{u})$, it is less sensitive to outliers compared with $\widehat{J}(\mathbf{u})$. Furthermore, it better preserves the robust property of the 0-1 loss against label corruption.

The difficulty of optimizing the 0-1 loss is that the 0-1 loss has zero gradients in almost everywhere (except at the breaking point). This issue prevents us from using first-order methods to optimize the 0-1 loss. Eq.(8.9) provides a surrogate of the 0-1 loss with non-zero subgradient for optimization, while preserving robust properties of the 0-1 loss. Note that our goal is to construct a tight upper bound of the 0-1 loss while maintaining informative (sub)gradients. Eq.(8.9) balances the 0-1 loss and conventional surrogate by selecting (the trust) samples (index) for training progressively.

Updating with all the samples at once is not efficient for deep models, while training with mini-batch is more efficient and well supported for many deep learning tools. We thus propose a batch based curriculum loss $\widehat{Q}(\mathbf{u})$ given as Eq.(8.10). We show that $\widehat{Q}(\mathbf{u})$ is also a tighter upper bound of 0-1 loss objective $J(\mathbf{u})$ compared with conventional loss $\widehat{J}(\mathbf{u})$. This property is summarized in Corollary 4.

Corollary 4. ***(Mini-batch Update)*** *Suppose that base loss function* $l(u) \geq \mathbf{1}\big(u < 0\big), u \in \mathbb{R}$ *is an upper bound of the 0-1 loss function. Let* b, m *be the number of batches and batch size, respectively. Let* $u_{ij} \in \mathbb{R}$ *be the classification margin of the* i^{th} *sample in*

batch j for $i \in \{1,...,m\}$ and $j \in \{1,...,b\}$. Denote $\mathbf{u} = [u_{11},...,u_{mb}]$. Let $n = mb$. Define $\widehat{Q}(\mathbf{u})$ as follows:

$$\widehat{Q}(\mathbf{u}) = \sum_{j=1}^{b} \min_{\mathbf{v}\in\{0,1\}^m} \max\Big(\sum_{i=1}^{m} v_{ij} l(u_{ij}), m - \sum_{i=1}^{m} v_{ij} + \sum_{i=1}^{m} \mathbf{1}\big(u_{ij} < 0\big)\Big). \tag{8.10}$$

Then $J(\mathbf{u}) \leq Q(\mathbf{u}) \leq \widehat{Q}(\mathbf{u}) \leq \widehat{\mathrm{J}}(\mathbf{u})$ holds true.

Remark: Corollary 4 shows that a batch-based curriculum loss is also a tighter upper bound of 0-1 loss $J(\mathbf{u})$ compared with the conventional surrogate loss $\widehat{J}(\mathbf{u})$. This enables us to train deep models with mini-batch update. Note that random shuffle in different epoch results in a different batch-based curriculum loss. Nevertheless, we at least know that all the induced losses are upper bounds of 0-1 loss objective and are tighter than $\widehat{J}(\mathbf{u})$. Moreover, all these losses are induced by the same base loss function $l(\cdot)$. Note that, our goal is to minimize the 0-1 loss. Random shuffle leads to a multiple surrogate training scheme. In addition, training deep models without shuffle does not have this issue.

We now present another curriculum loss $E(\mathbf{u})$ which is tighter than $Q(\mathbf{u})$. $E(\mathbf{u})$ is an (scaled) upper bound of 0-1 loss. This property is summarized as Theorem 30.

Theorem 30. ***(Scaled Bound)*** *Suppose that base loss function $l(u) \geq \mathbf{1}\big(u < 0\big), u \in \mathbb{R}$ is an upper bound of the 0-1 loss function. Let $u_i \in \mathbb{R}$ be the classification margin of the i^{th} sample for $i \in \{1,...,n\}$. Denote $\mathbf{u} = [u_1,...,u_n]$. Define $E(\mathbf{u})$ as follows:*

$$E(\mathbf{u}) = \min_{\mathbf{v}\in\{0,1\}^n} \max\Big(\sum_{i=1}^{n} v_i l(u_i), n - \sum_{i=1}^{n} v_i\Big). \tag{8.11}$$

Then $J(\mathbf{u}) \leq 2E(\mathbf{u}) \leq 2\widehat{\mathrm{J}}(\mathbf{u})$ holds true.

Remark: $E(\mathbf{u})$ has similar properties to $Q(\mathbf{u})$ discussed above. Moreover, it is tighter than $Q(\mathbf{u})$, i.e. $E(\mathbf{u}) \leq Q(\mathbf{u})$. Thus, it is less sensitive to outliers compared with $Q(\mathbf{u})$. However, $Q(\mathbf{u})$ can construct more adaptive curriculum by taking 0-1 loss into consideration during the training process.

Directly optimizing $E(\mathbf{u})$ is not as efficient as that optimizing $Q(\mathbf{u})$. We now present a batch loss objective $\widehat{E}(\mathbf{u})$ given as Eq.(8.12). $\widehat{E}(\mathbf{u})$ is also a tighter upper bound of 0-1 loss objective $J(\mathbf{u})$ compared with conventional surrogate loss $\widehat{J}(\mathbf{u})$.

Corollary 5. ***(Mini-batch Update for Scaled Bound)*** *Suppose that base loss function $l(u) \geq \mathbf{1}\big(u < 0\big), u \in \mathbb{R}$ is an upper bound of the 0-1 loss function. Let b, m be the number of batches and batch size, respectively. Let $u_{ij} \in \mathbb{R}$ be the classification*

Algorithm 13 Partial Optimization

Input: u_i for $i \in \{1, ..., n\}$, the selection threshold C;
Output: Index set $\mathbf{v} = (v_1, v_2, \ldots, v_n)$;
Compute the losses $l_i = l(u_i)$ for $i = 1, ..., n$;
Sort samples (index) w.r.t. the losses $\{l_i\}_{i=1}^n$ in a non-decreasing order; // Get $l_1 \leq \cdots \leq l_n$
Initialize $L_0 = 0$;
for $i = 1$ **to** n **do**
  $L_i = L_{i-1} + l_i$;
  if $L_i \leq (C + 1 - i)$ **then**
    Set $v_i = 1$;
  else
    Set $v_i = 0$;
  end if
end for

margin of the i^{th} *sample in batch* j *for* $i \in \{1, ..., m\}$ *and* $j \in \{1, ..., b\}$. *Denote* $\mathbf{u} = [u_{11}, ..., u_{mb}]$. *Let* $n = mb$. *Define* $\widehat{E}(\mathbf{u})$ *as follows:*

$$\widehat{E}\,(\mathbf{u}) = \sum_{j=1}^{b} \min_{\mathbf{v}\in\{0,1\}^m} \max\Big(\sum_{i=1}^{m} v_{ij} l(u_{ij}), m - \sum_{i=1}^{m} v_{ij}\Big). \tag{8.12}$$

Then $J(\mathbf{u}) \leq 2E\,(\mathbf{u}) \leq 2\widehat{E}\,(\mathbf{u}) \leq 2\widehat{\mathrm{J}}\,(\mathbf{u})$ *holds true.*

All the curriculum losses defined above rely on minimizing a partial optimization problem (Eq.(8.13)) to find the selection index set $\mathbf{v}^*$. We now show that the optimization of $\mathbf{v}$ with given classification margin $u_i \in \mathbb{R}, i \in \{1, ..., n\}$ can be done in $\mathcal{O}(n \log n)$.

Theorem 31. ***(Partial Optimization)*** *Suppose that base loss function* $l(u) \geq \mathbf{1}\big(u < 0\big), u \in \mathbb{R}$ *is an upper bound of the 0-1 loss function. For fixed* $u_i \in \mathbb{R}$, $i \in \{1, ..., n\}$, *an minimum solution* $\mathbf{v}^*$ *of the minimization problem in Eq. (8.13) can be achieved by Algorithm 13:*

$$\min_{\mathbf{v}\in\{0,1\}^n} \max\Big(\sum_{i=1}^{n} v_i l(u_i), C - \sum_{i=1}^{n} v_i\Big), \tag{8.13}$$

where C *is the threshold parameter such that* $0 \leq C \leq 2n$.

Remark: The time complexity of Algorithm 13 is $\mathcal{O}(n \log n)$. Moreover, it does not involve complex operations, and is very simple and efficient to compute.

Algorithm 13 can adaptively select samples for training. It has some useful properties to help us better understand the objective after partial minimization, we present them in Proposition 2.

Proposition 2. ***(Optimum of Partial Optimization)*** *Suppose that base loss function* $l(u) \geq \mathbf{1}\big(u < 0\big), u \in \mathbb{R}$ *is an upper bound of the 0-1 loss function. Let* $u_i \in \mathbb{R}$ *for* $i \in \{1, ..., n\}$ *be fixed values. Without loss of generality, assume* $l(u_1) \leq l(u_2) \cdots \leq l(u_n)$. *Let* $\mathbf{v}^*$ *be an optimum solution of the partial optimization problem in Eq.(8.13). Let* $T^* = \sum_{i=1}^{n} v_i^*$ *and* $L_{T^*} = \sum_{i=1}^{T^*} l(u_i)$. *Then we have*

$$L_{T^*} \leq C + 1 - T^* \tag{8.14}$$

$$L_{T^*+1} > C - T^* \tag{8.15}$$

$$L_{T^*+1} > \max(L_{T^*}, C - T^*) \tag{8.16}$$

$$\min_{\mathbf{v}\in\{0,1\}^n} \max\Big(\sum\nolimits_{i=1}^{n} v_i l(u_i), C - \sum\nolimits_{i=1}^{n} v_i\Big) = \max(L_{T^*}, C - T^*). \tag{8.17}$$

Remark: When $C \leq n + \sum_{i=1}^{n} \mathbf{1}\big(u_i < 0\big)$, Eq.(8.17) is tighter than the conventional loss $\widehat{J}(\mathbf{u})$. When $C \geq n$, Eq. (8.17) is a scaled upper bound of 0-1 loss $J(\mathbf{u})$. From Eq.(8.17) , we know the optimum of the partial optimization problem (8.13) (i.e. our objective) is $\max(L_{T^*}, C - T^*)$. When $L_{T^*} \geq C - T^*$, we can directly optimize L_{T^*} with the selected samples for training. When $L_{T^*} < C - T^*$, note that $L_{T^*+1} > \max(L_{T^*}, C - T^*)$ from Eq.(8.16), we can optimize L_{T^*+1} for training. Note that when $T^* < n$, we have that $L_{T^*+1} \leq L_n = \sum_{i=1}^{n} l(u_i)$, which is still tighter than the conventional loss $\widehat{J}(\mathbf{u})$. When $T^* = n$, for the parameter $C \leq n + \sum_{i=1}^{n} \mathbf{1}\big(u_i < 0\big)$, we have that $L_{T^*} = \widehat{J}(\mathbf{u}) \geq J(\mathbf{u}) \geq C - n = C - T^*$. Thus we can optimize $\max(L_{T^*}, C - T^*) = \widehat{J}(\mathbf{u})$. In practice, when training with random mini-batch, we find that optimizing L_{T^*} in both cases instead of L_{T^*+1} does not make much influence.

8.2.3 Noise Pruned Curriculum Loss

The curriculum loss in Eq.(8.9) and Eq.(8.11) expect to minimize the upper bound of the 0-1 loss for all the training samples. When model capability (complexity) is high, (deep network) model will still attain small (zero) training loss and overfit to the noisy samples.

The ideal model is that it correctly classifies the clean training samples and misclassifies the noisy samples with wrong labels. Suppose that the rate of noisy samples (by label corruption) is $\epsilon \in [0, 1]$. The ideal model is to correctly classify the $(1-\epsilon)n$ clean training samples, and misclassify the ϵn noisy training samples. This is because the label is corrupted. Correctly classify the training samples with corrupted (wrong) label means that the model has already overfitted to noisy samples. This will harm the generalization to the unseen data.

Considering all the above reasons, we thus propose the Noise Pruned Curriculum Loss (NPCL) as

$$\mathcal{L}(\mathbf{u}) = \min_{\mathbf{v}\in\{0,1\}^n} \max\big(\sum\nolimits_{i=1}^{n} v_i l(u_i), C - \sum\nolimits_{i=1}^{n} v_i\big), \tag{8.18}$$

where $C = (1-\epsilon)n$ or $C = (1-\epsilon)^2 n + (1-\epsilon)\sum_{i=1}^{n} \mathbf{1}\big(u_i < 0\big)$.

When we know there are ϵn noisy samples in the training set, we can leverage this as our prior. (The impact of misspecification of the prior is included in the supplement.) When $C = (1-\epsilon)n$ (assume $C, \epsilon n$ are integers for simplicity), from the selection procedure in Algorithm 13, we know ϵn[1] samples with largest losses $l(u)$ will be pruned. This is because $C - \sum_{i=1}^{n} v_i + 1 \leq 0$ when $\sum_{i=1}^{n} v_i \geq (1-\epsilon)n + 1$. Without loss of generality, assume $l(u_1) \leq l(u_2) \cdots \leq l(u_n)$. After pruning, we have $v_{(1-\epsilon)n+1} = \cdots = v_n = 0$, the pruned loss becomes

$$\widetilde{\mathcal{L}}(\mathbf{u}) = \min_{\mathbf{v}\in\{0,1\}^{(1-\epsilon)n}} \max\big(\sum\nolimits_{i=1}^{(1-\epsilon)n} v_i l(u_i), (1-\epsilon)n - \sum\nolimits_{i=1}^{(1-\epsilon)n} v_i\big). \tag{8.19}$$

It is the basic CL for $(1-\epsilon)n$ samples and it is the upper bound of $\sum_{i=1}^{(1-\epsilon)n} \mathbf{1}\big(u_i < 0\big)$. If we prune more noisy samples than clean samples, it will reduce the noise ratio. Then the basic CL can handle. Fortunately, this assumption is supported by the "memorization" effect in deep networks [11], i.e. deep networks tend to learn clean and easy pattern first. Thus, the loss of noisy or hard data tend to remain high for a period (before being overfitted). Therefore, the pruned samples with largest loss are more likely to be the noisy samples. After the rough pruning, the problem becomes optimizing basic CL for the remaining samples as in Eq.(8.19). Note that our CL is a tight upper bound approximation to the 0-1 loss, it preserves the robust property to some extent. Thus, it can handle case with small noise rate. Specifically, our CL(Eq.8.19) further select samples from the remaining samples for training adaptively according to the state of training process. This generally will further reduce the noise ratio. Thus, we may expect our NPCL to be robust to noisy samples. Note that, all the above can be done by the simple and efficient Algorithm 13 without explicit pruning samples in a separated step. Namely, our loss can do all these automatically under a unified objective form in Eq.(8.18).

When $C = (1-\epsilon)n$, the NPCL in Eq.(8.18) reduces to basic CL $E(\mathbf{u})$ in Eq.(8.11) with $\epsilon = 0$. When $C = (1-\epsilon)^2 n + (1-\epsilon)\sum_{i=1}^{n} \mathbf{1}\big(u_i < 0\big)$, for an ideal target model (that misclassifies noisy samples only), we know that $\mathbb{E}[C] = (1-\epsilon)^2 n + (1-\epsilon)\mathbb{E}[\sum_{i=1}^{n} \mathbf{1}\big(u_i < 0\big)] = (1-\epsilon)^2 n + (1-\epsilon)\epsilon n = (1-\epsilon)n$. It has similar properties as

[1] When $\sum_{i=1}^{(1-\epsilon)n+1} l(u_i) \neq 0$, ϵn samples will be pruned. Otherwise, $\epsilon n - 1$ samples will be pruned.

Algorithm 14 Training with Batch Noise Pruned Curriculum Loss

Input: Number of epochs N, batch size m, noise ratio ϵ;
Output: The model parameter $\mathbf{w}$;
Initialize model parameter $\mathbf{w}$.
for $k = 1$ **to** N **do**
  Shuffle training set $\mathcal{D}$;
  while Not fetch all the data from $\mathcal{D}$ **do**
    Fetch a mini-batch $\widehat{\mathcal{D}}$ from $\mathcal{D}$;
    Compute losses $\{l_i\}_{i=1}^m$ for data in $\widehat{\mathcal{D}}$;
    Compute the selection threshold C according to Eq.(8.21).
    Compute selection index $\mathbf{v}^*$ by Algorithm 13;
    Update $\mathbf{w} = \mathbf{w} - \alpha \nabla l \left(\widehat{\mathcal{D}}_{\mathbf{v}^*} \right)$ w.r.t the subset $\widehat{\mathcal{D}}_{\mathbf{v}^*}$ of $\widehat{\mathcal{D}}$ selected by $\mathbf{v}^*$;
  end while
end for

choosing $C = (1 - \epsilon)n$. Moreover, it is more adaptive by considering 0-1 loss during training at different stages. In this case, the NPCL in Eq.(8.18) reduces to the CL $Q(\mathbf{u})$ in Eq.(8.9) when $\epsilon = 0$. Note that C is a prior, users can defined it based on their domain knowledge.

To leverage the benefit of deep learning, we present the batched NPCL as

$$\widehat{\mathcal{L}}\left(\mathbf{u}\right) = \sum\nolimits_{j=1}^{b} \min_{\mathbf{v}\in\{0,1\}^m} \max\left(\sum\nolimits_{i=1}^{m} v_{ij} l(u_{ij}), \widehat{C}_j - \sum\nolimits_{i=1}^{m} v_{ij} \right), \tag{8.20}$$

where $\widehat{C}_j = (1 - \epsilon)m$ or as in Eq.(8.21):

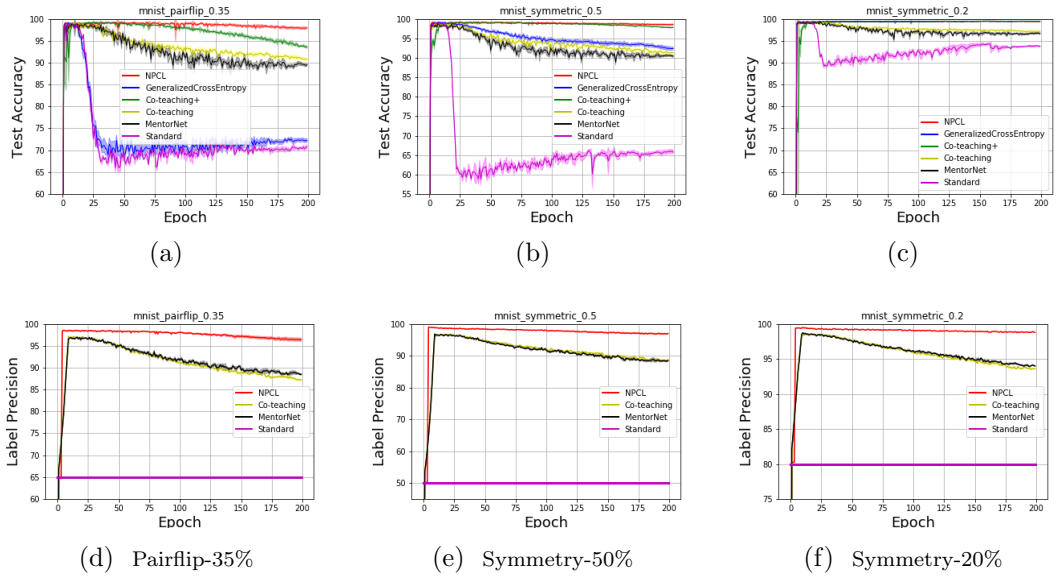
$$\widehat{C}_j = (1 - \epsilon)^2 m + (1 - \epsilon) \sum\nolimits_{i=1}^{m} \mathbf{1}\big(u_{ij} < 0\big). \tag{8.21}$$

Similar to Corollary 4, we know that $\mathcal{L}\left(\mathbf{u}\right) \leq \widehat{\mathcal{L}}\left(\mathbf{u}\right)$. Thus, optimizing the batched NPCL is indeed minimizing the upper bound of NPCL. This enables us to train the model with mini-batch update, which is very efficient for modern deep learning tools. The training procedure is summarized in Algorithm 14. It uses Algorithm 13 to select a subset of samples from every mini-batch. Then, it uses the selected samples to perform gradient update.

8.3 Empirical Study

8.3.1 Evaluation of Robustness against Label Corruption

We evaluate our NPCL by comparing Generalized Cross-Entropy (GCE) loss [188], Co-teaching [67], Co-teaching+ [182], MentorNet [84] and standard network training on MNIST, CIFAR10 and CIFAR100 dataset as in [59, 67, 138]. Two types of



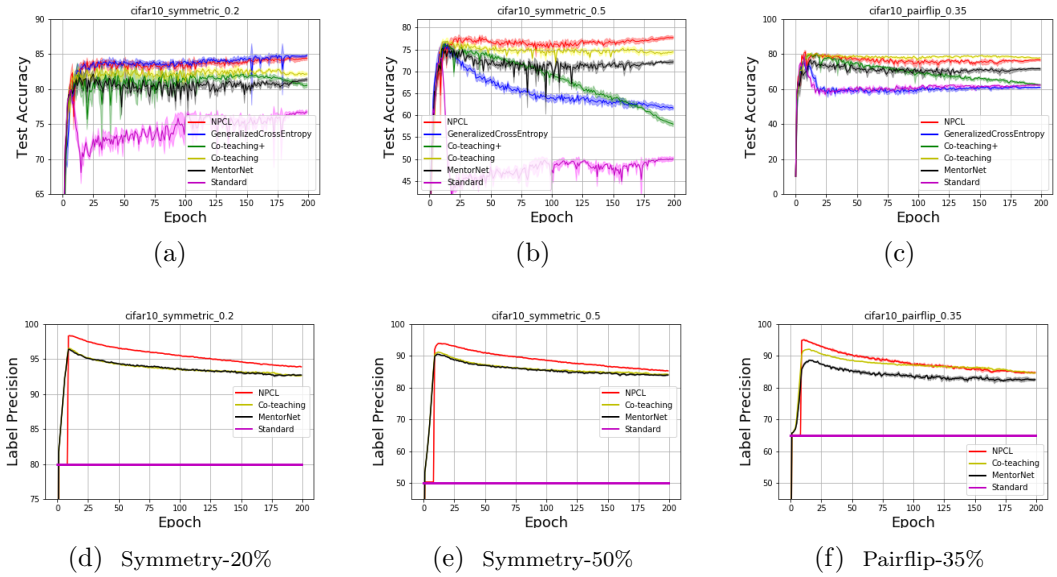
(a) (b) (c)

(d) Pairflip-35% (e) Symmetry-50% (f) Symmetry-20%

Figure 8.1: Test accuracy and label precision vs. number of epochs on MNIST dataset.

random label corruption, i.e. Symmetry flipping [167] and Pair flipping [66], are considered in this work. Symmetry flipping is that the corrupted label is uniformly assign to one of $K-1$ incorrect classes. Pair flipping is that the corrupted label is assign to one specific class similar to the ground truth. The noise rate ϵ of label flipping is chosen from $\{20\%, 50\%, 35\%\}$ as a representative. As a robust loss function, we further compare NPCL with GCE loss in detail with noise rate in $\{0\%, 10\%, 20\%, 30\%, 40\%, 50\%\}$. We employ same network architecture and network hyperparameters as in Co-teaching [67] for all the methods in comparison. Specifically, the batch size and the number of epochs is set to $m = 128$ and $N = 200$, respectively. The Adam optimizer with the same parameter as [67] is employed. For NPCL, we employ hinge loss as the base upper bound function of 0-1 loss. In the first few epochs, we train model using full batch with soft hinge loss (in the supplement) as a burn-in period suggested in [84]. Specifically, we start NPCL at 5^{th} epoch on MNIST and 10^{th} epoch on CIFAR10 and CIFAR100, respectively. For Co-teaching [67] and MentorNet in [84], we employ the open sourced code of Co-teaching [67]. For Co-teaching+ [182], we employ the code provided by the authors. We implement NPCL by Pytorch. For NPCL, Co-teaching and Co-teaching+, we employ the true noise rate as parameter. Experiments are performed five independent runs. The error bar for STD is shaded.

For performance measurements, we employ both test accuracy and label precision as in [67]. Label precision is defined as : *number of clean samples / number of selected*



(a) (b) (c)

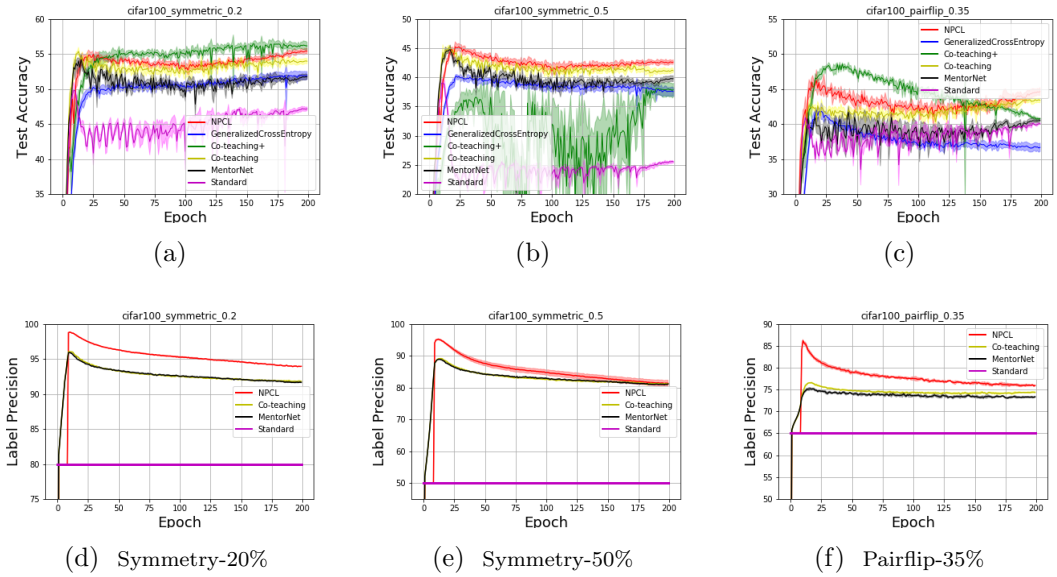
(d) Symmetry-20% (e) Symmetry-50% (f) Pairflip-35%

Figure 8.2: Test accuracy and label precision vs. number of epochs on CIFAR10 dataset.

samples, which measures the selection accuracy for sample selection based methods. A higher label precision in the mini-batch after sample selection can lead to a update with less noisy samples, which means that model suffers less influence of noisy samples and thus preforms more robustly to label corruption.

The pictures of test accuracy and label precision vs. number of epochs on MNIST are presented in Figure 8.1. The results on CIFAR10 and CIFAR100 are shown in Figure 8.2 and Figure 8.3, respectively. It shows that NCPL achieves superior performance compared with GCE loss in terms of test accuracy. Particularly, NPCL obtains significant better performance compared with GCE loss in hard cases: Symmetry-50% and Pair-flip-35%, which shows that NPCL is more robust to label corruption compared with GCE loss. Moreover, NPCL obtains better performance on MNIST, and competitive performance on CIFAR10 and CIFAR100 compared with Co-teaching. Furthermore, NPCL achieves better performance than Co-teaching+ on CIFAR10 and two cases on MNIST. In addition, we find that Co-teaching+ is not stable on CIFAR100 with 50% symmetric noise. Note that NPCL is a simple plug-in for a single network, while Co-teaching/Co-teaching+ employs two networks to train the model concurrently. Thus, both the space complexity and time complexity of Co-teaching/Co-teaching+ is doubled compared with our NPCL.

Both our NPCL and Generalized Cross Entropy (GCE) loss are robust loss functions as plug-in for single network. Thus, we provide a more detailed comparison between our NPCL and GCE loss with noise rate in $\{0\%, 10\%, 20\%, 30\%, 40\%, 50\%\}$.



(a) (b) (c)

(d) Symmetry-20% (e) Symmetry-50% (f) Pairflip-35%

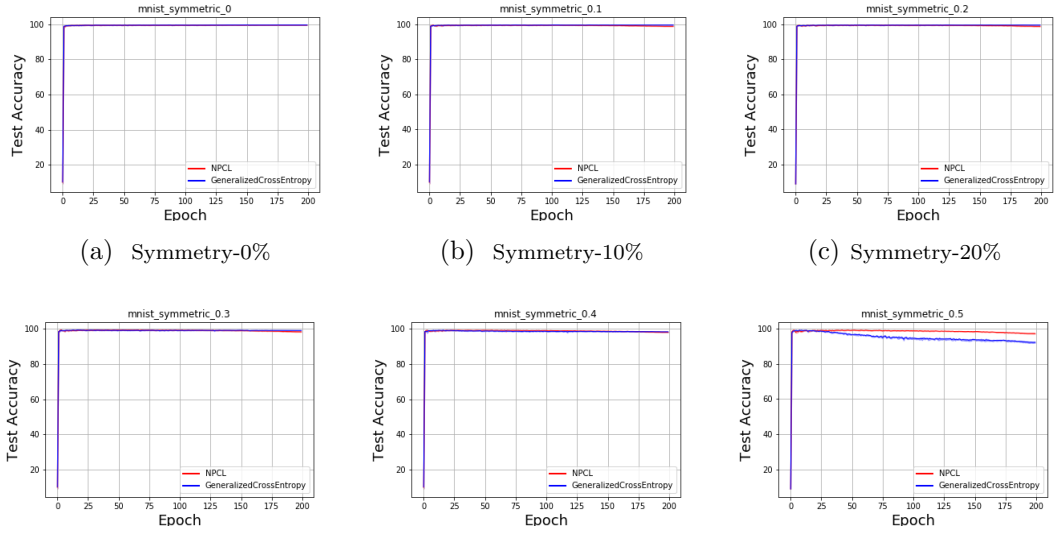
Figure 8.3: Test accuracy and label precision vs. number of epochs on CIFAR100 dataset.

The experimental results on CIFAR10 are presented in Figure 8.7. The experimental results on CIFAR100 and MNIST are provided in Figure 8.5 and Figure 8.4. From Figure 8.7, Figure 8.5 and Figure 8.4, we can observe that NPCL obtains similar and higher test accuracy in all the cases. Moreover, from Figure 8.7 and Figure 8.4, we can see that NPCL achieves similar test accuracy compared with the GCE loss when the noise rate is small. The improvement increases with the increase of the noise rate. Particularly, NPCL obtains remarkable improvement compared with the GCE loss on CIFAR10 with noise rate 50%. It shows that NPCL is more robust compared with GCE loss against label corruption. GCE loss employs all samples for training, while NPCL prunes the noisy samples adaptively. As a result, GCE loss still employs samples with wrong labels for training, which misleads the model. Thus, NPCL obtains better performance when the noise rate becomes large.

8.3.2 More experiments with different network architectures

We follow the experiments setup in [110]. We use the online code of [110] , and only change the loss for comparison. We cite the numbers of Softmax, RoG and D2L [120] in [110] for comparison.

The test accuracy results on uniform noise, semantic noise and open-set noise are shown in Table 8.1, Table 8.2 and Table 8.3, respectively. From Table 8.1, we can observe that both NPCL and CL outperforms Softmax (cross-entropy) and RoG



(a) Symmetry-0% (b) Symmetry-10% (c) Symmetry-20%

(d) Symmetry-30% (e) Symmetry-40% (f) Symmetry-50%

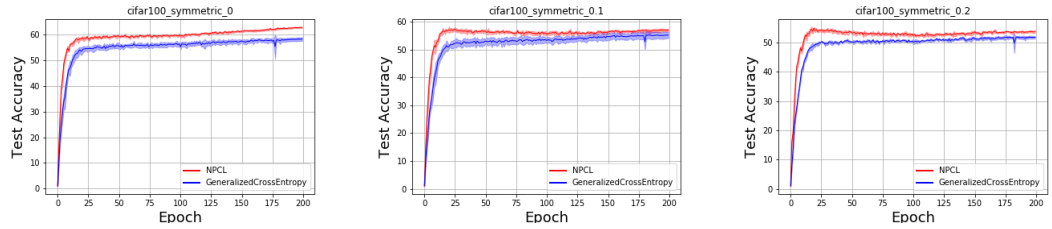
Figure 8.4: Test accuracy vs. number of epochs on MNIST dataset.

(cross-entropy) on five cases for uniform noise. Note that RoG is an ensemble method, while CL/NPCL is a single loss for network training, one can combine them to boost the performance. From Table 8.2, we can see that CL obtains consistently better performance than cross-entropy and D2L [120] for the semantic noise. Table 8.3 shows that NPCL achieves competitive performance compared with RoG for open-set noise.

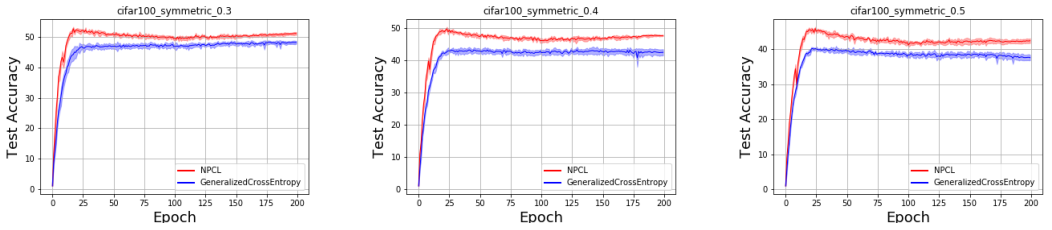
Table 8.1: Test accuracy(%) of DenseNet on CIFAR10 and CIFAR100.

Noise type	CIFAR10				CIFAR100			
	NPCL	CL	Softmax	RoG	NPCL	CL	Softmax	RoG
uniform (20%)	**89.49**	89.32	81.01	87.41	64.88	**67.92**	61.72	64.29
uniform (40%)	83.24	**85.57**	72.34	81.83	56.34	**58.63**	50.89	55.68
uniform (60%)	66.2	68.52	55.42	**75.45**	44.49	**46.65**	38.33	44.12

We further evaluate the performance of CL/NPCL on the Tiny-ImageNet dataset. We use the ResNet18 network as the test-bed. For GCE loss, we employ the default hyper-parameter $q = 0.7$ in all cases. All the methods are performed five runs with seeds $\{1, 2, 3, 4, 5\}$. The curve of mean test accuracy (shaded in std) are provided in Figure 8.6. We can see that NPCL and CL obtain higher test accuracy than generalized cross-entropy loss and stand cross-entropy loss on both cases. Note that CL does not have parameters, it is much convenient to use.



(a) Symmetry-0% (b) Symmetry-10% (c) Symmetry-20%



(d) Symmetry-30% (e) Symmetry-40% (f) Symmetry-50%

Figure 8.5: Test accuracy vs. number of epochs on CIFAR100 dataset.

Table 8.2: Test accuracy(%) of DenseNet on CIFAR10 and CIFAR100 with semantic noise.

Dataset	Label generator (noise rate)	NPCL	CL	Cross-entropy	D2L
CIFAR10	DenseNet(32%)	66.5	**67.45**	67.24	66.91
	ResNet(38%)	61.88	**62.88**	62.26	59.10
	VGG(34%)	68.37	**69.61**	68.77	57.97
CIFAR100	DenseNet(34%)	**57.59**	55.14	50.72	5.00
	ResNet(37%)	**54.49**	53.20	50.68	23.71
	VGG(37%)	**55.41**	52.71	51.08	40.97

8.3.3 Impact of Misspecified Estimation of Noise Rate ϵ

When the noise rate ϵ is not known as a prior, we can select the parameter ϵ by cross-validation. A more interesting method is to set ϵ as a parameter, and we update ϵ automatically by minimizing a loss on a validation set.

We empirically analyze the impact of misspecified prior for the noise rate ϵ. The average test accuracy over last ten epochs on MNIST for different priors are reported in Table 8.4. We can observe that NPCL is robust to misspecified prior for small noise cases (Symmetry-20%). Moreover, it becomes a bit more sensitive on large noise case (Symmetry-50%) and on the pair flipping case (Pair-35%).

Table 8.3: Test accuracy(%) of DenseNet on CIFAR10 with open-set noise.

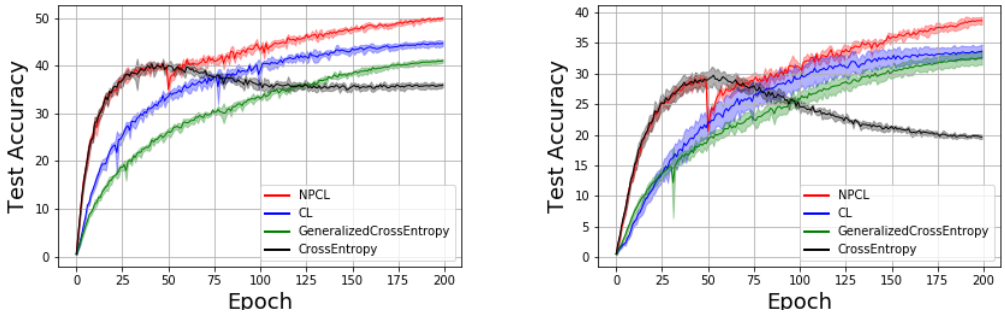
Open-set Data	NPCL	Softmax	RoG
CIFAR100	82.85	79.01	**83.37**
ImageNet	**87.95**	86.88	87.05
CIFAR100-ImageNet	84.28	81.58	**84.35**

Table 8.4: Average test accuracy of NPCL with different ϵ on MNIST over last ten epochs

Flipping Rate	0.5ϵ	0.75ϵ	ϵ	1.25ϵ	1.5ϵ
Symmetry-20%	96.31% ± 0.17%	97.72% ± 0.09%	99.41% ± 0.01%	**99.55% ± 0.02%**	99.10% ± 0.04%
Symmetry-50%	78.67% ± 0.36%	87.36% ± 0.29%	**98.53% ± 0.02%**	97.92% ± 0.06%	67.61% ± 0.06%
Pair-35%	80.59% ± 0.40%	87.86% ± 0.48%	97.90% ± 0.04%	**99.33% ± 0.02%**	86.66% ± 0.08%

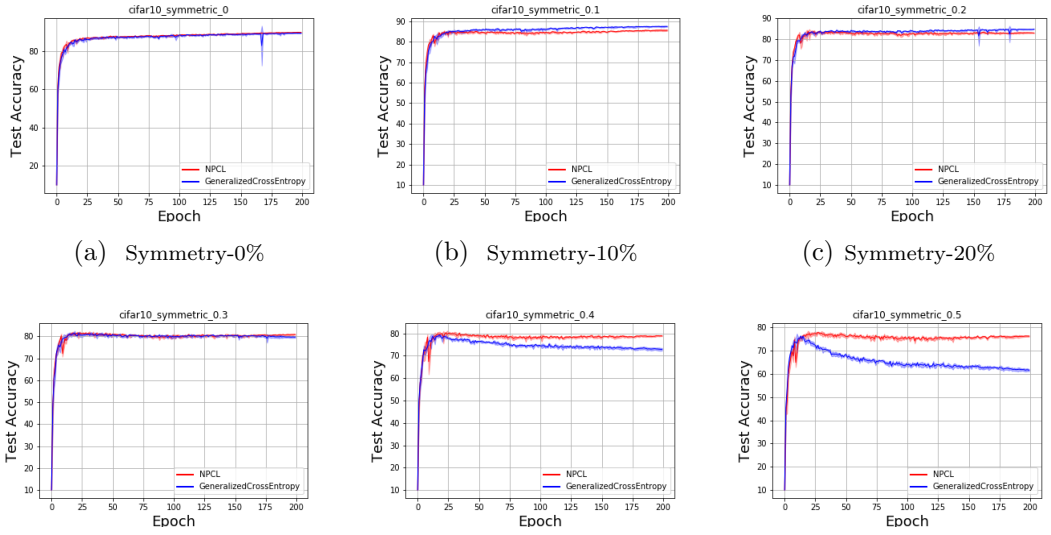
8.4 Summary

In this work, we proposed a curriculum loss (CL) for robust learning. Theoretically, we analyzed the properties of CL and proved that it is tighter upper bound of the 0-1 loss compared with conventional summation based surrogate losses. We extended our CL to a more general form (NPCL) to handle large rate of label corruption. Empirically, experimental results on benchmark datasets show the robustness of the proposed loss. As a further work, we may improve our CL to handle imbalanced distribution by considering diversity for each class. Moreover, it is interesting to investigate the influence of different base loss functions in CL and NPCL.



(a) Symmetric-20% (b) Symmetric-50%

Figure 8.6: Test accuracy (%) on Tiny-ImageNet dataset with symmetric noise



(a) Symmetry-0% (b) Symmetry-10% (c) Symmetry-20%

(d) Symmetry-30% (e) Symmetry-40% (f) Symmetry-50%

Figure 8.7: Test accuracy vs. number of epochs on CIFAR10 dataset.

Chapter 9

Conclusion

In this thesis, we investigate black-box integral approximation and black-box optimization by considering the closed relationship between them. For integral approximation, we develop a simple closed-form rank-1 lattice construction method based on group theory (Chapter 5).. Our method reduces the number of distinct pairwise distance values to generate a more regular lattice. Furthermore, we investigate structured points set for integral approximation on hyper-sphere (Chapter 5 and Chapter 6). Our structured point sets can serve as a good initialization for black-box optimization. Moreover, we propose stochastic black-box optimization with implicit natural gradients for black-box optimization (Chapter 3). Our method is very simple and has only the step-size hyper-parameter. Furthermore, we develop a batch Bayesian optimization algorithm from the perspective of frequentist kernel methods, which is powerful for low-dimensional black-box optimization problems (Chapter 4). We further apply our structured integral approximation techniques for kernel approximation (Chapter 6). In addition, we develop structured approximation for robust deep neural network architecture, which results in an elegant and simple architecture that preserves optimization properties (Chapter 7). Moreover, we develop adaptive loss as a tighter upper bound approximation for expected 0-1 risk, robust and trainable with SGD (Chapter 8).

Chapter 10

Appendix

10.1 Proof of Theorem 2

Proof. For Gaussian distribution $p := \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, the gradient of $\mathbb{E}_p[f(\boldsymbol{x})]$ w.r.t $\boldsymbol{\mu}$ can be derived as follows:

$$\nabla_{\boldsymbol{\mu}} \mathbb{E}_p[f(\boldsymbol{x})] = \mathbb{E}_p[f(\boldsymbol{x}) \nabla_{\boldsymbol{\mu}} \log(p(\boldsymbol{x}; \boldsymbol{\mu}, \Sigma))] \tag{10.1}$$

$$= \mathbb{E}_p \left[f(\boldsymbol{x}) \nabla_{\boldsymbol{\mu}} \left[-\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu})^\top \Sigma^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \right] \right] \tag{10.2}$$

$$= \mathbb{E}_p \left[\Sigma^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) f(\boldsymbol{x}) \right] \tag{10.3}$$

The gradient of $\mathbb{E}_p[f(\boldsymbol{x})]$ w.r.t Σ can be derived as follows:

$$\nabla_{\Sigma} \mathbb{E}_p[f(\boldsymbol{x})] = \mathbb{E}_p[f(\boldsymbol{x}) \nabla_{\Sigma} \log(p(\boldsymbol{x}; \boldsymbol{\mu}, \Sigma))] \tag{10.4}$$

$$= \mathbb{E}_p \left[f(\boldsymbol{x}) \nabla_{\Sigma} \left[-\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu})^\top \Sigma^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) - \frac{1}{2} \log \det(\Sigma) \right] \right] \tag{10.5}$$

$$= \frac{1}{2} \mathbb{E}_p \left[\left(\Sigma^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) (\boldsymbol{x} - \boldsymbol{\mu})^\top \Sigma^{-1} - \Sigma^{-1} \right) f(\boldsymbol{x}) \right] \tag{10.6}$$

□

10.2 Proof of Theorem 3

Proof. For Guassian distribution with parameter $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \Sigma\}$, problem (3.32) can be rewrited as

$$\left\langle \boldsymbol{\theta}, \nabla_{\boldsymbol{\theta}} \bar{J}(\boldsymbol{\theta}_t) \right\rangle + \frac{1}{\beta_t} \mathrm{KL}\left(p_\theta \| p_{\theta_t}\right) = \boldsymbol{\mu}^\top \nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t) + \mathrm{tr}(\Sigma \nabla_{\Sigma} \bar{J}(\boldsymbol{\theta}_t)) + \frac{1}{2\beta_t} \left[\mathrm{tr}(\Sigma_t^{-1} \Sigma) + (\boldsymbol{\mu} - \boldsymbol{\mu}_t)^\top \Sigma_t^{-1} (\boldsymbol{\mu} - \boldsymbol{\mu}_t) + \log \frac{|\Sigma_t|}{|\Sigma|} - d \right] \tag{10.7}$$

where $\nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t)$ denotes the derivative w.r.t $\boldsymbol{\mu}$ taking at $\boldsymbol{\mu} = \boldsymbol{\mu}_t, \Sigma = \Sigma_t$. $\nabla_{\Sigma} \bar{J}(\boldsymbol{\theta}_t)$ denotes the derivative w.r.t Σ taking at $\boldsymbol{\mu} = \boldsymbol{\mu}_t, \Sigma = \Sigma_t$. Note that $\nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t)$ and $\nabla_{\Sigma} \bar{J}(\boldsymbol{\theta}_t)$ are not functions now.

From Eq.(10.7), we can see that the problem is convex with respect to $\boldsymbol{\mu}$ and Σ. Taking the derivative of (10.7) w.r.t $\boldsymbol{\mu}$ and Σ, and setting them to zero, we can obtain that

$$\nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t) + \frac{1}{\beta_t} \Sigma_t^{-1} (\boldsymbol{\mu} - \boldsymbol{\mu}_t) = \mathbf{0} \tag{10.8}$$

$$\nabla_{\Sigma} \bar{J}(\boldsymbol{\theta}_t)) + \frac{1}{2\beta_t} \left[\Sigma_t^{-1} - \Sigma^{-1} \right] = \mathbf{0} \tag{10.9}$$

It follows that

$$\boldsymbol{\mu} = \boldsymbol{\mu}_t - \beta_t \Sigma_t \nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t) \tag{10.10}$$

$$\Sigma^{-1} = \Sigma_t^{-1} + 2\beta_t \nabla_{\Sigma} \bar{J}(\boldsymbol{\theta}_t) \tag{10.11}$$

By definition, $\boldsymbol{\mu}_{t+1}$ and Σ_{t+1} are the optimum of this convex optimization problem. Thus, we achieve that

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \Sigma_t \nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t) \tag{10.12}$$

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + 2\beta_t \nabla_{\Sigma} \bar{J}(\boldsymbol{\theta}_t) \tag{10.13}$$

□

10.3 Proof of Theorem 4

Lemma 1. *For Gaussian distribution with parameter* $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \Sigma\} \in \boldsymbol{\Theta}$. *Let* $F_t(\boldsymbol{m}) = \beta_t \langle \boldsymbol{m}, \widehat{v}_t \rangle$ *for all* $t \geq 1$, *where* $\boldsymbol{m} := \{\boldsymbol{m}_1, \boldsymbol{m}_2\} = \{\boldsymbol{\mu}, \Sigma + \boldsymbol{\mu}\boldsymbol{\mu}^\top\} \in \boldsymbol{\mathcal{M}}$, $\boldsymbol{\mathcal{M}}$ *denotes a convex set. Let* $\boldsymbol{m}^{t+1}$ *as the solution of*

$$\boldsymbol{m}^{t+1} = \arg\min_{\boldsymbol{m} \in \boldsymbol{\mathcal{M}}} F_t(\boldsymbol{m}) + KL\left(p_{\boldsymbol{m}} \| p_{\boldsymbol{m}^t}\right) \tag{10.14}$$

Then, for $\forall \boldsymbol{m} \in \boldsymbol{\mathcal{M}}$, *we have*

$$F(\boldsymbol{m}) + KL\left(p_{\boldsymbol{m}} \| p_{\boldsymbol{m}^t}\right) \geq F(\boldsymbol{m}^{t+1}) + KL\left(p_{\boldsymbol{m}^{t+1}} \| p_{\boldsymbol{m}^t}\right) + KL\left(p_{\boldsymbol{m}} \| p_{\boldsymbol{m}^{t+1}}\right) \tag{10.15}$$

Proof. Since KL-divergence of Gaussian is a Bregman divergence associated with base function $A^*(\boldsymbol{m})$ w.r.t mean parameter $\boldsymbol{m}$, we know problem in Eq.(10.14) is convex. Since $\boldsymbol{m}^{t+1}$ is the optimum of the convex optimization problem in Eq.(10.14), we have that

$$\left\langle \beta_t \widehat{v}_t + \nabla_{\boldsymbol{m}=\boldsymbol{m}^{t+1}} \mathrm{KL}\left(p_{\boldsymbol{m}} \| p_{\boldsymbol{m}^t}\right), \boldsymbol{m} - \boldsymbol{m}^{t+1} \right\rangle \geq 0, \forall \boldsymbol{m} \in \boldsymbol{\mathcal{M}} \tag{10.16}$$

Note that $\nabla_{\boldsymbol{m}=\boldsymbol{m}^{t+1}}\mathrm{KL}\left(p_{\boldsymbol{m}}\|p_{\boldsymbol{m}^t}\right)=\nabla A^*(\boldsymbol{m}^{t+1})-\nabla A^*(\boldsymbol{m}^t)$. For $\forall \boldsymbol{m}\in\boldsymbol{\mathcal{M}}$ we have that

$$
\begin{aligned}
F(\boldsymbol{m}) &= \beta_t\left\langle \widehat{v}_t, \boldsymbol{m}^{t+1}\right\rangle + \left\langle \beta_t\widehat{v}_t, \boldsymbol{m}-\boldsymbol{m}^{t+1}\right\rangle && (10.17)\\
&\geq \beta_t\left\langle \widehat{v}_t, \boldsymbol{m}^{t+1}\right\rangle - \left\langle \nabla A^*(\boldsymbol{m}^{t+1})-\nabla A^*(\boldsymbol{m}^t), \boldsymbol{m}-\boldsymbol{m}^{t+1}\right\rangle && (10.18)
\end{aligned}
$$

Rewritten the term $-\left\langle \nabla A^*(\boldsymbol{m}^{t+1})-\nabla A^*(\boldsymbol{m}^t), \boldsymbol{m}-\boldsymbol{m}^{t+1}\right\rangle$, we have that

$$
\begin{aligned}
&-\left\langle \nabla A^*(\boldsymbol{m}^{t+1})-\nabla A^*(\boldsymbol{m}^t), \boldsymbol{m}-\boldsymbol{m}^{t+1}\right\rangle\\
&= A^*(\boldsymbol{m}^{t+1})-A^*(\boldsymbol{m}^t)-\left\langle \nabla A^*(\boldsymbol{m}^t), \boldsymbol{m}^{t+1}-\boldsymbol{m}^t\right\rangle && (10.19)\\
&\quad - A^*(\boldsymbol{m})+A^*(\boldsymbol{m}^t)+\left\langle \nabla A^*(\boldsymbol{m}^t), \boldsymbol{m}-\boldsymbol{m}^t\right\rangle && (10.20)\\
&\quad + A^*(\boldsymbol{m})-A^*(\boldsymbol{m}^{t+1})-\left\langle \nabla A^*(\boldsymbol{m}^{t+1}), \boldsymbol{m}-\boldsymbol{m}^{t+1}\right\rangle && (10.21)\\
&= \mathrm{KL}\left(p_{\boldsymbol{m}^{t+1}}\|p_{\boldsymbol{m}^t}\right)-\mathrm{KL}\left(p_{\boldsymbol{m}}\|p_{\boldsymbol{m}^t}\right)+\mathrm{KL}\left(p_{\boldsymbol{m}}\|p_{\boldsymbol{m}^{t+1}}\right) && (10.22)
\end{aligned}
$$

Plug Eq.(10.22) into (10.18), we obtain that

$$
F(\boldsymbol{m})+\mathrm{KL}\left(p_{\boldsymbol{m}}\|p_{\boldsymbol{m}^t}\right)\geq F(\boldsymbol{m}^{t+1})+\mathrm{KL}\left(p_{\boldsymbol{m}^{t+1}}\|p_{\boldsymbol{m}^t}\right)+\mathrm{KL}\left(p_{\boldsymbol{m}}\|p_{\boldsymbol{m}^{t+1}}\right) \quad (10.23)
$$

□

Lemma 2. *Let $\widehat{v}_t=\{\widehat{g}_t-2\widehat{G}_t\boldsymbol{\mu}_t, \widehat{G}_t\}$, updating parameter as (10.14), then we have*

$$
\frac{1}{2}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\|^2_{\Sigma^{-1}_{t+1}}\leq \frac{1}{2}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\Sigma^{-1}_t}+\beta_t\left\langle \widehat{g}_t, \boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\right\rangle-\frac{1}{2}\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\Sigma^{-1}_{t+1}}+\beta_t\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\widehat{G}_t} \quad (10.24)
$$

Proof. First, recall that the KL-divergence is defined as

$$
\mathrm{KL}\left(p_{\boldsymbol{m}}\|p_{\boldsymbol{m}^t}\right)=\frac{1}{2}\left\{\|\boldsymbol{\mu}-\boldsymbol{\mu}_t\|^2_{\Sigma^{-1}_t}+\mathbf{tr}\left(\Sigma\Sigma_t^{-1}\right)+\log\frac{|\Sigma_t|}{|\Sigma|}-d\right\} \quad (10.25)
$$

From Lemma 1, we know that

$$
\mathrm{KL}\left(p_{\boldsymbol{m}^*}\|p_{\boldsymbol{m}^{t+1}}\right)\leq \mathrm{KL}\left(p_{\boldsymbol{m}^*}\|p_{\boldsymbol{m}^t}\right)-\mathrm{KL}\left(p_{\boldsymbol{m}^{t+1}}\|p_{\boldsymbol{m}^t}\right)+F(\boldsymbol{m}^*)-F(\boldsymbol{m}^{t+1}) \quad (10.26)
$$

It follows that

$$
\begin{aligned}
&\frac{1}{2}\left\{\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\|^2_{\Sigma^{-1}_{t+1}}+\mathbf{tr}\left(\Sigma^*\Sigma^{-1}_{t+1}\right)+\log\frac{|\Sigma_{t+1}|}{|\Sigma^*|}\right\}\\
&\leq \frac{1}{2}\left\{\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\Sigma^{-1}_t}+\mathbf{tr}\left(\Sigma^*\Sigma^{-1}_t\right)+\log\frac{|\Sigma_t|}{|\Sigma^*|}\right\} && (10.27)\\
&-\frac{1}{2}\left\{\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\Sigma^{-1}_t}+\mathbf{tr}\left(\Sigma_{t+1}\Sigma^{-1}_t\right)+\log\frac{|\Sigma_t|}{|\Sigma_{t+1}|}-d\right\}\\
&+\beta_t\left\langle \widehat{v}_t, \boldsymbol{m}^*-\boldsymbol{m}^{t+1}\right\rangle
\end{aligned}
$$

Then, we obtain that

$$\begin{aligned}\frac{1}{2}\left\{\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\|^2_{\Sigma_{t+1}^{-1}}+\mathbf{tr}\left(\Sigma^*\Sigma_{t+1}^{-1}\right)\right\}\leq&\frac{1}{2}\left\{\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\Sigma_t^{-1}}+\mathbf{tr}\left(\Sigma^*\Sigma_t^{-1}\right)\right\}\\&-\frac{1}{2}\left\{\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\Sigma_t^{-1}}+\mathbf{tr}\left(\Sigma_{t+1}\Sigma_t^{-1}\right)-d\right\}\\&+\beta_t\left\langle\widehat{v}_t,\boldsymbol{m}^*-\boldsymbol{m}^{t+1}\right\rangle\end{aligned}\tag{10.28}$$

In addition, we have that

$$\begin{aligned}&\mathbf{tr}\left(\Sigma^*\Sigma_t^{-1}\right)-\mathbf{tr}\left(\Sigma^*\Sigma_{t+1}^{-1}\right)-\mathbf{tr}\left(\Sigma_{t+1}\Sigma_t^{-1}\right)+d\\&=\mathbf{tr}\left(\Sigma^*(\Sigma_t^{-1}-\Sigma_{t+1}^{-1})\right)-\mathbf{tr}\left(\Sigma_{t+1}\Sigma_t^{-1}-I\right)&(10.29)\\&=\mathbf{tr}\left(\Sigma^*(\Sigma_t^{-1}-\Sigma_{t+1}^{-1})\right)-\mathbf{tr}\left(\Sigma_{t+1}(\Sigma_t^{-1}-\Sigma_{t+1}^{-1})\right)&(10.30)\\&=\mathbf{tr}\left((\Sigma^*-\Sigma_{t+1})(\Sigma_t^{-1}-\Sigma_{t+1}^{-1})\right)&(10.31)\\&=\mathbf{tr}\left((\boldsymbol{m}_2^*-\boldsymbol{\mu}^*\boldsymbol{\mu}^{*\top}-\boldsymbol{m}_2^{t+1}+\boldsymbol{\mu}_{t+1}\boldsymbol{\mu}_{t+1}^\top)(\Sigma_t^{-1}-\Sigma_{t+1}^{-1})\right)&(10.32)\end{aligned}$$

Note that $\Sigma_t^{-1}-\Sigma_{t+1}^{-1}=-2\beta_t\widehat{G}_t$ by updating rule, it follows that

$$\mathbf{tr}\left(\Sigma^*\Sigma_t^{-1}\right)-\mathbf{tr}\left(\Sigma^*\Sigma_{t+1}^{-1}\right)-\mathbf{tr}\left(\Sigma_{t+1}\Sigma_t^{-1}\right)+d=-2\beta_t\mathbf{tr}\left((\boldsymbol{m}_2^*-\boldsymbol{\mu}^*\boldsymbol{\mu}^{*\top}-\boldsymbol{m}_2^{t+1}+\boldsymbol{\mu}_{t+1}\boldsymbol{\mu}_{t+1}^\top)\widehat{G}_t\right)\tag{10.33}$$

Then, recall that

$$\left\langle\widehat{v}_t,\boldsymbol{m}^*-\boldsymbol{m}^{t+1}\right\rangle=\left\langle\widehat{g}_t-2\widehat{G}_t\boldsymbol{\mu}_t,\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\right\rangle+\mathbf{tr}\left((\boldsymbol{m}_2^*-\boldsymbol{m}_2^{t+1})\widehat{G}_t\right)\tag{10.34}$$

Plug (10.34) and (10.33) into (10.28), we can get that

$$\begin{aligned}\frac{1}{2}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\|^2_{\Sigma_{t+1}^{-1}}\leq&\frac{1}{2}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\Sigma_t^{-1}}-\frac{1}{2}\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\Sigma_t^{-1}}+\beta_t\left\langle\widehat{g}_t,\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\right\rangle\\&-2\beta_t\left\langle\widehat{G}_t\boldsymbol{\mu}_t,\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\right\rangle+\beta_t\mathbf{tr}\left((\boldsymbol{\mu}^*\boldsymbol{\mu}^{*\top}-\boldsymbol{\mu}_{t+1}\boldsymbol{\mu}_{t+1}^\top)\widehat{G}_t\right)\end{aligned}\tag{10.35}$$

Note that

$$\begin{aligned}&-2\left\langle\widehat{G}_t\boldsymbol{\mu}_t,\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\right\rangle+\mathbf{tr}\left((\boldsymbol{\mu}^*\boldsymbol{\mu}^{*\top}-\boldsymbol{\mu}_{t+1}\boldsymbol{\mu}_{t+1}^\top)\widehat{G}_t\right)\\&=\left\langle\widehat{G}_t\boldsymbol{\mu}^*,\boldsymbol{\mu}^*\right\rangle-2\left\langle\widehat{G}_t\boldsymbol{\mu}_t,\boldsymbol{\mu}^*\right\rangle+\left\langle\widehat{G}_t\boldsymbol{\mu}_t,\boldsymbol{\mu}_t\right\rangle&(10.36)\\&-\left\langle\widehat{G}_t\boldsymbol{\mu}_{t+1},\boldsymbol{\mu}_{t+1}\right\rangle+2\left\langle\widehat{G}_t\boldsymbol{\mu}_t,\boldsymbol{\mu}_{t+1}\right\rangle-\left\langle\widehat{G}_t\boldsymbol{\mu}_t,\boldsymbol{\mu}_t\right\rangle\\&=\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\widehat{G}_t}-\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\widehat{G}_t}&(10.37)\end{aligned}$$

Plug into (10.35), we can obtain that

$$\begin{aligned}\frac{1}{2}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\|^2_{\Sigma_{t+1}^{-1}}\leq&\frac{1}{2}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\Sigma_t^{-1}}-\frac{1}{2}\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\Sigma_t^{-1}}+\beta_t\left\langle\widehat{g}_t,\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\right\rangle\\&+\beta_t\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\widehat{G}_t}-\beta_t\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\widehat{G}_t}\end{aligned}\tag{10.38}$$

Also note that $\frac{1}{2}\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\Sigma_t^{-1}}+\beta_t\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\widehat{G}_t}=\frac{1}{2}\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\Sigma_{t+1}^{-1}}$, we obtain that

$$\frac{1}{2}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\|^2_{\Sigma_{t+1}^{-1}}\leq\frac{1}{2}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\Sigma_t^{-1}}-\frac{1}{2}\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\Sigma_{t+1}^{-1}}+\beta_t\left\langle\widehat{g}_t,\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\right\rangle+\beta_t\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\widehat{G}_t} \tag{10.39}$$

□

Lemma 3. *Given a convex function $f(x)$, for Gaussian distribution with parameters $\boldsymbol{\theta}:=\{\boldsymbol{\mu},\Sigma^{\frac{1}{2}}\}$, let $\bar{J}(\boldsymbol{\theta}):=\mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})]$. Then $\bar{J}(\boldsymbol{\theta})$ is a convex function with respect to $\boldsymbol{\theta}$.*

Proof. For $\lambda\in[0,1]$, we have

$$\begin{aligned}
\lambda\bar{J}(\boldsymbol{\theta}_1)+(1-\lambda)\bar{J}(\boldsymbol{\theta}_2)&=\lambda\mathbb{E}[f(\boldsymbol{\mu}_1+\Sigma_1^{\frac{1}{2}}\boldsymbol{z})]+(1-\lambda)\mathbb{E}[f(\boldsymbol{\mu}_2+\Sigma_2^{\frac{1}{2}}\boldsymbol{z})] &(10.40)\\
&=\mathbb{E}[\lambda f(\boldsymbol{\mu}_1+\Sigma_1^{\frac{1}{2}}\boldsymbol{z})+(1-\lambda)f(\boldsymbol{\mu}_2+\Sigma_2^{\frac{1}{2}}\boldsymbol{z})] &(10.41)\\
&\geq\mathbb{E}[f\left(\lambda\boldsymbol{\mu}_1+(1-\lambda)\lambda\boldsymbol{\mu}_2+(\lambda\Sigma_1^{\frac{1}{2}}+(1-\lambda)\Sigma_2^{\frac{1}{2}})\boldsymbol{z}\right)] &(10.42)\\
&=\bar{J}(\lambda\boldsymbol{\theta}_1+(1-\lambda)\boldsymbol{\theta}_2) &(10.43)
\end{aligned}$$

□

Lemma 4. *Let $\bar{J}(\boldsymbol{\theta}):=\mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})]$ for Guassian distribution with parameter $\boldsymbol{\theta}:=\{\boldsymbol{\mu},\Sigma^{\frac{1}{2}}\}\in\boldsymbol{\Theta}$ and $\boldsymbol{\Theta}:=\{\boldsymbol{\mu},\Sigma^{\frac{1}{2}}\big|\ \boldsymbol{\mu}\in\mathcal{R}^d,\Sigma\in\mathcal{S}^+\}$ be a γ-strongly convex function. Suppose $bI\preceq\widehat{G}_t\preceq\frac{\gamma}{2}I$ be positive definite matrix and $\Sigma_1\in\boldsymbol{\Theta}$, then we have*

$$\frac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\|^2_{\Sigma_{t+1}^{-1}}\leq\frac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\Sigma_t^{-1}}+\beta_t\mathbb{E}(\bar{J}(\boldsymbol{\theta}^*)-\bar{J}(\boldsymbol{\theta}_t))+\beta_t\mathbb{E}\left\langle G_t,2\boldsymbol{\Sigma}_t\right\rangle+\frac{\beta_t^2}{2}\mathbb{E}\|\Sigma_{t+1}\|_2\|\widehat{g}_t\|_2^2 \tag{10.44}$$

Proof. From Lemma 2, we know that

$$\frac{1}{2}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\|^2_{\Sigma_{t+1}^{-1}}\leq\frac{1}{2}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\Sigma_t^{-1}}-\frac{1}{2}\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\Sigma_{t+1}^{-1}}+\beta_t\left\langle\widehat{g}_t,\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\right\rangle+\beta_t\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\widehat{G}_t} \tag{10.45}$$

It follows that

$$\frac{1}{2}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_{t+1}\|^2_{\Sigma_{t+1}^{-1}}\leq\frac{1}{2}\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\Sigma_t^{-1}}-\frac{1}{2}\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\Sigma_{t+1}^{-1}}+\beta_t\left\langle\widehat{g}_t,\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\right\rangle+\beta_t\left\langle\widehat{g}_t,\boldsymbol{\mu}_t-\boldsymbol{\mu}_{t+1}\right\rangle+\beta_t\|\boldsymbol{\mu}^*-\boldsymbol{\mu}_t\|^2_{\widehat{G}_t} \tag{10.46}$$

Note that

$$\begin{aligned}
&-\frac{1}{2}\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\Sigma_{t+1}^{-1}}+\beta_t\left\langle\widehat{g}_t,\boldsymbol{\mu}_t-\boldsymbol{\mu}_{t+1}\right\rangle\\
&=-\frac{1}{2}\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t\|^2_{\Sigma_{t+1}^{-1}}+\beta_t\left\langle\Sigma_{t+1}\widehat{g}_t,\Sigma_{t+1}^{-1}(\boldsymbol{\mu}_t-\boldsymbol{\mu}_{t+1})\right\rangle &(10.47)\\
&=-\frac{1}{2}\|\boldsymbol{\mu}_{t+1}-\boldsymbol{\mu}_t+\beta_t\Sigma_{t+1}\widehat{g}_t\|^2_{\Sigma_{t+1}^{-1}}+\frac{\beta_t^2}{2}\|\Sigma_{t+1}\widehat{g}_t\|^2_{\Sigma_{t+1}^{-1}} &(10.48)\\
&\leq\frac{\beta_t^2}{2}\|\Sigma_{t+1}\widehat{g}_t\|^2_{\Sigma_{t+1}^{-1}}\leq\frac{\beta_t^2}{2}\|\Sigma_{t+1}\|_2\|\widehat{g}_t\|_2^2 &(10.49)
\end{aligned}$$

Note that $\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + 2\beta_t \widehat{G}_t$ and $\widehat{G}_t \succeq bI$, we have smallest eigenvalues $\lambda_{min}(\Sigma_{t+1}^{-1}) \geq \lambda_{min}(\Sigma_t^{-1}) \geq \cdots \geq \lambda_{min}(\Sigma_1^{-1})$. Then, we know $\|\Sigma_{t+1}\|_2 \leq \|\Sigma_1\|_2$. In addition, Σ_{t+1} is positive definite matrix, thus $\Sigma_{t+1} \in \boldsymbol{\Theta}$ for $t \in \{1, 2, 3 \cdots\}$.

Plug (10.49) into (10.46), we can achieve that

$$\frac{1}{2}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_{t+1}\|_{\Sigma_{t+1}^{-1}}^2 \leq \frac{1}{2}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_t\|_{\Sigma_t^{-1}}^2 + \beta_t \langle \widehat{g}_t, \boldsymbol{\mu}^* - \boldsymbol{\mu}_t \rangle + \beta_t \|\boldsymbol{\mu}^* - \boldsymbol{\mu}_t\|_{\widehat{G}_t}^2 + \frac{\beta_t^2}{2}\|\Sigma_{t+1}\|_2 \|\widehat{g}_t\|_2^2 \tag{10.50}$$

Since $bI \preceq \widehat{G}_t \preceq \frac{\gamma}{2} I$, we get that

$$\frac{1}{2}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_{t+1}\|_{\Sigma_{t+1}^{-1}}^2 \leq \frac{1}{2}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_t\|_{\Sigma_t^{-1}}^2 + \beta_t \langle \widehat{g}_t, \boldsymbol{\mu}^* - \boldsymbol{\mu}_t \rangle + \beta_t \frac{\gamma}{2}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_t\|_2^2 + \frac{\beta_t^2}{2}\|\Sigma_{t+1}\|_2 \|\widehat{g}_t\|_2^2 \tag{10.51}$$

Taking conditional expectation on both sides, we obtain that

$$\begin{aligned}
&\frac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_{t+1}\|_{\Sigma_{t+1}^{-1}}^2 \\
&\leq \frac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_t\|_{\Sigma_t^{-1}}^2 + \beta_t \langle \mathbb{E}\widehat{g}_t, \boldsymbol{\mu}^* - \boldsymbol{\mu}_t \rangle + \beta_t \frac{\gamma}{2}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_t\|_2^2 + \frac{\beta_t^2}{2}\mathbb{E}\|\Sigma_{t+1}\|_2 \|\widehat{g}_t\|_2^2 \qquad (10.52) \\
&\leq \frac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_t\|_{\Sigma_t^{-1}}^2 + \beta_t \langle \mathbb{E}\widehat{g}_t, \boldsymbol{\mu}^* - \boldsymbol{\mu}_t \rangle - \beta_t \langle G_t, 2\boldsymbol{\Sigma}_t \rangle + \beta_t \langle G_t, 2\boldsymbol{\Sigma}_t \rangle \\
&+ \beta_t \frac{\gamma}{2}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_t\|_2^2 + \frac{\beta_t^2}{2}\mathbb{E}\|\Sigma_{t+1}\|_2 \|\widehat{g}_t\|_2^2 \qquad (10.53)
\end{aligned}$$

Note that $g_t = \mathbb{E}\widehat{g}_t = \nabla_{\boldsymbol{\mu}=\boldsymbol{\mu}_t} \bar{J}$ and $G_t = \nabla_{\Sigma=\Sigma_t} \bar{J}$ and $\nabla_{\Sigma^{\frac{1}{2}}} \bar{J} = \Sigma^{\frac{1}{2}} \nabla_\Sigma \bar{J} + \nabla_\Sigma \bar{J} \Sigma^{\frac{1}{2}}$, where G_t, $\nabla_\Sigma \bar{J}$ and $\Sigma^{\frac{1}{2}}$ are symmetric matrix. Since $\bar{J}(\boldsymbol{\theta})$ is a γ-strongly convex function with optimum at $\boldsymbol{\theta}^* = \{\boldsymbol{\mu}^*, \mathbf{0}\}$, we have that

$$\begin{aligned}
&\langle \nabla_{\boldsymbol{\mu}=\boldsymbol{\mu}_t} \bar{J}, \boldsymbol{\mu}^* - \boldsymbol{\mu}_t \rangle + \left\langle \nabla_{\Sigma^{\frac{1}{2}}=\Sigma_t^{\frac{1}{2}}} \bar{J}, \mathbf{0} - \Sigma_t^{\frac{1}{2}} \right\rangle \\
&= \langle g_t, \boldsymbol{\mu}^* - \boldsymbol{\mu}_t \rangle + \left\langle \Sigma_t^{\frac{1}{2}} G_t + G_t \Sigma_t^{\frac{1}{2}}, \mathbf{0} - \Sigma_t^{\frac{1}{2}} \right\rangle \qquad (10.54) \\
&= \langle g_t, \boldsymbol{\mu}^* - \boldsymbol{\mu}_t \rangle - \langle G_t, 2\Sigma_t \rangle \qquad (10.55) \\
&\leq (\bar{J}(\boldsymbol{\theta}^*) - \bar{J}(\boldsymbol{\theta}_t)) - \frac{\gamma}{2}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_t\|_2^2 \qquad (10.56)
\end{aligned}$$

Plug it into (10.53), we can obtain that

$$\tfrac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_{t+1}\|_{\Sigma_{t+1}^{-1}}^2 \leq \tfrac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_t\|_{\Sigma_t^{-1}}^2 + \beta_t(\bar{J}(\boldsymbol{\theta}^*) - \bar{J}(\boldsymbol{\theta}_t)) + \beta_t \langle G_t, 2\boldsymbol{\Sigma}_t \rangle + \tfrac{\beta_t^2}{2}\mathbb{E}\|\Sigma_{t+1}\|_2 \|\widehat{g}_t\|_2^2$$

Taking expectation on both sides, we know that

$$\tfrac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_{t+1}\|_{\Sigma_{t+1}^{-1}}^2 \leq \tfrac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_t\|_{\Sigma_t^{-1}}^2 + \beta_t \mathbb{E}(\bar{J}(\boldsymbol{\theta}^*) - \bar{J}(\boldsymbol{\theta}_t)) + \beta_t \mathbb{E}\langle G_t, 2\boldsymbol{\Sigma}_t \rangle + \tfrac{\beta_t^2}{2}\mathbb{E}\|\Sigma_{t+1}\|_2 \|\widehat{g}_t\|_2^2 \tag{10.57}$$

□

Lemma 5. *Given a symmetric matrix X and a symmetric positive semi-definite matrix Y, then we have $\boldsymbol{tr}(XY) \leq \|Y\|_2\|X\|_{tr}$, where $\|X\|_{tr} := \sum_{i=1}^{d} |\lambda_i|$ with λ_i denotes the eigenvalues.*

Proof. Since X is symmetric, it can be orthogonal diagonalized as $X = U\Lambda U^\top$, where Λ is a diagonal matrix contains eigenvalues $\lambda_i, i \in \{1, \cdots, d\}$. Since Y is a symmetric positive semi-definite matrix, it can be written as $Y = Y^{\frac{1}{2}}Y^{\frac{1}{2}}$. It follows that

$$\mathbf{tr}\left(XY\right) = \mathbf{tr}\left(U\Lambda U^\top Y^{\frac{1}{2}}Y^{\frac{1}{2}}\right) = \mathbf{tr}\left(Y^{\frac{1}{2}}U\Lambda U^\top Y^{\frac{1}{2}}\right) = \sum_{i=1}^{d} \lambda_i \boldsymbol{a}_i^\top \boldsymbol{a}_i \tag{10.58}$$

where $\boldsymbol{a}_i$ denotes the i^{th} column of the matrix $A = U^\top Y^{\frac{1}{2}}$. Then, we have

$$\mathbf{tr}\left(XY\right) \leq \sum_{i=1}^{d} |\lambda_i| \boldsymbol{a}_i^\top \boldsymbol{a}_i = \mathbf{tr}\left(Y^{\frac{1}{2}}U|\Lambda|U^\top Y^{\frac{1}{2}}\right) = \mathbf{tr}\left(Y^{\frac{1}{2}}\bar{X}^{\frac{1}{2}}\bar{X}^{\frac{1}{2}}Y^{\frac{1}{2}}\right) = \|Y^{\frac{1}{2}}\bar{X}^{\frac{1}{2}}\|_F^2 \tag{10.59}$$

where $\bar{X}^{\frac{1}{2}} = U|\Lambda|^{\frac{1}{2}}U^\top$.

Using the fact $\|Y^{\frac{1}{2}}\bar{X}^{\frac{1}{2}}\|_F^2 \leq \|Y^{\frac{1}{2}}\|_2^2\|\bar{X}^{\frac{1}{2}}\|_F^2$, we can obtain that

$$\mathbf{tr}\left(XY\right) = \|Y^{\frac{1}{2}}\bar{X}^{\frac{1}{2}}\|_F^2 \leq \|Y^{\frac{1}{2}}\|_2^2\|\bar{X}^{\frac{1}{2}}\|_F^2 = \|Y\|_2\|\bar{X}\|_{tr} = \|Y\|_2\|X\|_{tr} \tag{10.60}$$

□

Lemma 6. *Suppose gradients $\|G_t\|_{tr} \leq B_1$ and $\widehat{G}_t \succeq bI$ with $b > 0$, by setting $\beta_t = \beta$ as a constant step size, we have*

$$\sum_{t=1}^{T} \beta_t \mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t \right\rangle \leq 2B_1 \left(\beta\|\Sigma_1\|_2 + \frac{1+\log T}{2b}\right) \tag{10.61}$$

Proof. Note that $\Sigma_{t+1}^{-1} - \Sigma_t^{-1} = 2\beta_t\widehat{G}_t$ and $\widehat{G}_t \succeq bI$ with $b > 0$, we know the smallest eigenvalue of Σ_{t+1}^{-1}, i.e. $\lambda_{min}(\Sigma_{t+1}^{-1})$ satisfies that

$$\lambda_{min}(\Sigma_{t+1}^{-1}) \geq \lambda_{min}(\Sigma_t^{-1}) + 2\beta_t b \geq \lambda_{min}(\Sigma_1^{-1}) + 2\sum_{i=1}^{t} \beta_i b \geq 2\sum_{i=1}^{t} \beta_i b \tag{10.62}$$

Thus, we know that

$$\|\Sigma_{t+1}\|_2 = \frac{1}{\lambda_{min}(\Sigma_{t+1}^{-1})} \leq \frac{1}{2\sum_{i=1}^{t} \beta_i b} = \frac{1}{2t\beta b} \tag{10.63}$$

Note that Σ_t is symmetric positive semi-definite and G_t is symmetric. From Lemma 5, we know that $\mathbf{tr}(G_t\Sigma_t) \leq \|\Sigma_t\|_2\|G_t\|_{tr}$. It follows that

$$\sum_{t=1}^{T} \beta_t \mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t \right\rangle \leq 2\beta \sum_{t=1}^{T} \mathbb{E}[\|G_t\|_{tr}\|\Sigma_t\|_2] \leq 2\beta B_1 \sum_{t=1}^{T} \mathbb{E}\|\Sigma_t\|_2 \tag{10.64}$$

$$\leq 2\beta B_1\|\Sigma_1\|_2 + 2B_1(\sum_{t=1}^{T-1}\frac{1}{2bt}) \tag{10.65}$$

Since $\sum_{t=1}^{T}\frac{1}{t} \leq 1+\log T$, we know that

$$\sum_{t=1}^{T} \beta_t \mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t \right\rangle \leq 2\beta B_1\|\Sigma_1\|_2 + 2B_1\left(\frac{1+\log T}{2b}\right) = 2B_1\left(\beta\|\Sigma_1\|_2 + \frac{1+\log T}{2b}\right) \tag{10.66}$$

□

Theorem. *Given a convex function $f(\boldsymbol{x})$, define $\bar{J}(\boldsymbol{\theta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})]$ for Guassian distribution with parameter $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}}\} \in \boldsymbol{\Theta}$ and $\boldsymbol{\Theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}} \big| \ \boldsymbol{\mu} \in \mathcal{R}^d, \Sigma \in \mathcal{S}^+\}$. Suppose $\bar{J}(\boldsymbol{\theta})$ be γ-strongly convex. Let $\widehat{G}_t$ be positive semi-definite matrix such that $bI \preceq \widehat{G}_t \preceq \frac{\gamma}{2}I$. Suppose $\Sigma_1 \in \mathcal{S}^{++}$ and $\|\Sigma_1\| \leq \rho$, $\mathbb{E}\widehat{g}_t = \nabla_{\boldsymbol{\mu}=\boldsymbol{\mu}_t}\bar{J}$. Assume furthermore $\|\nabla_{\Sigma=\Sigma_t}\bar{J}\|_{tr} \leq B_1$ and $\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_1\|^2_{\Sigma_1^{-1}} \leq R$, $\mathbb{E}\|\widehat{g}_t\|_2^2 \leq \mathcal{B}$. Set $\beta_t = \beta$, then Algorithm 4 can achieve*

$$\frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}f(\boldsymbol{\mu}_t)\right] - f(\boldsymbol{\mu}^*) \leq \frac{2bR+2b\beta\rho(4B_1+\beta\mathcal{B})+4B_1(1+\log T)+(1+\log T)\beta\mathcal{B}}{4\beta bT} = \mathcal{O}\left(\frac{\log T}{T}\right) \tag{10.67}$$

Proof. From Lemma 1 to Lemma 4, we know that

$$\frac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_{t+1}\|^2_{\Sigma_{t+1}^{-1}} \leq \frac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_t\|^2_{\Sigma_t^{-1}} + \beta_t\mathbb{E}(\bar{J}(\boldsymbol{\theta}^*) - \bar{J}(\boldsymbol{\theta}_t)) + \beta_t\mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t\right\rangle + \frac{\beta_t^2}{2}\mathbb{E}\|\Sigma_{t+1}\|_2\|\widehat{g}_t\|_2^2 \tag{10.68}$$

Sum up both sides from $t=1$ to $t=T$ and rearrange terms, we get

$$\sum_{t=1}^{T}\beta_t\mathbb{E}\left[\bar{J}(\boldsymbol{\theta}_t) - \bar{J}(\boldsymbol{\theta}^*)\right] \leq \frac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_1\|^2_{\Sigma_1^{-1}} - \frac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_{T+1}\|^2_{\Sigma_{T+1}^{-1}} \tag{10.69}$$

$$+ \sum_{t=1}^{T}\beta_t\mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t\right\rangle + \sum_{t=1}^{T}\frac{\beta_t^2}{2}\mathbb{E}\|\Sigma_{t+1}\|_2\|\widehat{g}_t\|_2^2 \tag{10.70}$$

Since $\beta_t = \beta$, we can obtain that

$$\frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}\bar{J}(\boldsymbol{\theta}_t)\right] - \bar{J}(\boldsymbol{\theta}^*)$$

$$\leq \frac{\frac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_1\|^2_{\Sigma_1^{-1}} + \sum_{t=1}^{T}\beta_t\mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t\right\rangle + \frac{\beta^2}{2}\sum_{t=1}^{T}\mathbb{E}\|\Sigma_{t+1}\|_2\|\widehat{g}_t\|_2^2}{T\beta} \tag{10.71}$$

$$\leq \frac{\frac{1}{2}R + \sum_{t=1}^{T}\beta_t\mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t\right\rangle + \frac{\beta^2}{2}\mathcal{B}\sum_{t=1}^{T}\mathbb{E}\|\Sigma_{t+1}\|_2}{T\beta} \tag{10.72}$$

From Eq.(10.63), we know that

$$\|\Sigma_{t+1}\|_2 \leq \frac{1}{2tb\beta} \tag{10.73}$$

Since $\sum_{t=1}^{T}\frac{1}{t} \leq 1+\log T$, we know that $\sum_{t=1}^{T}\mathbb{E}\|\Sigma_{t+1}\|_2 \leq \|\Sigma_1\|_2 + \frac{1+\log T}{2\beta b} \leq \rho + \frac{1+\log T}{2\beta b}$

In addition, from Lemma 6, we know that

$$\sum_{t=1}^{T}\beta_t\mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t\right\rangle \leq 2B_1\left(\beta\|\Sigma_1\|_2 + \frac{1+\log T}{2b}\right) \leq 2B_1\left(\beta\rho + \frac{1+\log T}{2b}\right) \tag{10.74}$$

Plug all them into (10.72), we can get

$$\frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}\bar{J}(\boldsymbol{\theta}_t)\right] - \bar{J}(\boldsymbol{\theta}^*)$$

$$\leq \frac{\frac{1}{2}R + 2B_1\left(\beta\rho + \frac{1+\log T}{2b}\right) + \frac{\beta^2\rho\mathcal{B}}{2} + \frac{(1+\log T)\beta\mathcal{B}}{4b}}{T\beta} \tag{10.75}$$

$$= \frac{2bR + 8B_1b\beta\rho + 4B_1(1+\log T) + 2b\beta^2\rho\mathcal{B} + (1+\log T)\beta\mathcal{B}}{4\beta bT} \tag{10.76}$$

$$= \frac{2bR + 2b\beta\rho(4B_1 + \beta\mathcal{B}) + 4B_1(1+\log T) + (1+\log T)\beta\mathcal{B}}{4\beta bT} \tag{10.77}$$

$$= \mathcal{O}\left(\frac{\log T}{T}\right) \tag{10.78}$$

Since $f(\boldsymbol{x})$ is a convex function, we know $f(\boldsymbol{\mu}) \leq \bar{J}(\boldsymbol{\mu}, \Sigma) = \mathbb{E}[f(\boldsymbol{x})]$. Note that for an optimum point $\boldsymbol{\mu}^*$ of $f(\boldsymbol{x})$, $\boldsymbol{\theta}^* = (\boldsymbol{\mu}^*, \mathbf{0})$ is an optimum of $\bar{J}(\boldsymbol{\theta})$, i.e., $f(\boldsymbol{\mu}^*) = \bar{J}(\boldsymbol{\theta}^*)$. Thus, we can obtain that

$$\frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}f(\boldsymbol{\mu}_t)\right] - f(\boldsymbol{\mu}^*) \leq \frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}\bar{J}(\boldsymbol{\theta}_t)\right] - \bar{J}(\boldsymbol{\theta}^*) \tag{10.79}$$

$$\leq \frac{2bR + 2b\beta\rho(4B_1 + \beta\mathcal{B}) + 4B_1(1+\log T) + (1+\log T)\beta\mathcal{B}}{4\beta bT} \tag{10.80}$$

$$\leq \mathcal{O}\left(\frac{\log T}{T}\right) \tag{10.81}$$

□

10.4 Proof of Theorem 5

Lemma 7. *For a L-Lipschitz continuous black box function $f(\boldsymbol{x})$. Let $\widehat{G}_t$ be positive semi-definite matrix such that $bI \preceq \widehat{G}_t$ with $b > 0$. Suppose the gradient estimator $\widehat{g}_t$ is defined as*

$$\widehat{g}_t = \Sigma_t^{-\frac{1}{2}} \boldsymbol{z} \left(f(\boldsymbol{\mu}_t + \Sigma_t^{\frac{1}{2}} \boldsymbol{z}) - f(\boldsymbol{\mu}_t) \right) \tag{10.82}$$

where $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, I)$. Then $\widehat{g}_t$ is an unbiased estimator of $\nabla_{\boldsymbol{\mu}} \mathbb{E}_p[f(\boldsymbol{x})]$ and $\mathbb{E}\|\Sigma_{t+1}\|_2 \|\widehat{g}_t\|_2^2 \leq L^2 \|\Sigma_t\|_2 (d+4)^2$

Proof. We first show the unbiased estimator.

$$\begin{aligned}
\mathbb{E}[\widehat{g}_t] &= \mathbb{E}\left[\Sigma_t^{-\frac{1}{2}} \boldsymbol{z} f(\boldsymbol{\mu}_t + \Sigma_t^{\frac{1}{2}} \boldsymbol{z})\right] - \mathbb{E}\left[\Sigma_t^{-\frac{1}{2}} \boldsymbol{z} f(\boldsymbol{\mu}_t)\right] && (10.83)\\
&= \mathbb{E}\left[\Sigma_t^{-\frac{1}{2}} \boldsymbol{z} f(\boldsymbol{\mu}_t + \Sigma_t^{\frac{1}{2}} \boldsymbol{z})\right] && (10.84)\\
&= \mathbb{E}_{p(\boldsymbol{\mu}_t, \Sigma_t)}\left[\Sigma_t^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_t) f(\boldsymbol{x})\right] && (10.85)\\
&= \nabla_{\boldsymbol{\mu}} \mathbb{E}_p[f(\boldsymbol{x})] && (10.86)
\end{aligned}$$

The last equality holds by Theorem 2.

Now, we prove the bound of $\mathbb{E}_p \|\Sigma_{t+1}\|_2 \|\widehat{g}_t\|_2^2$.

$$\begin{aligned}
\|\Sigma_{t+1}\|_2 \, \|\widehat{g}_t\|_2^2 &= \|\Sigma_{t+1}\|_2 \, \|\Sigma_t^{-\frac{1}{2}} \boldsymbol{z}\|_2^2 \left(f(\boldsymbol{\mu}_t + \Sigma_t^{\frac{1}{2}} \boldsymbol{z}) - f(\boldsymbol{\mu}_t) \right)^2 && (10.87)\\
&\leq \|\Sigma_{t+1}\|_2 \, \|\Sigma_t^{-\frac{1}{2}} \boldsymbol{z}\|_2^2 \, L^2 \|\Sigma_t^{\frac{1}{2}} \boldsymbol{z}\|_2^2 && (10.88)\\
&\leq \|\Sigma_{t+1}\|_2 \, \|\Sigma_t^{-\frac{1}{2}}\|_2^2 \, \|\boldsymbol{z}\|_2^2 \, L^2 \|\Sigma_t^{\frac{1}{2}} \boldsymbol{z}\|_2^2 && (10.89)\\
&= \|\Sigma_{t+1}\|_2 \, \|\Sigma_t^{-1}\|_2 \, \|\boldsymbol{z}\|_2^2 \;\; L^2 \|\Sigma_t^{\frac{1}{2}} \boldsymbol{z}\|_2^2 && (10.90)
\end{aligned}$$

Since $\|\Sigma_{t+1}\|_2 \leq \|\Sigma_t\|_2$ proved in Lemma 4 (below Eq.(10.49)), we get that

$$\|\Sigma_{t+1}\|_2 \, \|\widehat{g}_t\|_2^2 \leq \|\boldsymbol{z}\|_2^2 \;\; L^2 \|\Sigma_t^{\frac{1}{2}} \boldsymbol{z}\|_2^2 \leq L^2 \|\Sigma_t\|_2 \, \|\boldsymbol{z}\|_2^4 \tag{10.91}$$

Since $\mathbb{E}\|\boldsymbol{z}\|_2^4 \leq (d+4)^2$ shown in [129], we can obtain that

$$\mathbb{E}\|\Sigma_{t+1}\|_2 \|\widehat{g}_t\|_2^2 \leq L^2 \|\Sigma_t\|_2 (d+4)^2 \tag{10.92}$$

□

Theorem. *For a L-Lipschitz continuous convex black box function $f(\boldsymbol{x})$, define $\bar{J}(\boldsymbol{\theta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})]$ for Guassian distribution with parameter $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}}\} \in \boldsymbol{\Theta}$ and $\boldsymbol{\Theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}} \big| \;\; \boldsymbol{\mu} \in \mathcal{R}^d, \Sigma \in \mathcal{S}^+\}$. Suppose $\bar{J}(\boldsymbol{\theta})$ be γ-strongly convex. Let $\widehat{G}_t$ be positive*

semi-definite matrix such that $b\boldsymbol{I} \preceq \widehat{G}_t \preceq \frac{\gamma}{2}\boldsymbol{I}$. *Suppose* $\Sigma_1 \in \mathcal{S}^{++}$ *and* $\|\Sigma_1\|_2 \leq \rho$. *Assume furthermore* $\|\nabla_{\Sigma=\Sigma_t}\bar{J}\|_{tr} \leq B_1$ *and* $\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_1\|^2_{\Sigma_1^{-1}} \leq R$, . *Set* $\beta_t = \beta$ *and employ estimator* $\widehat{g}_t$ *in Eq.(3.50), then Algorithm 4 can achieve*

$$\frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}f(\boldsymbol{\mu}_t)\right] - f(\boldsymbol{\mu}^*)$$
$$\leq \frac{2bR + 2b\beta\rho(4B_1 + 2\beta L^2(d+4)^2) + 4B_1(1+\log T) + (1+\log T)\beta L^2(d+4)^2}{4\beta bT} \tag{10.93}$$
$$= \mathcal{O}\left(\frac{d^2\log T}{T}\right) \tag{10.94}$$

Proof. We are now ready to prove Theorem 5.

From Lemma 7, we know $\mathbb{E}\|\Sigma_{t+1}\|_2\|\widehat{g}_t\|_2^2 \leq L^2\|\Sigma_t\|_2(d+4)^2$. Note that $\|\Sigma_{t+1}\|_2 \leq \frac{1}{2t\beta b}$ from Eq.(10.63), we can obtain that

$$\mathbb{E}\|\Sigma_{t+1}\|_2\|\widehat{g}_t\|_2^2 \leq L^2\|\Sigma_t\|_2(d+4)^2 \leq \frac{L^2(d+4)^2}{2(t-1)\beta b} \tag{10.95}$$

Plug it into Eq.(10.71), also note that $\|\Sigma_2\|_2 \leq \|\Sigma_1\|_2$, we get that

$$\frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}\bar{J}(\boldsymbol{\theta}_t)\right] - \bar{J}(\boldsymbol{\theta}^*)$$
$$\leq \frac{\frac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_1\|^2_{\Sigma_1^{-1}} + \sum_{t=1}^{T}\beta_t\mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t\right\rangle + \beta^2\|\Sigma_1\|_2L^2(d+4)^2 + \frac{\beta L^2(d+4)^2}{4b}\sum_{t=1}^{T}\frac{1}{t}}{T\beta} \tag{10.96}$$
$$\leq \frac{\frac{1}{2}R + \sum_{t=1}^{T}\beta_t\mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t\right\rangle + \beta^2\|\Sigma_1\|_2L^2(d+4)^2 + \frac{\beta L^2(d+4)^2}{4b}(1+\log T)}{T\beta} \tag{10.97}$$

In addition, from Lemma 6, we know that

$$\sum_{t=1}^{T}\beta_t\mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t\right\rangle \leq 2B_1\left(\beta\|\Sigma_1\|_2 + \frac{1+\log T}{2b}\right) \leq 2B_1\left(\beta\rho + \frac{1+\log T}{2b}\right) \tag{10.98}$$

Then, we can get that

$$\frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}\bar{J}(\boldsymbol{\theta}_t)\right]-\bar{J}(\boldsymbol{\theta}^*)$$

$$\leq \frac{\frac{1}{2}R+2B_1\left(\beta\rho+\frac{1+\log T}{2b}\right)+\beta^2\rho L^2(d+4)^2+\frac{\beta L^2(d+4)^2}{4b}(1+\log T)}{T\beta} \tag{10.99}$$

$$=\frac{2bR+2b\beta\rho(4B_1+2\beta L^2(d+4)^2)+4B_1(1+\log T)+(1+\log T)\beta L^2(d+4)^2}{4\beta bT} \tag{10.100}$$

$$=\mathcal{O}\left(\frac{d^2\log T}{T}\right) \tag{10.101}$$

Since $f(\boldsymbol{x})$ is a convex function, we know that

$$\frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}f(\boldsymbol{\mu}_t)\right]-f(\boldsymbol{\mu}^*)$$

$$\leq \frac{1}{T}\left[\sum_{t=1}^{T}\bar{J}(\boldsymbol{\theta}_t)\right]-\bar{J}(\boldsymbol{\theta}^*) \tag{10.102}$$

$$\leq \frac{2bR+2b\beta\rho(4B_1+2\beta L^2(d+4)^2)+4B_1(1+\log T)+(1+\log T)\beta L^2(d+4)^2}{4\beta bT} \tag{10.103}$$

$$=\mathcal{O}\left(\frac{d^2\log T}{T}\right) \tag{10.104}$$

□

10.5 Variance Reduction

Lemma 8. *For a L-Lipschitz continuous black box function $f(\boldsymbol{x})$. Suppose $\Sigma_t=\sigma_t^2\boldsymbol{I}$ with $\sigma_t>0$ for $t\in\{1,\cdots,T\}$. Suppose the gradient estimator $\widehat{g}_t$ is defined as*

$$\widehat{g}_t=\frac{1}{N}\sum_{i=1}^{N}\Sigma_t^{-\frac{1}{2}}\boldsymbol{z}_i\left(f(\boldsymbol{\mu}_t+\Sigma_t^{\frac{1}{2}}\boldsymbol{z}_i)-f(\boldsymbol{\mu}_t)\right) \tag{10.105}$$

where $\boldsymbol{Z}=[\boldsymbol{z}_1,\cdots,\boldsymbol{z}_N]$ has marginal distribution $\mathcal{N}(\boldsymbol{0},\boldsymbol{I})$ and $\boldsymbol{Z}^\top\boldsymbol{Z}=\boldsymbol{I}$. Then $\widehat{g}_t$ is an unbiased estimator of $\nabla_{\boldsymbol{\mu}}\mathbb{E}_p[f(\boldsymbol{x})]$ and $\mathbb{E}_{\boldsymbol{Z}}\|\Sigma_{t+1}\|_2\|\widehat{g}_t\|_2^2\leq\frac{\sigma_{t+1}^2L^2(d+4)^2}{N}$ for $N\leq d$.

Proof. We first show the unbiased estimator.

$$\mathbb{E}_{\boldsymbol{Z}}[\widehat{g}_t] = \mathbb{E}_{\boldsymbol{Z}}\left[\frac{1}{N}\sum_{i=1}^{N}\Sigma_t^{-\frac{1}{2}}\boldsymbol{z}_i\left(f(\boldsymbol{\mu}_t+\Sigma_t^{\frac{1}{2}}\boldsymbol{z}_i)-f(\boldsymbol{\mu}_t)\right)\right] \tag{10.106}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{\boldsymbol{Z}}\left[\Sigma_t^{-\frac{1}{2}}\boldsymbol{z}_i\left(f(\boldsymbol{\mu}_t+\Sigma_t^{\frac{1}{2}}\boldsymbol{z}_i)-f(\boldsymbol{\mu}_t)\right)\right] \tag{10.107}$$

$$= \mathbb{E}_{\boldsymbol{z}}\left[\Sigma_t^{-\frac{1}{2}}\boldsymbol{z}\left(f(\boldsymbol{\mu}_t+\Sigma_t^{\frac{1}{2}}\boldsymbol{z})-f(\boldsymbol{\mu}_t)\right)\right] \tag{10.108}$$

$$= \mathbb{E}_{\boldsymbol{z}}\left[\Sigma_t^{-\frac{1}{2}}\boldsymbol{z}f(\boldsymbol{\mu}_t+\Sigma_t^{\frac{1}{2}}\boldsymbol{z})\right]-\mathbb{E}_{\boldsymbol{z}}\left[\Sigma_t^{-\frac{1}{2}}\boldsymbol{z}f(\boldsymbol{\mu}_t)\right] \tag{10.109}$$

$$= \mathbb{E}\left[\Sigma_t^{-\frac{1}{2}}\boldsymbol{z}f(\boldsymbol{\mu}_t+\Sigma_t^{\frac{1}{2}}\boldsymbol{z})\right] \tag{10.110}$$

$$= \mathbb{E}_{p(\boldsymbol{\mu}_t,\Sigma_t)}\left[\Sigma_t^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_t)f(\boldsymbol{x})\right] \tag{10.111}$$

$$= \nabla_{\boldsymbol{\mu}}\mathbb{E}_p[f(\boldsymbol{x})] \tag{10.112}$$

The last equality holds by Theorem 2.

Now, we prove the bound of $\mathbb{E}_p\|\Sigma_{t+1}\|_2\|\widehat{g}_t\|_2^2$.

$$\|\Sigma_{t+1}\|_2\ \|\widehat{g}_t\|_2^2 \tag{10.113}$$

$$= \sigma_{t+1}^2\left\|\frac{1}{N}\sum_{i=1}^{N}\sigma_t^{-1}\boldsymbol{z}_i\left(f(\boldsymbol{\mu}_t+\sigma_t\boldsymbol{z}_i)-f(\boldsymbol{\mu}_t)\right)\right\|_2^2 \tag{10.114}$$

$$= \frac{\sigma_{t+1}^2}{N^2}\sum_{i=1}^{N}\left\|\sigma_t^{-1}\boldsymbol{z}_i\left(f(\boldsymbol{\mu}_t+\sigma_t\boldsymbol{z}_i)-f(\boldsymbol{\mu}_t)\right)\right\|_2^2$$

$$+\frac{\sigma_{t+1}^2\sigma_t^{-2}}{N^2}\sum_{i=1}^{N}\sum_{i\neq j}^{N}\boldsymbol{z}_i^\top\boldsymbol{z}_j(f(\boldsymbol{\mu}_t+\sigma_t\boldsymbol{z}_i)-f(\boldsymbol{\mu}_t))(f(\boldsymbol{\mu}_t+\sigma_t\boldsymbol{z}_j)-f(\boldsymbol{\mu}_t)) \tag{10.115}$$

$$= \frac{\sigma_{t+1}^2}{N^2}\sum_{i=1}^{N}\left\|\sigma_t^{-1}\boldsymbol{z}_i\left(f(\boldsymbol{\mu}_t+\sigma_t\boldsymbol{z}_i)-f(\boldsymbol{\mu}_t)\right)\right\|_2^2 \tag{10.116}$$

$$\leq \frac{\sigma_{t+1}^2\sigma_t^{-2}\sigma_t^2L^2}{N^2}\sum_{i=1}^{N}\|\boldsymbol{z}_i\|_2^4 = \frac{\sigma_{t+1}^2L^2}{N^2}\sum_{i=1}^{N}\|\boldsymbol{z}_i\|_2^4 \tag{10.117}$$

Thus,we know that

$$\mathbb{E}_{\boldsymbol{Z}}\left[\|\Sigma_{t+1}\|_2\|\widehat{g}_t\|_2^2\right] \leq \frac{\sigma_{t+1}^2L^2}{N^2}\mathbb{E}_{\boldsymbol{Z}}\sum_{i=1}^{N}\|\boldsymbol{z}_i\|_2^4 = \frac{\sigma_{t+1}^2L^2}{N}\mathbb{E}_{\boldsymbol{z}}[\|\boldsymbol{z}\|_2^4] \tag{10.118}$$

Since $\mathbb{E}_{\boldsymbol{z}}\|\boldsymbol{z}\|_2^4 \leq (d+4)^2$ shown in [129], we can obtain that

$$\mathbb{E}_{\boldsymbol{Z}}\|\Sigma_{t+1}\|_2\|\widehat{g}_t\|_2^2 \leq \frac{\sigma_{t+1}^2L^2(d+4)^2}{N} \tag{10.119}$$

□

Theorem. *For a L-Lipschitz continuous convex black box function $f(\boldsymbol{x})$, define $\bar{J}(\boldsymbol{\theta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})]$ for Guassian distribution with parameter $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \sigma_t \boldsymbol{I} \in \boldsymbol{\Theta}$ and $\boldsymbol{\Theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}} \big| \ \boldsymbol{\mu} \in \mathcal{R}^d, \Sigma \in \mathcal{S}^+\}$. Suppose $\bar{J}(\boldsymbol{\theta})$ be γ-strongly convex. Let $\widehat{G}_t = b\boldsymbol{I}$ with $b \leq \frac{\gamma}{2}$. Suppose $\|\Sigma_1\|_2 \leq \rho = \frac{1}{d}$. Assume furthermore $\|\nabla_{\Sigma=\Sigma_t} \bar{J}\|_{tr} \leq B_1$ and $\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_1\|^2_{\Sigma_1^{-1}} \leq R$, . Set $\beta_t = \beta$ and employ orthogonal estimator $\widehat{g}_t$ in Eq.(10.105) with $N = d$, then Algorithm 4 can achieve*

$$\frac{1}{T}\left[\sum_{t=1}^{T} \mathbb{E} f(\boldsymbol{\mu}_t)\right] - f(\boldsymbol{\mu}^*)$$

$$\leq \frac{2bR + 2b\beta(4B_1/d + 2\beta L^2(d+4)^2/d) + 4B_1(1+\log T) + (1+\log T)\beta L^2(d+4)^2/d}{4\beta bT} \tag{10.120}$$

$$= \mathcal{O}\left(\frac{d\log T}{T}\right) \tag{10.121}$$

Proof. The proof is similar to the proof of Theorem 5.

From Lemma 8 and $N = d$, we know $\mathbb{E}\|\Sigma_{t+1}\|_2 \|\widehat{g}_t\|_2^2 \leq \frac{\sigma_{t+1}^2 L^2 (d+4)^2}{d}$. Note that $\sigma_{t+1}^2 = \|\Sigma_{t+1}\|_2 \leq \frac{1}{2t\beta b}$ from Eq.(10.63), we can obtain that

$$\mathbb{E}\|\Sigma_{t+1}\|_2 \|\widehat{g}_t\|_2^2 \leq \frac{\sigma_{t+1}^2 L^2 (d+4)^2}{d} \leq \frac{L^2(d+4)^2}{2t\beta bd} \tag{10.122}$$

Plug it into Eq.(10.71), we get that

$$\frac{1}{T}\left[\sum_{t=1}^{T} \mathbb{E}\bar{J}(\boldsymbol{\theta}_t)\right] - \bar{J}(\boldsymbol{\theta}^*)$$

$$\leq \frac{\frac{1}{2}\mathbb{E}\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_1\|^2_{\Sigma_1^{-1}} + \sum_{t=1}^{T} \beta_t \mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t \right\rangle + \beta^2 \|\Sigma_1\|_2 L^2 (d+4)^2 + \frac{\beta L^2 (d+4)^2}{4bd} \sum_{t=1}^{T} \frac{1}{t}}{T\beta} \tag{10.123}$$

$$\leq \frac{\frac{1}{2}R + \sum_{t=1}^{T} \beta_t \mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t \right\rangle + \beta^2 \|\Sigma_1\|_2 L^2 (d+4)^2 + \frac{\beta L^2 (d+4)^2}{4bd}(1+\log T)}{T\beta} \tag{10.124}$$

In addition, from Lemma 6, we know that

$$\sum_{t=1}^{T} \beta_t \mathbb{E}\left\langle G_t, 2\boldsymbol{\Sigma}_t \right\rangle \leq 2B_1 \left(\beta\|\Sigma_1\|_2 + \frac{1+\log T}{2b}\right) \leq 2B_1 \left(\beta\rho + \frac{1+\log T}{2b}\right) \tag{10.125}$$

Then, we can get that

$$\frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}\bar{J}(\boldsymbol{\theta}_t)\right]-\bar{J}(\boldsymbol{\theta}^*)$$

$$\leq \frac{\frac{1}{2}R+2B_1\left(\beta\rho+\frac{1+\log T}{2b}\right)+\beta^2\rho L^2(d+4)^2+\frac{\beta L^2(d+4)^2}{4bd}(1+\log T)}{T\beta} \tag{10.126}$$

$$= \frac{2bR+2b\beta\rho(4B_1+2\beta L^2(d+4)^2)+4B_1(1+\log T)+(1+\log T)\beta L^2(d+4)^2/d}{4\beta bT} \tag{10.127}$$

$$= \frac{2bR+2b\beta(4B_1/d+2\beta L^2(d+4)^2/d)+4B_1(1+\log T)+(1+\log T)\beta L^2(d+4)^2/d}{4\beta bT} \tag{10.128}$$

$$= \mathcal{O}\left(\frac{d\log T}{T}\right) \tag{10.129}$$

Since $f(\boldsymbol{x})$ is a convex function, we know that

$$\frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}f(\boldsymbol{\mu}_t)\right]-f(\boldsymbol{\mu}^*)\leq\frac{1}{T}\left[\sum_{t=1}^{T}\mathbb{E}\bar{J}(\boldsymbol{\theta}_t)\right]-\bar{J}(\boldsymbol{\theta}^*)\leq\mathcal{O}\left(\frac{d\log T}{T}\right) \tag{10.130}$$

□

10.6 Proof of Updating Theorem

Theorem. *For Gaussian distribution with parameter* $\boldsymbol{m}:=\{\boldsymbol{m}_1,\boldsymbol{m}_2\}=\{\boldsymbol{\mu},\Sigma+\boldsymbol{\mu}\boldsymbol{\mu}^\top\}$*, let* $\widehat{v}_t=\{\widehat{g}_t-2\widehat{G}_t\boldsymbol{\mu}_t,\widehat{G}_t\}$*, then the optimum of problem (10.131) leads to the closed-form update (10.132) and (10.133):*

$$\boldsymbol{m}^{t+1}=\arg\min_{\boldsymbol{m}\in\boldsymbol{\mathcal{M}}}\beta_t\left\langle\boldsymbol{m},\widehat{v}_t\right\rangle+KL\left(p_{\boldsymbol{m}}\|p_{\boldsymbol{m}^t}\right) \tag{10.131}$$

$$\Sigma_{t+1}^{-1}=\Sigma_t^{-1}+2\beta_t\widehat{G}_t \tag{10.132}$$

$$\boldsymbol{\mu}_{t+1}=\boldsymbol{\mu}_t-\beta_t\Sigma_{t+1}\widehat{g}_t \tag{10.133}$$

Proof. For Guassian distribution with mean parameter $\boldsymbol{m}:=\{\boldsymbol{m}_1,\boldsymbol{m}_2\}=\{\boldsymbol{\mu},\Sigma+\boldsymbol{\mu}\boldsymbol{\mu}^\top\}$, also note that $\widehat{v}_t:=\{\widehat{g}_t-2\widehat{G}_t\boldsymbol{\mu}_t,\widehat{G}_t\}$, the problem (3.45) can be rewrited as

$$\beta_t\left\langle\boldsymbol{m},\widehat{v}_t\right\rangle+\mathrm{KL}\left(p_{\boldsymbol{m}}\|p_{\boldsymbol{m}^t}\right)=\beta_t\left\langle\boldsymbol{m}_1,\widehat{g}_t-2\widehat{G}_t\boldsymbol{\mu}_t\right\rangle+\beta_t\left\langle\boldsymbol{m}_2,\widehat{G}_t\right\rangle+\mathrm{KL}\left(p_{\boldsymbol{m}}\|p_{\boldsymbol{m}^t}\right) \tag{10.134}$$

Taking derivative and set to zero, also note that $\nabla_{\boldsymbol{m}}\mathrm{KL}\left(p_{\boldsymbol{m}}\|p_{\boldsymbol{m}^t}\right)=\boldsymbol{\eta}-\boldsymbol{\eta}^t$, $\boldsymbol{\eta}_1:=\Sigma^{-1}\boldsymbol{\mu}$ and $\boldsymbol{\eta}_2:=-\frac{1}{2}\Sigma^{-1}$, we can obtain that

$$-\frac{1}{2}\Sigma_{t+1}^{-1} = -\frac{1}{2}\Sigma_t^{-1} - \beta_t \widehat{G}_t \tag{10.135}$$

$$\Sigma_{t+1}^{-1}\boldsymbol{\mu}_{t+1} = \Sigma_t^{-1}\boldsymbol{\mu}_t - \beta_t\left(\widehat{g}_t - 2\widehat{G}_t\boldsymbol{\mu}_t\right) \tag{10.136}$$

Rearrange terms, we can obtain that

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + 2\beta_t\widehat{G}_t \tag{10.137}$$

$$\boldsymbol{\mu}_{t+1} = \Sigma_{t+1}\Sigma_t^{-1}\boldsymbol{\mu}_t - \beta_t\Sigma_{t+1}\left(\widehat{g}_t - 2\widehat{G}_t\boldsymbol{\mu}_t\right) \tag{10.138}$$

Merge terms in Eq.(10.138), we get that

$$\boldsymbol{\mu}_{t+1} = \Sigma_{t+1}\Sigma_t^{-1}\boldsymbol{\mu}_t - \beta_t\Sigma_{t+1}\left(\widehat{g}_t - 2\widehat{G}_t\boldsymbol{\mu}_t\right) \tag{10.139}$$

$$= \Sigma_{t+1}\left(\Sigma_t^{-1} + 2\beta_t\widehat{G}_t\right)\boldsymbol{\mu}_t - \beta_t\Sigma_{t+1}\widehat{g}_t \tag{10.140}$$

$$= \Sigma_{t+1}\Sigma_{t+1}^{-1}\boldsymbol{\mu}_t - \beta_t\Sigma_{t+1}\widehat{g}_t \tag{10.141}$$

$$= \boldsymbol{\mu}_t - \beta_t\Sigma_{t+1}\widehat{g}_t \tag{10.142}$$

□

10.7 Proof of Gradient and Hessian Theorem

Theorem. *Suppose $f(\boldsymbol{x})$ be an integrable and twice differentiable function under a Gaussian distribution $p := \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ such that $\mathbb{E}_p\left[\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\right]$ and $\mathbb{E}_p\left[\frac{\partial^2 f(\boldsymbol{x})}{\partial \boldsymbol{x}\partial \boldsymbol{x}^\top}\right]$ exists. Then, the expectation of the gradient and Hessian of $f(\boldsymbol{x})$ can be expressed as Eq.(10.143) and Eq.(10.144), respectively.*

$$\mathbb{E}_p\left[\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\right] = \mathbb{E}_p\left[\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})f(\boldsymbol{x})\right] \tag{10.143}$$

$$\mathbb{E}_p\left[\frac{\partial^2 f(\boldsymbol{x})}{\partial \boldsymbol{x}\partial \boldsymbol{x}^\top}\right] = \mathbb{E}_p\left[\left(\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^\top\Sigma^{-1} - \Sigma^{-1}\right) f(\boldsymbol{x})\right] \tag{10.144}$$

Proof. For Gaussian distribution, from Bonnet's theorem [146], we know that

$$\nabla_{\boldsymbol{\mu}}\mathbb{E}_p[f(\boldsymbol{x})] = \mathbb{E}_p\left[\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\right]]. \tag{10.145}$$

From Theorem 2, we know that

$$\nabla_{\boldsymbol{\mu}}\mathbb{E}_p[f(\boldsymbol{x})] = \mathbb{E}_p\left[\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})f(\boldsymbol{x})\right]. \tag{10.146}$$

Thus, we can obtain that

$$\mathbb{E}_p\left[\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\right] = \mathbb{E}_p\left[\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})f(\boldsymbol{x})\right]. \tag{10.147}$$

From Price's Theorem [146], we know that

$$\nabla_\Sigma \mathbb{E}_p[f(\boldsymbol{x})] = \mathbb{E}_p\left[\frac{\partial^2 f(\boldsymbol{x})}{\partial \boldsymbol{x}\partial \boldsymbol{x}^\top}\right]. \tag{10.148}$$

From Theorem 2, we know that

$$\nabla_\Sigma \mathbb{E}_p[f(\boldsymbol{x})] = \frac{1}{2}\mathbb{E}_p\left[\left(\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^\top\Sigma^{-1} - \Sigma^{-1}\right) f(\boldsymbol{x})\right] \tag{10.149}$$

It follows that

$$\mathbb{E}_p\left[\frac{\partial^2 f(\boldsymbol{x})}{\partial \boldsymbol{x}\partial \boldsymbol{x}^\top}\right] = \mathbb{E}_p\left[\left(\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^\top\Sigma^{-1} - \Sigma^{-1}\right) f(\boldsymbol{x})\right] \tag{10.150}$$

□

10.8 Discrete Update

For function $f(\boldsymbol{x})$ over binary variable $\boldsymbol{x} \in \{0,1\}^d$, we employ Bernoulli distribution with parameter $\boldsymbol{p} = [p_1, \cdots, p_d]^\top$ as the underlying distribution, where p_i denote the probability of $x_i = 1$. The gradient of $\mathbb{E}_p[f(\boldsymbol{x})]$ w.r.t $\boldsymbol{p}$ can be derived as follows:

$$\nabla_{\boldsymbol{p}} \mathbb{E}_p[f(\boldsymbol{x})] = \mathbb{E}_p[f(\boldsymbol{x})\nabla_{\boldsymbol{p}} \log(p(\boldsymbol{x};\boldsymbol{p}))] \tag{10.151}$$

$$= \sum_{\boldsymbol{x}\in\{0,1\}^d} \prod_{i=1}^d p_i^{\mathbf{1}(x_i=1)}(1-p_i)^{\mathbf{1}(x_i=0)} f(\boldsymbol{x})\nabla_{\boldsymbol{p}} \log\left(\prod_{i=1}^d p_i^{\mathbf{1}(x_i=1)}(1-p_i)^{\mathbf{1}(x_i=0)}\right) \tag{10.152}$$

$$= \mathbb{E}_p\left[f(\boldsymbol{x})\nabla_{\boldsymbol{p}}\left(\sum_{i=1}^d \mathbf{1}(x_i=1)\log p_i + \mathbf{1}(x_i=0)\log(1-p_i)\right)\right] \tag{10.153}$$

$$= \mathbb{E}_p\left[f(\boldsymbol{x})\boldsymbol{h}\right] \tag{10.154}$$

where $\boldsymbol{h}_i = \frac{1}{p_i}\mathbf{1}(\boldsymbol{x}_i = 1) - \frac{1}{1-p_i}\mathbf{1}(\boldsymbol{x}_i = 0)$.

For function $f(\boldsymbol{x})$ over discrete variable $\boldsymbol{x} \in \{1, \cdots, K\}^d$, we employ categorical distribution with parameter $\boldsymbol{P} = [\boldsymbol{p}_1, \cdots, \boldsymbol{p}_d]^\top$ as the underlying distribution, where the ij-th element of $\boldsymbol{P}$ (i.e., $\boldsymbol{P}_{ij}$) denote the probability of $\boldsymbol{x}_i = j$. The gradient of

$\mathbb{E}_p[f(\boldsymbol{x})]$ w.r.t $\boldsymbol{P}$ can be derived as follows:

$$\nabla_{\boldsymbol{P}}\mathbb{E}_p[f(\boldsymbol{x})] = \mathbb{E}_p[f(\boldsymbol{x})\nabla_{\boldsymbol{P}}\log(p(\boldsymbol{x};\boldsymbol{P}))] \tag{10.155}$$

$$= \sum_{\boldsymbol{x}\in\{1,\cdots,K\}^d}\prod_{i=1}^{d}\prod_{j=1}^{K}\boldsymbol{P}_{ij}^{\mathbf{1}(\boldsymbol{x}_i=j)}f(\boldsymbol{x})\nabla_{\boldsymbol{P}}\log\big(\prod_{i=1}^{d}\prod_{j=1}^{K}\boldsymbol{P}_{ij}^{\mathbf{1}(\boldsymbol{x}_i=j)}\big) \tag{10.156}$$

$$= \mathbb{E}_p\left[f(\boldsymbol{x})\nabla_{\boldsymbol{P}}\big(\sum_{i=1}^{d}\sum_{j=1}^{K}\mathbf{1}(\boldsymbol{x}_i=j)\log\boldsymbol{P}_{ij}\big)\right] \tag{10.157}$$

$$= \mathbb{E}_p\left[f(\boldsymbol{x})\boldsymbol{H}\right] \tag{10.158}$$

where $\boldsymbol{H}_{ij} = \frac{1}{\boldsymbol{P}_{ij}}\mathbf{1}(\boldsymbol{x}_i=j)$.

10.9 Proof of Theorem 6

Lemma 3. *Suppose $f\in\mathcal{H}_k$ associated with $k(x,x)$, then $(m_t(x)-f(x))^2 \le \|f\|_{\mathcal{H}_k}^2\,\sigma_t^2(x)$*

Proof. Let $\alpha = \mathbf{K}_t^{-}\mathbf{k}_t(x)$. Then we have

$$(m_t(x)-f(x))^2 = \left(\sum_{i=1}^{t}\alpha_i f(x_i) - f(x)\right)^2 \tag{10.159}$$

$$= \left(\left\langle\sum_{i=1}^{t}\alpha_i k(x_i,\cdot) - k(x,\cdot), f\right\rangle\right)^2 \tag{10.160}$$

$$\le \langle f,f\rangle\left\langle\sum_{i=1}^{t}\alpha_i k(x_i,\cdot) - k(x,\cdot), \sum_{i=1}^{t}\alpha_i k(x_i,\cdot) - k(x,\cdot),\right\rangle \tag{10.161}$$

$$= \|f\|_{\mathcal{H}_k}^2\left\|\sum_{i=1}^{t}\alpha_i k(x_i,\cdot) - k(x,\cdot)\right\|_{\mathcal{H}_k}^2 \tag{10.162}$$

In addition, we can achieve that

$$\left\|\sum_{i=1}^{t}\alpha_i k(x_i,\cdot) - k(x,\cdot)\right\|_{\mathcal{H}_k}^2 = k(x,x) - 2\sum_{i=1}^{t}\alpha_i k(x_i,x) + \sum_{i=1}^{t}\sum_{j=1}^{t}\alpha_i\alpha_j k(x_i,x_j) \tag{10.163}$$

$$= k(x,x) - 2\alpha^T\mathbf{k}_t(x) + \alpha^T\mathbf{K}_t\alpha \tag{10.164}$$

$$= k(x,x) - 2\mathbf{k}_t(x)^T\mathbf{K}_t^{-}\mathbf{k}_t(x) + \mathbf{k}_t(x)^T\mathbf{K}_t^{-}\mathbf{K}_t\mathbf{K}_t^{-}\mathbf{k}_t(x) \tag{10.165}$$

$$= k(x,x) - \mathbf{k}_t(x)^T\mathbf{K}_t^{-}\mathbf{k}_t(x) \tag{10.166}$$

$$= \sigma_t^2(x) \tag{10.167}$$

Plug (10.167) into (10.162), we can attain $(m_t(x)-f(x))^2 \le \|f\|_{\mathcal{H}_k}^2\,\sigma_t^2(x)$.

□

Lemma 4. $f(x^*) - f(x_t) \leq 2 \|f\|_{\mathcal{H}_k} \sigma_{t-1}(x_t)$.

Proof. From Lemma 1 and Algorithm 1, we can achieve that

$$f(x^*) - f(x_t) \leq m_{t-1}(x^*) + \|f\|_{\mathcal{H}_k} \sigma_{t-1}(x^*) - f(x_t) \tag{10.168}$$
$$\leq m_{t-1}(x_t) + \|f\|_{\mathcal{H}_k} \sigma_{t-1}(x_t) - f(x_t) \tag{10.169}$$
$$\leq \|f\|_{\mathcal{H}_k} \sigma_{t-1}(x_t) + \|f\|_{\mathcal{H}_k} \sigma_{t-1}(x_t) \tag{10.170}$$
$$= 2\|f\|_{\mathcal{H}_k} \sigma_{t-1}(x_t) \tag{10.171}$$

□

Lemma 5. *Let* $\widehat{\sigma}_t^2(x) = k(x,x) - \mathbf{k}_t(x)^T(\sigma^2 I + \mathbf{K}_t)^{-}\mathbf{k}_t(x)$. *Then* $\sigma_t^2(x) \leq \widehat{\sigma}_t^2(x)$.

Proof. Since kernel matrix $\mathbf{K}_t$ is positive semi-definite, it follows that $\mathbf{K}_t = U^T \Lambda U$, where U is orthonormal matrix consists of eigenvectors, Λ is a diagonal matrix consists of eigenvalues.

Let $\beta = U\mathbf{k}_t(x)$, then we can achieve that

$$\mathbf{k}_t(x)^T(\sigma^2 I + \mathbf{K}_t)^{-}\mathbf{k}_t(x) = \sum_{i=1}^{t} \frac{\beta_i^2}{\sigma^2 + \lambda_i} \tag{10.172}$$
$$\leq \sum_{i=1}^{t} \frac{\beta_i^2}{\lambda_i} = \beta^T \Lambda^{-} \beta \tag{10.173}$$
$$= \mathbf{k}_t(x)^T U^T \Lambda^{-} U \mathbf{k}_t(x) \tag{10.174}$$
$$= \mathbf{k}_t(x)^T \mathbf{K}_t^{-} \mathbf{k}_t(x) \tag{10.175}$$

It follows that

$$\sigma_t^2(x) = k(x,x) - \mathbf{k}_t(x)^T \mathbf{K}_t^{-} \mathbf{k}_t(x) \tag{10.176}$$
$$\leq k(x,x) - \mathbf{k}_t(x)^T(\sigma^2 I + \mathbf{K}_t)^{-}\mathbf{k}_t(x) \tag{10.177}$$
$$= \widehat{\sigma}_t^2(x) \tag{10.178}$$

□

Now, we are ready to prove Theorem 6.

Proof. First, we have

$$R_T = \sum_{i=1}^{T} f(x^*) - f(x_t) \tag{10.179}$$

$$\leq 2\|f\|_{\mathcal{H}_k} \sum_{i=1}^{T} \sigma_{t-1}(x_t) \tag{10.180}$$

$$\leq 2\|f\|_{\mathcal{H}_k} \sqrt{T \sum_{i=1}^{T} \sigma_{t-1}^2(x_t)} \tag{10.181}$$

Since $s \leq \frac{1}{\log(1+\sigma^{-2})} \log(1+\sigma^{-2}s)$ for $s \in [0,1]$ and $0 \leq \widehat{\sigma}_{t-1}^2(x_t) \leq k(x,x) \leq 1$ for all $t \geq 1$, it follows that

$$\sum_{i=1}^{T} \sigma_{t-1}^2(x_t) \leq \sum_{i=1}^{T} \widehat{\sigma}_{t-1}^2(x_t) \leq \frac{1}{\log(1+\sigma^{-2})} \sum_{i=1}^{T} \log(1+\sigma^{-2}\widehat{\sigma}_{t-1}^2(x_t)) \tag{10.182}$$

$$\leq \frac{2\gamma_T}{\log(1+\sigma^{-2})} \tag{10.183}$$

Together (10.181) and (10.183), we can attain that

$$R_T \leq 2\|f\|_{H_k} \sqrt{T \frac{2\gamma_T}{\log(1+\sigma^{-2})}} \tag{10.184}$$

$$= \|f\|_{\mathcal{H}_k} \sqrt{T C_1 \gamma_T} \tag{10.185}$$

It follows that $r_T \leq \frac{R_T}{T} \leq \|f\|_{\mathcal{H}_k} \sqrt{\frac{C_1\gamma_T}{T}}$.

□

10.10 Proof of Theorem 7

Lemma 6. *Suppose $f \in \mathcal{H}_k$ associated with kernel $k(x,x)$, then $\left(\sum_{i=1}^{L} m_t(\widehat{x}_i) - \sum_{i=1}^{L} f(\widehat{x}_i)\right)^2 \leq \|f\|_{\mathcal{H}_k}^2 (\mathbf{1}^T \mathbf{A} \mathbf{1})$, where $\mathbf{A}$ denotes the kernel matrix (covariance matrix) with $\mathbf{A}_{ij} = k(\widehat{x}_i, \widehat{x}_j) - \mathbf{k}_t(\widehat{x}_i)^T \mathbf{K}_t^{-} \mathbf{k}_t(\widehat{x}_j)$.*

Proof. Let $\alpha^i = \mathbf{k}_t(\widehat{x}_i)^T \mathbf{K}_t^{-}$. Then we have

$$\left(\sum_{i=1}^{L} m_t(\widehat{x}_i) - \sum_{i=1}^{L} f(\widehat{x}_i)\right)^2 = \left(\sum_{i=1}^{L}\sum_{l=1}^{t} \alpha_l^i f(x_l) - \sum_{i=1}^{L} f(\widehat{x}_i)\right)^2 \tag{10.186}$$

$$= \left(\left\langle \sum_{i=1}^{L}\sum_{l=1}^{t} \alpha_l^i k(x_l, \cdot) - \sum_{i=1}^{L} k(\widehat{x}_i, \cdot), f \right\rangle\right)^2 \tag{10.187}$$

$$\leq \|f\|_{\mathcal{H}_k}^2 \left\| \sum_{i=1}^{L}\sum_{l=1}^{t} \alpha_l^i k(x_l, \cdot) - \sum_{i=1}^{L} k(\widehat{x}_i, \cdot) \right\|_{\mathcal{H}_k}^2 \tag{10.188}$$

In addition, we have

$$\left\|\sum_{i=1}^{L}\sum_{l=1}^{t}\alpha_l^i k\left(x_l,\cdot\right)-\sum_{i=1}^{L}k(\widehat{x}_i,\cdot)\right\|_{\mathcal{H}_k}^2$$

$$=\sum_{i=1}^{L}\sum_{j=1}^{L}k(\widehat{x}_i,\widehat{x}_j)-2\sum_{i=1}^{L}\sum_{j=1}^{L}\sum_{l=1}^{t}\alpha_l^i k\left(x_l,\widehat{x}_j\right)+\sum_{i=1}^{L}\sum_{j=1}^{L}\sum_{n=1}^{t}\sum_{l=1}^{t}\alpha_l^i\alpha_n^j k\left(x_l,x_n\right) \tag{10.189}$$

$$=\sum_{i=1}^{L}\sum_{j=1}^{L}k(\widehat{x}_i,\widehat{x}_j)-2\sum_{i=1}^{L}\sum_{j=1}^{L}\mathbf{k}_t(\widehat{x}_i)^T\mathbf{K}_t^{-}\mathbf{k}_t(\widehat{x}_j)+\sum_{i=1}^{L}\sum_{j=1}^{L}\mathbf{k}_t(\widehat{x}_i)^T\mathbf{K}_t^{-}\mathbf{k}_t(\widehat{x}_j) \tag{10.190}$$

$$=\sum_{i=1}^{L}\sum_{j=1}^{L}\mathbf{A}_{ij}=\mathbf{1}^T\mathbf{A}\mathbf{1} \tag{10.191}$$

Thus, we obtain $\left(\sum_{i=1}^{L}m_t(\widehat{x}_i)-\sum_{i=1}^{L}f(\widehat{x}_i)\right)^2\leq\|f\|_{\mathcal{H}_k}^2\left(\mathbf{1}^T\mathbf{A}\mathbf{1}\right)$.

□

Lemma 7. *Suppose $f\in\mathcal{H}_k$ associated with kernel $k(x,x)$, then $\frac{1}{L}\sum_{i=1}^{L}\left(f(x^*)-f(x_{(n-1)L+i})\right)$ $\leq 2\|f\|_{\mathcal{H}_k}\sqrt{\frac{tr(\mathrm{cov}_{n-1}(X_n,X_n))}{L}}$, where covariance matrix $\mathrm{cov}_{n-1}(X_n,X_n)$ constructed as Eq.(4.8) and $X_n=\{x_{(n-1)L+1},...,x_{nL}\}$.*

Proof. Let $X^*=\{x^*,...,x^*\}$ be L copies of x^*. Then, we obtain that

$$\frac{1}{L}\sum_{i=1}^{L}\left(f(x^*)-f(x_{(n-1)L+i})\right)=f(x^*)-\frac{1}{L}\sum_{i=1}^{L}f(x_{(n-1)L+i}) \tag{10.192}$$

$$\leq m_{(n-1)L}(x^*)+\|f\|_{\mathcal{H}_k}\sigma_{(n-1)L}(x^*)-\frac{1}{L}\sum_{i=1}^{L}f(x_{(n-1)L+i}) \tag{10.193}$$

$$=\frac{1}{L}\sum_{i=1}^{L}m_{(n-1)L}(x^*)+\|f\|_{\mathcal{H}_k}\left(2\sqrt{\frac{tr\left(\mathrm{cov}_{n-1}(X^*,X^*)\right)}{L}}-\sqrt{\frac{\mathbf{1}^T\mathrm{cov}_{n-1}(X^*,X^*)\mathbf{1}}{L^2}}\right)$$
$$-\frac{1}{L}\sum_{i=1}^{L}f(x_{(n-1)L+i}) \tag{10.194}$$

$$\leq\frac{1}{L}\sum_{i=1}^{L}m_{(n-1)L}(x_{(n-1)L+i})+\|f\|_{\mathcal{H}_k}\left(2\sqrt{\frac{tr\left(\mathrm{cov}_{n-1}(X_n,X_n)\right)}{L}}-\sqrt{\frac{\mathbf{1}^T\mathrm{cov}_{n-1}(X_n,X_n)\mathbf{1}}{L^2}}\right)$$
$$-\frac{1}{L}\sum_{i=1}^{L}f(x_{(n-1)L+i}) \tag{10.195}$$

It follows that

$$\frac{1}{L}\sum_{i=1}^{L}\left(f(x^*)-f(x_{(n-1)L+i})\right)=f(x^*)-\frac{1}{L}\sum_{i=1}^{L}f(x_{(n-1)L+i}) \tag{10.196}$$

$$\leq \|f\|_{\mathcal{H}_k}\left(2\sqrt{\frac{tr\left(\mathrm{cov}_{n-1}(X_n,X_n)\right)}{L}}-\sqrt{\frac{\mathbf{1}^T\mathrm{cov}_{n-1}(X_n,X_n)\mathbf{1}}{L^2}}\right)+\|f\|_{\mathcal{H}_k}\sqrt{\frac{\mathbf{1}^T\mathrm{cov}_{n-1}(X_n,X_n)\mathbf{1}}{L^2}} \tag{10.197}$$

$$=2\|f\|_{\mathcal{H}_k}\sqrt{\frac{tr\left(\mathrm{cov}_{n-1}(X_n,X_n)\right)}{L}} \tag{10.198}$$

□

Lemma 8. *Let B_n and A_n be the covariance matrix constructed by Eq.(4.8) and Eq.(4.17), respectively. Then $tr(B_n)\leq tr(A_n)$*

Proof. It follows directly from Lemma 5. □

Lemma 9. *Let matrix $A_{n-1}=\mathrm{cov}_{n-1}(X_n,X_n)$ as Eq.(4.17). Denote the spectral norm of matrix A_{n-1} as $\beta_{n-1}=\|A_{n-1}\|_2$.*
Then $\mathrm{tr}(A_{n-1})\leq\frac{\beta_{n-1}}{\log(1+\beta_{n-1}\sigma^{-2})}\log\det(I+\sigma^{-2}A_{n-1})$ for any $\sigma\neq 0$.

Proof. Since A_{n-1} is a positive semidefinite matrix, we can attain that the eigenvalues of A_{n-1} are all nonnegative. Without loss of generality, assume eigenvalues of A_{n-1} as $0\leq\lambda_L\leq...\leq\lambda_1$. By the definition of the spectral norm $\beta_{n-1}=\|A_{n-1}\|_2$, we obtain that $0\leq\lambda_L\leq...\leq\lambda_1\leq\beta_{n-1}$

Since $s\leq\frac{\beta_{n-1}}{\log(1+\beta_{n-1}\sigma^{-2})}\log(1+\sigma^{-2}s)$ for $s\in[0,\beta_{n-1}]$ and $0\leq\lambda_i\leq\beta_{n-1}$, $i\in\{1,...,L\}$, we can obtain that inequality (10.199) holds true for all $i\in\{1,...,L\}$

$$\lambda_i\leq\frac{\beta_{n-1}}{\log(1+\beta_{n-1}\sigma^{-2})}\log(1+\sigma^{-2}\lambda_i) \tag{10.199}$$

Because $\log\det(I+\sigma^{-2}A_{n-1})=\sum_{i=1}^{L}\log(1+\sigma^{-2}\lambda_i)$, we can achieve that

$$\mathrm{tr}(A_{n-1})=\sum_{i=1}^{L}\lambda_i\leq\frac{\beta_{n-1}}{\log(1+\beta_{n-1}\sigma^{-2})}\log\det(I+\sigma^{-2}A_{n-1}) \tag{10.200}$$

□

Lemma 10. *Let $T=NL$, $\mathbf{K}_T$ be the $T\times T$ sized kernel matrix and $\mathbf{I}_L$ be the $L\times L$ sized idendity matrix. Then $\frac{1}{2}\log\det(I+\sigma^{-2}\mathbf{K}_T)=\frac{1}{2}\sum_{n=1}^{N}\log\det(\mathbf{I}_L+\sigma^{-2}A_{n-1})$, where matrix $A_{n-1}=\widehat{\mathrm{cov}}_{n-1}(X_n,X_n)$ as Eq.(4.17).*

Proof.

$$\frac{1}{2}\log\det\left(\mathbf{I}_T+\sigma^{-2}\mathbf{K}_T\right)=\frac{1}{2}\log\det\left(\sigma^2\mathbf{I}_T+\mathbf{K}_T\right)-\frac{1}{2}\log\det\left(\sigma^2\mathbf{I}_T\right) \tag{10.201}$$

Using the Schur's determinant identity $\det\begin{pmatrix} A & B \\ C & D \end{pmatrix}=\det(A)\cdot\det(D-CA^{-1}B)$ in linear algebra, set $A=\sigma^2\mathbf{I}_{(N-1)L}+\mathbf{K}\left(\overline{X}_{N-1},\overline{X}_{N-1}\right)$, $B=\mathbf{K}\left(\overline{X}_{N-1},X_N\right)$, $C=B^T$ and $D=\sigma^2\mathbf{I}_L+\mathbf{K}\left(X_N,X_N\right)$, where $\overline{X}_{N-1}=\{x_1,...,x_{(N-1)L}\}$ denote all previous $N-1$ batch of points, $X_N=\{x_{(N-1)L+1},...,x_{NL}\}$ denote the N^{th} batch of points and $\mathbf{K}(\cdot,\cdot)$ denote the kernel matrix constructed by its input. Then, we can achieve that

$$\begin{aligned}
&\frac{1}{2}\log\det\left(\sigma^2\mathbf{I}_T+\mathbf{K}_T\right)-\frac{1}{2}\log\det\left(\sigma^2\mathbf{I}_T\right) \qquad (10.202)\\
&=\frac{1}{2}\log\det\left(\sigma^2\mathbf{I}_{(N-1)L}+\mathbf{K}\left(\overline{X}_{N-1},\overline{X}_{N-1}\right)\right)+\frac{1}{2}\log\det\left(\sigma^2\mathbf{I}_L+A_{N-1}\right)-\frac{1}{2}\log\det\left(\sigma^2\mathbf{I}_T\right)\\
&=\frac{1}{2}\log\det\left(\sigma^2\mathbf{I}_{(N-1)L}+\mathbf{K}\left(\overline{X}_{N-1},\overline{X}_{N-1}\right)\right)+\frac{1}{2}\log\det\left(\mathbf{I}_L+\sigma^{-2}A_{N-1}\right)-\frac{1}{2}\log\det\left(\sigma^2\mathbf{I}_{(N-1)L}\right)
\end{aligned}$$

where $A_{N-1}=\mathrm{cov}_{N-1}\left(X_N,X_N\right)$ is the covariance matrix between X_N and X_N constructed as Eq.(4.17).

By induction, we can achieve $\frac{1}{2}\log\det\left(\mathbf{I}_T+\sigma^{-2}\mathbf{K}_T\right)=\frac{1}{2}\sum_{n=1}^{N}\log\det\left(\mathbf{I}_L+\sigma^{-2}A_{n-1}\right)$

□

Finally, we are ready to attain Theorem 2.

Proof. Let covariance matrix $\mathbf{A}_{n-1}$ and $\mathbf{B}_{n-1}$ be constructed as Eq.(4.17) and Eq. (4.8),

respectively. Let $\beta_{n-1} = \|\mathbf{A}_{n-1}\|_2$. Then, we can achieve that

$$R_T = \sum_{t=1}^{T} f(x^*) - f(x_t) \tag{10.203}$$

$$\leq 2\|f\|_{\mathcal{H}_k} \sum_{n=1}^{N} \sqrt{L\ tr\left(\mathbf{B}_{n-1}\right)} \tag{10.204}$$

$$\leq 2\|f\|_{\mathcal{H}_k} \sum_{n=1}^{N} \sqrt{L\ tr\left(\mathbf{A}_{n-1}\right)} \tag{10.205}$$

$$\leq 2\|f\|_{\mathcal{H}_k} \sqrt{NL \sum_{n=1}^{N} tr\left(\mathbf{A}_{n-1}\right)} \tag{10.206}$$

$$\leq 2\|f\|_{\mathcal{H}_k} \sqrt{T \sum_{n=1}^{N} \frac{\beta_{n-1}}{\log\left(1+\beta_{n-1}\sigma^{-2}\right)} \log\det\left(I+\sigma^{-2}\mathbf{A}_{n-1}\right)} \tag{10.207}$$

$$\leq \|f\|_{\mathcal{H}_k} \sqrt{TC_2 \sum_{n=1}^{N} \log\det\left(I+\sigma^{-2}\mathbf{A}_{n-1}\right)} \tag{10.208}$$

$$\leq \|f\|_{\mathcal{H}_k} \sqrt{TC_2\gamma_T} \tag{10.209}$$

It follows that $r_T \leq \frac{R_T}{T} \leq \|f\|_{\mathcal{H}_k} \sqrt{\frac{C_2\gamma_T}{T}}$

□

10.11 Proof of Theorem 8

Lemma 11. *Suppose $h = f + g \in \mathcal{H}_k^\sigma$ associated with kernel $k^\sigma(x,y) = k(x,y) + \sigma^2\delta(x,y)$. Suppose $f \in \mathcal{H}_k$ associated with k and $g \in \mathcal{H}_{\sigma^2\delta}$ associated with kernel $\sigma^2\delta$. Then for $x \neq x_i, i \in \{1,...,t\}$, we have $|\widehat{m}_t(x) - f(x)| \leq \|h\|_{\mathcal{H}_{k^\sigma}} \widehat{\sigma}_t(x) + \left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma$.*

Proof. Let $\alpha = (\mathbf{K}_t + \sigma^2 I)^{-1}\mathbf{k}_t(x)$. Then we have

$$\left(\widehat{m}_t(x) - h(x)\right)^2 = \left(\sum_{i=1}^{t} \alpha_i h\left(x_i\right) - h(x)\right)^2 \tag{10.210}$$

$$= \left(\left\langle \sum_{i=1}^{t} \alpha_i k^\sigma\left(x_i,\cdot\right) - k^\sigma(x,\cdot), h \right\rangle\right)^2 \tag{10.211}$$

$$\leq \|h\|_{\mathcal{H}_{k^\sigma}}^2 \left\| \sum_{i=1}^{t} \alpha_i k^\sigma\left(x_i,\cdot\right) - k^\sigma(x,\cdot) \right\|_{\mathcal{H}_{k^\sigma}}^2 \tag{10.212}$$

In addition, we can achieve that

$$\left\|\sum_{i=1}^{t}\alpha_i k^{\sigma}\left(x_i,\cdot\right)-k^{\sigma}(x,\cdot)\right\|_{\mathcal{H}_{k^{\sigma}}}^{2}=k^{\sigma}(x,x)-2\sum_{i=1}^{t}\alpha_i k^{\sigma}\left(x_i,x\right)+\sum_{i=1}^{t}\sum_{j=1}^{t}\alpha_i\alpha_j k^{\sigma}\left(x_i,x_j\right)$$

$$=k(x,x)+\sigma^2-2\alpha^T\mathbf{k}_t(x)+\alpha^T(\mathbf{K}_t+\sigma^2 I)\alpha \tag{10.213}$$

$$=k(x,x)+\sigma^2-\mathbf{k}_t(x)^T(\mathbf{K}_t+\sigma^2 I)^{-1}\mathbf{k}_t(x) \tag{10.214}$$

$$=\widehat{\sigma}_t^2(x)+\sigma^2 \tag{10.215}$$

Plug (10.215) into (10.212), we can obtain $\left(\widehat{m}_t(x)-h(x)\right)^2\leq\|h\|_{\mathcal{H}_{k^{\sigma}}}^2\left(\widehat{\sigma}_t^2(x)+\sigma^2\right)$. Thus, we achieve that

$$\left|\widehat{m}_t(x)-f(x)\right|\leq\left|\widehat{m}_t(x)-h(x)\right|+|g(x)| \tag{10.216}$$

$$\leq\|h\|_{\mathcal{H}_{k^{\sigma}}}\sqrt{\widehat{\sigma}_t^2(x)+\sigma^2}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\sigma \tag{10.217}$$

$$\leq\|h\|_{\mathcal{H}_{k^{\sigma}}}\widehat{\sigma}_t(x)+\left(\|h\|_{\mathcal{H}_{k^{\sigma}}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{10.218}$$

□

Lemma 12. *Under same condition as Lemma 11, we have* $f(x^*)-f(x_t)\leq 2\|h\|_{\mathcal{H}_{k^{\sigma}}}\widehat{\sigma}_{t-1}(x_t)+2\left(\|h\|_{\mathcal{H}_{k^{\sigma}}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma$.

Proof. From Lemma 11 and Algorithm **??**, we can achieve that

$$f(x^*)-f(x_t)\leq\widehat{m}_{t-1}(x^*)+\|h\|_{\mathcal{H}_{k^{\sigma}}}\widehat{\sigma}_{t-1}(x^*)+\left(\|h\|_{\mathcal{H}_{k^{\sigma}}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma-f(x_t) \tag{10.219}$$

$$\leq\widehat{m}_{t-1}(x_t)+\|h\|_{\mathcal{H}_{k^{\sigma}}}\widehat{\sigma}_{t-1}(x_t)+\left(\|h\|_{\mathcal{H}_{k^{\sigma}}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma-f(x_t) \tag{10.220}$$

$$\leq 2\|h\|_{\mathcal{H}_{k^{\sigma}}}\widehat{\sigma}_{t-1}(x_t)+2\left(\|h\|_{\mathcal{H}_{k^{\sigma}}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{10.221}$$

□

Finally, we are ready to prove Theorem 8.

Proof. First, we have

$$R_T=\sum_{i=1}^{T}f(x^*)-f(x_t) \tag{10.222}$$

$$\leq 2\|h\|_{\mathcal{H}_{k^{\sigma}}}\sum_{i=1}^{T}\widehat{\sigma}_{t-1}(x_t)+2T\left(\|h\|_{\mathcal{H}_{k^{\sigma}}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{10.223}$$

$$\leq 2\|h\|_{\mathcal{H}_{k^{\sigma}}}\sqrt{T\sum_{i=1}^{T}\widehat{\sigma}_{t-1}^2(x_t)}+2T\left(\|h\|_{\mathcal{H}_{k^{\sigma}}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{10.224}$$

Since $s \leq \frac{B}{\log(1+B\sigma^{-2})} \log\left(1+\sigma^{-2}s\right)$ for $s \in [0, B]$ and $0 \leq \widehat{\sigma}_{t-1}^2(x_t) \leq k^\sigma(x,x) \leq B$ for all $t \geq 1$, it follows that

$$\sum_{i=1}^{T} \widehat{\sigma}_{t-1}^2(x_t) \leq \frac{B}{\log(1+B\sigma^{-2})} \sum_{i=1}^{T} \log(1+\sigma^{-2}\widehat{\sigma}_{t-1}^2(x_t)) \tag{10.225}$$

$$\leq \frac{2B\gamma_T}{\log(1+B\sigma^{-2})} \tag{10.226}$$

Together (10.224) and (10.226), we can attain that

$$R_T \leq 2\|h\|_{\mathcal{H}_{k^\sigma}} \sqrt{T\frac{2B\gamma_T}{\log(1+B\sigma^{-2})}} + 2T\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{10.227}$$

$$= \|h\|_{\mathcal{H}_{k^\sigma}} \sqrt{TC_3\gamma_T} + 2T\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{10.228}$$

It follows that $r_T \leq \frac{R_T}{T} \leq \|h\|_{\mathcal{H}_{k^\sigma}} \sqrt{\frac{C_3\gamma_T}{T}} + 2\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma$.

□

10.12 Proof of Theorem 9

Lemma 13. *Suppose $h = f + g \in \mathcal{H}_k^\sigma$ associated with kernel $k^\sigma(x,y) = k(x,y) + \sigma^2\delta(x,y)$. Suppose $f \in \mathcal{H}_k$ associated with k and $g \in \mathcal{H}_{\sigma^2\delta}$ associated with kernel $\sigma^2\delta$. Suppose $\widehat{x}_i \neq x_j, i \in \{1,...,L\}, j \in \{1,...,t\}$, then we have*

$$\left|\sum_{i=1}^{L} m_t(\hat{x}_i) - \sum_{i=1}^{L} f(\hat{x}_i)\right| \leq \|h\|_{\mathcal{H}_{k^\sigma}} \sqrt{\mathbf{1}^T\mathbf{A}\mathbf{1} + L^2\sigma^2} + L\|g\|_{\mathcal{H}_{\sigma^2\delta}}\sigma \tag{10.229}$$

where $\mathbf{A}$ denotes the kernel covariance matrix with $\mathbf{A}_{ij} = k(\widehat{x}_i, \widehat{x}_j) - \mathbf{k}_t(\widehat{x}_i)^T(\mathbf{K}_t + \sigma^2 I)^{-1}\mathbf{k}_t(\widehat{x}_j)$

Remark: Further require $\widehat{x}_i \neq \widehat{x}_j, \forall i,j \in \{1,...,L\}$ can lead to a tighter bound as

$$\left|\sum_{i=1}^{L} m_t(\hat{x}_i) - \sum_{i=1}^{L} f(\hat{x}_i)\right| \leq \|h\|_{\mathcal{H}_{k^\sigma}} \sqrt{\mathbf{1}^T\mathbf{A}\mathbf{1} + L\sigma^2} + L\|g\|_{\mathcal{H}_{\sigma^2\delta}}\sigma \tag{10.230}$$

Proof. Let $\alpha^i = (\mathbf{K}_t + \sigma^2 I)^{-1}\mathbf{k}_t(\widehat{x}_i)$. Then we have

$$\left(\sum_{i=1}^{L} \widehat{m}_t(\widehat{x}_i) - \sum_{i=1}^{L} h(\widehat{x}_i)\right)^2 = \left(\sum_{i=1}^{L}\sum_{l=1}^{t} \alpha_l^i h(x_l) - \sum_{i=1}^{L} h(\widehat{x}_i)\right)^2 \tag{10.231}$$

$$= \left(\left\langle \sum_{i=1}^{L}\sum_{l=1}^{t} \alpha_l^i k^\sigma(x_l,\cdot) - \sum_{i=1}^{L} k^\sigma(\widehat{x}_i,\cdot), h\right\rangle\right)^2 \tag{10.232}$$

$$\leq \|h\|_{\mathcal{H}_{k^\sigma}}^2 \left\|\sum_{i=1}^{L}\sum_{l=1}^{t} \alpha_l^i k^\sigma(x_l,\cdot) - \sum_{i=1}^{L} k^\sigma(\widehat{x}_i,\cdot)\right\|_{\mathcal{H}_{k^\sigma}}^2 \tag{10.233}$$

In addition, we have

$$\begin{aligned}
&\left\|\sum_{i=1}^{L}\sum_{l=1}^{t}\alpha_l^i k^\sigma\left(x_l,\cdot\right)-\sum_{i=1}^{L}k^\sigma(\widehat{x}_i,\cdot)\right\|_{\mathcal{H}_{k^\sigma}}^2 \\
&=\sum_{i=1}^{L}\sum_{j=1}^{L}k^\sigma(\widehat{x}_i,\widehat{x}_j)-2\sum_{i=1}^{L}\sum_{j=1}^{L}\sum_{l=1}^{t}\alpha_l^i k^\sigma\left(x_l,\widehat{x}_j\right)+\sum_{i=1}^{L}\sum_{j=1}^{L}\sum_{n=1}^{t}\sum_{l=1}^{t}\alpha_l^i\alpha_n^j k^\sigma\left(x_l,x_n\right) \qquad (10.234)\\
&\leq\sum_{i=1}^{L}\sum_{j=1}^{L}k(\widehat{x}_i,\widehat{x}_j)+L^2\sigma^2-2\sum_{i=1}^{L}\sum_{j=1}^{L}\mathbf{k}_t(\widehat{x}_i)^T(\mathbf{K}_t+\sigma^2 I)^{-1}\mathbf{k}_t(\widehat{x}_j) \\
&\quad+\sum_{i=1}^{L}\sum_{j=1}^{L}\mathbf{k}_t(\widehat{x}_i)^T(\mathbf{K}_t+\sigma^2 I)^{-1}\mathbf{k}_t(\widehat{x}_j) \qquad (10.235)\\
&=\sum_{i=1}^{L}\sum_{j=1}^{L}\mathbf{A}_{ij}+L^2\sigma^2=\mathbf{1}^T\mathbf{A}\mathbf{1}+L^2\sigma^2 \qquad (10.236)
\end{aligned}$$

Thus, we obtain $\left(\sum_{i=1}^{L}m_t(\widehat{x}_i)-\sum_{i=1}^{L}h(\widehat{x}_i)\right)^2\leq\|h\|_{\mathcal{H}_{k^\sigma}}^2\left(\mathbf{1}^T\mathbf{A}\mathbf{1}+L^2\sigma^2\right)$. Then, we can achieve that

$$\begin{aligned}
\left|\sum_{i=1}^{L}m_t(\hat{x}_i)-\sum_{i=1}^{L}f(\hat{x}_i)\right|&\leq\left|\sum_{i=1}^{L}m_t(\hat{x}_i)-\sum_{i=1}^{L}h(\hat{x}_i)\right|+\sum_{i=1}^{L}|g(\hat{x}_i)| \qquad (10.237)\\
&\leq\|h\|_{\mathcal{H}_{k^\sigma}}\sqrt{\mathbf{1}^T\mathbf{A}\mathbf{1}+L^2\sigma^2}+L\|g\|_{\mathcal{H}_{\sigma^2\delta}}\sigma \qquad (10.238)
\end{aligned}$$

□

Lemma 14. *Suppose $h=f+g\in\mathcal{H}_k^\sigma$ associated with kernel $k^\sigma(x,y)=k(x,y)+\sigma^2\delta(x,y)$. Suppose $f\in\mathcal{H}_k$ associated with k and $g\in\mathcal{H}_{\sigma^2\delta}$ associated with kernel $\sigma^2\delta$. Suppose $x_i\neq x_j$, then we have*

$$\frac{1}{L}\sum_{i=1}^{L}\left(f(x^*)-f(x_{(n-1)L+i})\right)\leq 2\|h\|_{\mathcal{H}_{k^\sigma}}\sqrt{\frac{tr\left(\widehat{\mathrm{cov}}_{n-1}(X_n,X_n)\right)}{L}}+2\left(\|h\|_{\mathcal{H}_{k^\sigma}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \qquad (10.239)$$

where covariance matrix $\widehat{\mathrm{cov}}_{n-1}(X_n,X_n)$ is constructed as Eq.(4.17) with $X_n=\{x_{(n-1)L+1},...,x_{nL}\}$.

Proof. Let $X^*=\{x^*,...,x^*\}$ be L copies of x^*. Then, we obtain that

$$\frac{1}{L}\sum_{i=1}^{L}\left(f(x^*)-f(x_{(n-1)L+i})\right)=f(x^*)-\frac{1}{L}\sum_{i=1}^{L}f(x_{(n-1)L+i}) \tag{10.240}$$

$$\leq \widehat{m}_{(n-1)L}(x^*)+\|h\|_{\mathcal{H}_{k^\sigma}}\widehat{\sigma}_{(n-1)L}(x^*)+\left(\|h\|_{\mathcal{H}_{k^\sigma}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma-\frac{1}{L}\sum_{i=1}^{L}f(x_{(n-1)L+i}) \tag{10.241}$$

$$=\frac{1}{L}\sum_{i=1}^{L}\widehat{m}_{(n-1)L}(x^*)+\|h\|_{\mathcal{H}_{k^\sigma}}\left(2\sqrt{\frac{tr\left(\widehat{\mathrm{cov}}_{n-1}(X^*,X^*)\right)}{L}}-\sqrt{\frac{\mathbf{1}^T\widehat{\mathrm{cov}}_{n-1}(X^*,X^*)\mathbf{1}}{L^2}}\right)$$
$$+\left(\|h\|_{\mathcal{H}_{k^\sigma}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma-\frac{1}{L}\sum_{i=1}^{L}f(x_{(n-1)L+i}) \tag{10.242}$$

$$\leq\frac{1}{L}\sum_{i=1}^{L}\widehat{m}_{(n-1)L}(x_{(n-1)L+i})+\|h\|_{\mathcal{H}_{k^\sigma}}\left(2\sqrt{\frac{tr\left(\widehat{\mathrm{cov}}_{n-1}(X_n,X_n)\right)}{L}}-\sqrt{\frac{\mathbf{1}^T\widehat{\mathrm{cov}}_{n-1}(X_n,X_n)\mathbf{1}}{L^2}}\right)$$
$$+\left(\|h\|_{\mathcal{H}_{k^\sigma}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma-\frac{1}{L}\sum_{i=1}^{L}f(x_{(n-1)L+i}) \tag{10.243}$$

$$\leq\|h\|_{\mathcal{H}_{k^\sigma}}\left(2\sqrt{\frac{tr\left(\widehat{\mathrm{cov}}_{n-1}(X_n,X_n)\right)}{L}}-\sqrt{\frac{\mathbf{1}^T\widehat{\mathrm{cov}}_{n-1}(X_n,X_n)\mathbf{1}}{L^2}}\right)+\left(\|h\|_{\mathcal{H}_{k^\sigma}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma$$
$$+\|h\|_{\mathcal{H}_{k^\sigma}}\sqrt{\frac{\mathbf{1}^T\widehat{\mathrm{cov}}_{n-1}(X_n,X_n)\mathbf{1}+L^2\sigma^2}{L^2}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\sigma \tag{10.244}$$

$$\leq 2\|h\|_{\mathcal{H}_{k^\sigma}}\sqrt{\frac{tr\left(\widehat{\mathrm{cov}}_{n-1}(X_n,X_n)\right)}{L}}+2\left(\|h\|_{\mathcal{H}_{k^\sigma}}+\|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{10.245}$$

□

Finally, we are ready to attain Theorem 4.

Proof. Let $\mathbf{A}_{n-1}=\widehat{\mathrm{cov}}_{n-1}(X_n,X_n)$ be the covariance matrix constructed as Eq.(4.17)

with $X_n = \{x_{(n-1)L+1}, ..., x_{nL}\}$. Let $\beta_{n-1} = \|\mathbf{A}_{n-1}\|_2$. Then, we can achieve that

$$R_T = \sum_{t=1}^{T} f(x^*) - f(x_t) \tag{10.246}$$

$$\leq 2\|h\|_{\mathcal{H}_{k^\sigma}} \sum_{n=1}^{N} \sqrt{L\, tr\left(\mathbf{A}_{n-1}\right)} + 2T\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{10.247}$$

$$\leq 2\|h\|_{\mathcal{H}_{k^\sigma}} \sqrt{NL \sum_{n=1}^{N} tr\left(\mathbf{A}_{n-1}\right)} + 2T\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{10.248}$$

$$\leq 2\|h\|_{\mathcal{H}_{k^\sigma}} \sqrt{T \sum_{n=1}^{N} \frac{\beta_{n-1}}{\log\left(1+\beta_{n-1}\sigma^{-2}\right)} \log\det\left(I+\sigma^{-2}\mathbf{A}_{n-1}\right)} + 2T\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{10.249}$$

$$\leq \|h\|_{\mathcal{H}_{k^\sigma}} \sqrt{TC_4 \sum_{n=1}^{N} \log\det\left(I+\sigma^{-2}\mathbf{A}_{n-1}\right)} + 2T\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{10.250}$$

$$\leq \|h\|_{\mathcal{H}_{k^\sigma}} \sqrt{TC_4\gamma_T} + 2T\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma \tag{10.251}$$

It follows that $r_T \leq \frac{R_T}{T} \leq \|h\|_{\mathcal{H}_{k^\sigma}} \sqrt{\frac{C_4\gamma_T}{T}} + 2T\left(\|h\|_{\mathcal{H}_{k^\sigma}} + \|g\|_{\mathcal{H}_{\sigma^2\delta}}\right)\sigma$

□

10.13 Proof of Theorem 10

Proof.

$$\begin{aligned}
\widetilde{r}_T &= \min_{t\in[T]} \sup_{f_t\in\mathcal{B}_k, f_t(x_i)=f_i(x_i), \forall i\in[t-1]} \{f_t(x^*) - f_t(x_t)\} \\
&\leq \sup_{f_T\in\mathcal{B}_k, f_T(x_i)=f_i(x_i), \forall i\in[T-1]} \{f_T(x^*) - f_T(x_T)\} \\
&\leq \sup_{f_T\in\mathcal{B}_k, f_T(x_i)=f_i(x_i), \forall i\in[T-1]} \{m_{T-1}(x^*) + B\sigma_{T-1}(x^*) - f_T(x_T)\} \\
&\leq \sup_{f_T\in\mathcal{B}_k, f_T(x_i)=f_i(x_i), \forall i\in[T-1]} \{m_{T-1}(x_T) + B\sigma_{T-1}(x_T) - f_T(x_T)\} \\
&\leq \sup_{f_T\in\mathcal{B}_k, f_T(x_i)=f_i(x_i), \forall i\in[T-1]} \{B\sigma_{T-1}(x_T) + B\sigma_{T-1}(x_T)\} \\
&\leq 2B\sigma_{T-1}(x_T)
\end{aligned} \tag{10.252}$$

Applying Theorem 5.4 in [87] with $h_{\rho,X} \leq h_X$, we can obtain that

$$\sigma_{T-1}(x_T) \leq Ch_X^{s-d/2} \tag{10.253}$$

Together with (10.252) and (10.253), absorbing the constant into C, we can achieve that $\widetilde{r}_T \leq Ch_X^{s-d/2}$

□

10.14 Proof of Theorem 11

Proof. From , we know $\widetilde{r}_T \leq 2B\sigma_{T-1}(x_T)$. By applying Theorem 11.22 in [172], we can obtain that

$$2B\sigma_{T-1}(x_T) \leq 2B\exp(c\log(h_X)/(2\sqrt{h_X})) \tag{10.254}$$

It follows that $\widetilde{r}_T \leq 2B\exp(c\log(h_X)/(2\sqrt{h_X}))$. □

10.15 Proof of Theorem 12

Proof.

$$\begin{aligned}
\widetilde{r}_T &= \min_{t\in[T]} \sup_{f_t\in\mathcal{B}_k} \{f_t(x^*) - f_t(x_t)\} \\
&= \min_{t\in[T]} \sup_{f_t\in\mathcal{B}_k} \{\langle k(x^*,\cdot) - k(x_t,\cdot), f_t\rangle\} \\
&\leq \min_{t\in[T]} B\sqrt{\langle k(x^*,\cdot) - k(x_t,\cdot), k(x^*,\cdot) - k(x_t,\cdot)\rangle} \\
&\leq \min_{t\in[T]} B\sqrt{(k(x^*,x^*) + k(x_t,x_t) - 2k(x^*,x_t))} \\
&\leq B\sqrt{(2-2\Phi(h_X))}
\end{aligned} \tag{10.255}$$

□

10.16 Proof of Corollary 2

Proof. From Theorem 12, we can obtain that

$$\begin{aligned}
\widetilde{r}_T &= \min_{t\in[T]} \sup_{f_t\in\mathcal{B}_k} \{f_t(x^*) - f_t(x_t)\} \\
&\leq B\sqrt{(2-2\Phi(h_X))} \\
&= B\sqrt{(2-2\exp(-Ch_X^2))} \\
&\leq B\sqrt{2(Ch_X^2)} \\
&= B\sqrt{2C}h_X = \mathcal{O}(h_X)
\end{aligned} \tag{10.256}$$

□

10.17 Proof of Theorem 13

Theorem. *Suppose n is a prime number and $2d|(n-1)$. Let g be a primitive root of n. Let $\boldsymbol{z} = [g^0, g^{\frac{n-1}{2d}}, g^{\frac{2(n-1)}{2d}}, \cdots, g^{\frac{(d-1)(n-1)}{2d}}]$ mod n. Construct a rank-1 lattice $X = \{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_{n-1}\}$ with $\boldsymbol{x}_i = \frac{i\boldsymbol{z} \bmod n}{n}, i \in \{0, ..., n-1\}$. Then, there are $\frac{n-1}{2d}$ distinct pairwise toroidal distance values among X, and for each distance value, there are the same number of pairs that obtain this value.*

Proof. From the definition of the rank-1 lattice, we know that

$$\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_{T_p} = \left\| \frac{i\boldsymbol{z} \bmod n}{n} - \frac{j\boldsymbol{z} \bmod n}{n} \right\|_{T_p} = \left\| \frac{(i-j)\boldsymbol{z} \bmod n}{n} \right\|_{T_p} \tag{10.257}$$

$$= \left\| \frac{k\boldsymbol{z} \bmod n}{n} \right\|_{T_p} = \|\boldsymbol{x}_k\|_{T_p}, \tag{10.258}$$

where $\|\mathbf{x}\|_{T_p}$ denotes the l_p-norm-based toroidal distance between $\mathbf{x}$ and $\mathbf{0}$, and $k \equiv i - j \bmod n$.

For non-identical pair $\boldsymbol{x}_i, \boldsymbol{x}_j \in X = \{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_{n-1}\}$, we know $i \neq j$. Thus, $i - j \equiv k \in \{1, \cdots, n-1\}$. Moreover, for each k, there are n pairs of $i, j \in \{0, \cdots, n-1\}$ obtaining $i - j \equiv k \bmod n$. Therefore, the non-identical pairwise toroidal distance is determined by $\|\boldsymbol{x}_k\|_{T_p}$ for $k \in \{1, \cdots, n-1\}$. Moreover, each $\|\boldsymbol{x}_k\|_{T_p}$ corresponds to n pairwise distances.

From the definition of the l_p-norm-based toroidal distance, we know that

$$\begin{aligned} \|\boldsymbol{x}_k\|_{T_p} &= \left\| \min\left(\frac{k\boldsymbol{z} \bmod n}{n}, \frac{n - k\boldsymbol{z} \bmod n}{n} \right) \right\|_p \\ &= \left\| \min\left(\frac{k\boldsymbol{z} \bmod n}{n}, \frac{(-k\boldsymbol{z}) \bmod n}{n} \right) \right\|_p, \end{aligned} \tag{10.259}$$

where $\min(\cdot, \cdot)$ denotes the element-wise min operation between two inputs.

Since n is a prime number, from the number theory, we know that for a primitive root g, the residue of $\{g^0, g^1, \cdots, g^{n-2}\}$ modulo n forms a cyclic group under multiplication, and $g^{n-1} \equiv 1 \bmod n$. Moreover, there is a one-to-one correspondence between the residue of $\{g^0, g^1, \cdots, g^{n-2}\}$ modulo n and the set $\{1, 2, \cdots, n-1\}$. Then, we know that $\exists k', g^{k'} \equiv k \bmod n$. It follows that

$$\|\boldsymbol{x}_k\|_{T_p} = \left\| \min\left(\frac{g^{k'}\boldsymbol{z} \bmod n}{n}, \frac{(-g^{k'}\boldsymbol{z}) \bmod n}{n} \right) \right\|_p. \tag{10.260}$$

Since $(g^{\frac{n-1}{2}})^2 = g^{n-1} \equiv 1 \bmod n$ and g is a primitive root, we know that $g^{\frac{n-1}{2}} \equiv -1 \bmod n$. Denote $\{\boldsymbol{z}, -\boldsymbol{z}\} := \{z_1, z_2, \cdots, z_d, -z_1, z_2, \cdots, -z_d\}$. Since

$\boldsymbol{z} = [g^0, g^{\frac{n-1}{2d}}, g^{\frac{2(n-1)}{2d}}, \cdots, g^{\frac{(d-1)(n-1)}{2d}}] \bmod n$, we know that

$$\{\boldsymbol{z}, -\boldsymbol{z}\} \tag{10.261}$$
$$\equiv \{\boldsymbol{z}, g^{\frac{n-1}{2}}\boldsymbol{z}\} \bmod n \tag{10.262}$$
$$\equiv \{g^0, g^{\frac{n-1}{2d}}, g^{\frac{2(n-1)}{2d}}, \cdots, g^{\frac{(d-1)(n-1)}{2d}}, g^{\frac{n-1}{2}+0}, g^{\frac{n-1}{2}+\frac{n-1}{2d}}, \cdots, g^{\frac{n-1}{2}+\frac{(d-1)(n-1)}{2d}}\} \bmod n \tag{10.263}$$
$$\equiv \{g^0, g^{\frac{n-1}{2d}}, g^{\frac{2(n-1)}{2d}}, \cdots, g^{\frac{(d-1)(n-1)}{2d}}, g^{\frac{d(n-1)}{2d}}, g^{\frac{(d+1)(n-1)}{2d}}, \cdots, g^{\frac{(2d-1)(n-1)}{2d}}\} \bmod n. \tag{10.264}$$

It follows that $H := \{z_1, z_2, \cdots, z_d, -z_1, z_2, \cdots, -z_d\} \bmod n$ forms a subgroup of the group $\{g^0, g^1, \cdots, g^{n-2}\} \bmod n$. From Lagrange's theorem in group theory [53], we know that the cosets of the subgroup H partition the entire group $\{g^0, g^1, \cdots, g^{n-2}\}$ into equal-size, non-overlapping sets, i.e., cosets $g^0H, g^1H, \cdots, g^{\frac{n-1-2d}{2d}}H$, and the number of cosets of H is $\frac{n-1}{2d}$.

Together with Eq.(10.260), we know that distance $\|\boldsymbol{x}_k\|_{T_p}$ for $k' \in \{0, \cdots, n-2\}$ has $\frac{n-1}{2d}$ different values simultaneously hold for all $p \in (0, \infty)$, i.e, $\left\|\min\left(\frac{g^h\boldsymbol{z} \bmod n}{n}, \frac{(-g^h\boldsymbol{z}) \bmod n}{n}\right)\right\|_p$ for $h \in \{0, \cdots, \frac{n-1}{2d} - 1\}$. And for each distance value, there are the same number of terms $\|\boldsymbol{x}_k\|_{T_p}$ that obtain this value. Since each $\|\boldsymbol{x}_k\|_{T_p}$ corresponds to n pairwise distance $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_{T_p}$, where $k \equiv i - j \bmod n$, there are $\frac{n-1}{2d}$ distinct pairwise toroidal distance. Moreover, for each distance value, there are the same number of pairs that obtain this value.

□

10.18 Proof of Theorem 14

Theorem. *Suppose n is a prime number and $n \geq 2d+1$. Let $\boldsymbol{z} = [z_1, z_2, \cdots, z_d]$ with $1 \leq z_k \leq n-1$. Construct non-degenerate rank-1 lattice $X = \{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_{n-1}\}$ with $\boldsymbol{x}_i = \frac{i\boldsymbol{z} \bmod n}{n}, i \in \{0, ..., n-1\}$. Then, the minimum pairwise toroidal distance can be bounded as*

$$\frac{d(d+1)}{2n} \leq \min_{i,j \in \{0,\cdots,n-1\}, i \neq j} \|\mathbf{x}_i - \mathbf{x}_j\|_{T_1} \leq \frac{(n+1)d}{4n} \tag{10.265}$$
$$\frac{\sqrt{6d(d+1)(2d+1)}}{6n} \leq \min_{i,j \in \{0,\cdots,n-1\}, i \neq j} \|\mathbf{x}_i - \mathbf{x}_j\|_{T_2} \leq \sqrt{\frac{(n+1)d}{12n}}, \tag{10.266}$$

where $\|\cdot\|_{T_1}$ and $\|\cdot\|_{T_2}$ denotes the l_1-norm-based toroidal distance and the l_2-norm-based toroidal distance, respectively.

Proof. From the definition of the rank-1 lattice, we know that

$$\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_{T_p} = \left\| \frac{i\boldsymbol{z} \bmod n}{n} - \frac{j\boldsymbol{z} \bmod n}{n} \right\|_{T_p} = \left\| \frac{(i-j)\boldsymbol{z} \bmod n}{n} \right\|_{T_p} \tag{10.267}$$

$$= \left\| \frac{k\boldsymbol{z} \bmod n}{n} \right\|_{T_p} = \|\boldsymbol{x}_k\|_{T_p}, \tag{10.268}$$

where $\|\mathbf{x}\|_{T_p}$ denotes the l_p-norm-based toroidal distance, we know that between $\mathbf{x}$ and $\mathbf{0}$, and $k \equiv i - j \bmod n$.

Thus, the minimum pairwise toroidal distance is equivalent to Eq. (10.269)

$$\min_{i,j \in \{0,\cdots,n-1\}, i \neq j} \|\mathbf{x}_i - \mathbf{x}_j\|_{T_p} = \min_{k \in \{1,\cdots,n-1\}} \|\mathbf{x}_k\|_{T_p}. \tag{10.269}$$

Since the minimum value is smaller than the average value, it follows that

$$\min_{i,j \in \{0,\cdots,n-1\}, i \neq j} \|\mathbf{x}_i - \mathbf{x}_j\|_{T_p} = \min_{k \in \{1,\cdots,n-1\}} \|\mathbf{x}_k\|_{T_p} \leq \frac{\sum_{k=1}^{n-1} \|\boldsymbol{x}_k\|_{T_p}}{n-1}. \tag{10.270}$$

Since n is a prime number, from number theory, we know that for a primitive root g, the residue of $\{g^0, g^1, \cdots, g^{n-2}\}$ modulo n forms a cyclic group under multiplication, and $g^{n-1} \equiv 1 \bmod n$. Moreover, there is a one-to-one correspondence between the residue of $\{g^0, g^1, \cdots, g^{n-2}\}$ modulo n and the set $\{1, 2, \cdots, n-1\}$. Then, for each t^{th} component of $\boldsymbol{z} = [z_1, z_2, \cdots, z_d]$, we know that $\exists m_t$ such that $g^{m_t} \equiv z_t \bmod n$. Therefore, the set $\left\{ kz_t \bmod n \middle| \forall k \in \{1, \cdots, n-1\} \right\}$ is a permutation of the set $\{1, \cdots, n-1\}$.

From the definition of the l_p-norm-based toroidal distance, we know that each t^{th} component of $\|\mathbf{x}_k\|_{T_p}$ is determined by $\min(kz_t \bmod n, n - kz_t \bmod n)$. Because the set $\left\{ kz_t \bmod n \middle| \forall k \in \{1, \cdots, n-1\} \right\}$ is a permutation of set $\{1, \cdots, n-1\}$, we know that the set $\left\{ \min(kz_t \bmod n, n - kz_t \bmod n) \middle| \forall k \in \{1, \cdots, n-1\} \right\}$ consists of two copy of permutation of the set $\{1, \cdots, \frac{n-1}{2}\}$. It follows that

$$\sum_{k=1}^{n-1} \|\boldsymbol{x}_k\|_{T_1} = \frac{\sum_{t=1}^{d} \sum_{k=1}^{n-1} \min(kz_t \bmod n, n - kz_t \bmod n)}{n} = \frac{2d \sum_{k=1}^{\frac{n-1}{2}} k}{n} = \frac{d(n+1)(n-1)}{4n}. \tag{10.271}$$

Similarly, for l_2-norm-based toroidal distance, we have that

$$\sum_{k=1}^{n-1} \|\boldsymbol{x}_k\|_{T_2}^2 = \frac{\sum_{t=1}^{d} \sum_{k=1}^{n-1} \min(kz_t \bmod n, n - kz_t \bmod n)^2}{n^2} = \frac{2d \sum_{k=1}^{\frac{n-1}{2}} k^2}{n^2} = \frac{d(n-1)(n+1)}{12n}. \tag{10.272}$$

By Cauchy–Schwarz inequality, we know that

$$\sum_{k=1}^{n-1} \|\boldsymbol{x}_k\|_{T_2} \leq \sqrt{(n-1)\sum_{k=1}^{n-1} \|\boldsymbol{x}_k\|_{T_2}^2} = (n-1)\sqrt{\frac{d(n+1)}{12n}}. \tag{10.273}$$

Together with Eq.(10.270), it follows that

$$\min_{i,j\in\{0,\cdots,n-1\},i\neq j} \|\mathbf{x}_i - \mathbf{x}_j\|_{T_1} = \min_{k\in\{1,\cdots,n-1\}} \|\mathbf{x}_k\|_{T_1} \leq \frac{(n+1)d}{4n} \tag{10.274}$$

$$\min_{i,j\in\{0,\cdots,n-1\},i\neq j} \|\mathbf{x}_i - \mathbf{x}_j\|_{T_2} = \min_{k\in\{1,\cdots,n-1\}} \|\mathbf{x}_k\|_{T_2} \leq \sqrt{\frac{(n+1)d}{12n}}. \tag{10.275}$$

Now, we are going to prove the lower bound. For a non-degenerate rank-1 lattice, the components of generating vector $\boldsymbol{z} = [z_1, \cdots, z_d]$ should be all different. Then, we know the components of $\boldsymbol{x}_k, \forall k \in \{1, \cdots, n-1\}$ should be all different. Thus, the min norm point is achieved at $\boldsymbol{x}^* = [1/n, 2/n, \cdots, d/n]$. Since $n \geq 2d+1$, it follows that

$$\min_{i,j\in\{0,\cdots,n-1\},i\neq j} \|\mathbf{x}_i - \mathbf{x}_j\|_{T_1} = \min_{k\in\{1,\cdots,n-1\}} \|\mathbf{x}_k\|_{T_1} \geq \|\mathbf{x}^*\|_{T_1} = \frac{(d+1)d}{2n} \tag{10.276}$$

$$\min_{i,j\in\{0,\cdots,n-1\},i\neq j} \|\mathbf{x}_i - \mathbf{x}_j\|_{T_2} = \min_{k\in\{1,\cdots,n-1\}} \|\mathbf{x}_k\|_{T_2} \geq \|\mathbf{x}^*\|_{T_2} = \frac{\sqrt{6d(d+1)(2d+1)}}{6n}. \tag{10.277}$$

□

10.19 Proof of Corollary 1

Corollary 1. *Suppose $n = 2d+1$ is a prime number. Let g be a primitive root of n. Let $\boldsymbol{z} = [g^0, g^{\frac{n-1}{2d}}, g^{\frac{2(n-1)}{2d}}, \cdots, g^{\frac{(d-1)(n-1)}{2d}}]$ mod n. Construct rank-1 lattice $X = \{\boldsymbol{x}_0, \cdots, \boldsymbol{x}_{n-1}\}$ with $\boldsymbol{x}_i = \frac{i\boldsymbol{z} \bmod n}{n}, i \in \{0, ..., n-1\}$. Then, the pairwise toroidal distance of the lattice X attains the upper bound.*

$$\|\mathbf{x}_i - \mathbf{x}_j\|_{T_1} = \frac{(n+1)d}{4n}, \forall i, j \in \{0, \cdots, n-1\}, i \neq j, \tag{10.278}$$

$$\|\mathbf{x}_i - \mathbf{x}_j\|_{T_2} = \sqrt{\frac{(n+1)d}{12n}}, \forall i, j \in \{0, \cdots, n-1\}, i \neq j. \tag{10.279}$$

Proof. From the definition of the rank-1 lattice, we know that

$$\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_{T_p} = \left\| \frac{i\boldsymbol{z} \bmod n}{n} - \frac{j\boldsymbol{z} \bmod n}{n} \right\|_{T_p} = \left\| \frac{(i-j)\boldsymbol{z} \bmod n}{n} \right\|_{T_p} \tag{10.280}$$

$$= \left\| \frac{k\boldsymbol{z} \bmod n}{n} \right\|_{T_p} = \|\boldsymbol{x}_k\|_{T_p}, \tag{10.281}$$

where $\|\mathbf{x}\|_{T_p}$ denote the l_p-norm-based toroidal distance, we know that between $\mathbf{x}$ and $\mathbf{0}$, and $k \equiv i - j \bmod n$.

From Theorem 1, we know that $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_{T_p}\ \forall i, j \in \{0, \cdots, n-1\}, i \neq j$ has $\frac{n-1}{2d}$ different values. Since $n = 2d + 1$, we know the pairwise toroidal distance has the same value. Therefore, we know that

$$\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_{T_p} = \|\boldsymbol{x}_k\|_{T_p} = \frac{\sum_{k=1}^{n-1} \|\boldsymbol{x}_k\|_{T_p}}{n-1}, \forall i, j \in \{0, \cdots, n-1\}, i \neq j. \tag{10.282}$$

From the proof of Theorem 2, we know that

$$\sum_{k=1}^{n-1} \|\boldsymbol{x}_k\|_{T_1} = \frac{\sum_{t=1}^{d} \sum_{k=1}^{n-1} \min(kz_t \bmod n, n - kz_t \bmod n)}{n} = \frac{2d \sum_{k=1}^{\frac{n-1}{2}} k}{n} = \frac{d(n+1)(n-1)}{4n}. \tag{10.283}$$

and

$$\sum_{k=1}^{n-1} \|\boldsymbol{x}_k\|_{T_2}^2 = \frac{\sum_{t=1}^{d} \sum_{k=1}^{n-1} \min(kz_t \bmod n, n - kz_t \bmod n)^2}{n^2} = \frac{2d \sum_{k=1}^{\frac{n-1}{2}} k^2}{n^2} = \frac{d(n-1)(n+1)}{12n}. \tag{10.284}$$

Together Eq.(10.283) with Eq.(10.282), we know that

$$\|\mathbf{x}_i - \mathbf{x}_j\|_{T_1} = \frac{(n+1)d}{4n}, \forall i, j \in \{0, \cdots, n-1\}, i \neq j. \tag{10.285}$$

Since $\|\boldsymbol{x}_1\|_{T_p} = \|\boldsymbol{x}_2\|_{T_p} = \cdots = \|\boldsymbol{x}_{n-1}\|_{T_p}$, it follows that

$$\sum_{k=1}^{n-1} \|\boldsymbol{x}_k\|_{T_2} = \sqrt{(n-1) \sum_{k=1}^{n-1} \|\boldsymbol{x}_k\|_{T_2}^2}. \tag{10.286}$$

Together with Eq.(10.284), we know that

$$\sum_{k=1}^{n-1} \|\boldsymbol{x}_k\|_{T_2} = \sqrt{(n-1) \sum_{k=1}^{n-1} \|\boldsymbol{x}_k\|_{T_2}^2} = (n-1)\sqrt{\frac{d(n+1)}{12n}}. \tag{10.287}$$

Plug Eq.(10.287) into Eq.(10.282), if follows that

$$\|\mathbf{x}_i - \mathbf{x}_j\|_{T_2} = \sqrt{\frac{(n+1)d}{12n}}, \forall i, j \in \{0, \cdots, n-1\}, i \neq j. \tag{10.288}$$

From Theorem 2, we know that the l_1-norm-based and l_2-norm-based pairwise toroidal distance of the lattice X attains the upper bound.

□

10.20 Proof of Proposition 1

Kernel Property:

Proposition. *For* $\forall f \in \mathcal{F}$ *(* $\mathcal{F} = \mathcal{L}_2$ *or* $\mathcal{F} = \overline{\mathcal{L}}_2$*), define function* $k(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w}, \boldsymbol{x}) f(\boldsymbol{w}, \boldsymbol{y})] : \mathcal{X} \times \mathcal{X} \to \mathcal{R}$*, then* $k(\boldsymbol{x}, \boldsymbol{y})$ *is a bounded kernel, i.e.,* $k(\boldsymbol{x}, \boldsymbol{y}) = k(\boldsymbol{y}, \boldsymbol{x}) < \infty$ *and* $k(\boldsymbol{x}, \boldsymbol{y})$ *is positive definite.*

Proof. (i) Symmetric property is straightforward by definition.

(ii) From Cauchy–Schwarz inequality,

$$k(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w}, \boldsymbol{x}) f(\boldsymbol{w}, \boldsymbol{y})] \leq \sqrt{\mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w}, \boldsymbol{x})^2] \mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w}, \boldsymbol{y})^2]} < \infty \tag{10.289}$$

(iii) Positive definite property. For $\forall n \in \mathbb{N}$, $\forall \alpha_1 \cdots, \alpha_n \in \mathcal{R}$ and $\forall \boldsymbol{x}_1, \cdots, \boldsymbol{x}_n \in \mathcal{X}$, we have

$$\sum_i \sum_j \alpha_i \alpha_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \mathbb{E}_{\boldsymbol{w}}\big[\big(\sum_i \alpha_i f(\boldsymbol{w}, \boldsymbol{x}_i)\big)^2\big] \geq 0$$

□

10.21 Proof of Theorem 22

Convex ϕ-regularization:

$$\min_{f \in \mathcal{F}} \frac{1}{2} \|\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f(\boldsymbol{w})]\|_2^2 + \mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f(\boldsymbol{w}))] \tag{10.290}$$

where $\mathcal{F} = \mathcal{L}_2$ or $\mathcal{F} = \overline{\mathcal{L}}_2$. And $\mathcal{L}_2$ denotes the Gaussian square integrable functional space, i.e., $\mathcal{L}_2 := \{f \big| \mathbb{E}_{\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)}[f(\boldsymbol{w})^2] < \infty\}$, $\overline{\mathcal{L}}_2$ denotes the sphere square integrable functional space, i.e., $\overline{\mathcal{L}}_2 := \{f \big| \mathbb{E}_{\boldsymbol{w} \sim Uni[\sqrt{d}\mathbb{S}^{d-1}]}[f(\boldsymbol{w})^2] < \infty\}$ and $\phi_\lambda(\cdot)$ denotes a convex function bounded from below.

Lemma 9. $\mathbb{E}_{\boldsymbol{w} \sim Uni[\sqrt{d}\mathbb{S}^{d-1}]}[\boldsymbol{w}\boldsymbol{w}^\top] = \boldsymbol{I}_d$.

Proof.

$$
\begin{aligned}
\boldsymbol{I}_d &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I}_d)}[\boldsymbol{x}\boldsymbol{x}^\top] \\
&= \int \frac{1}{(2\pi)^{\frac{d}{2}}} e^{-\frac{\|\boldsymbol{x}\|_2^2}{2}} \boldsymbol{x}\boldsymbol{x}^\top \, d\boldsymbol{x} \\
&= \int_0^\infty \int_{\mathbb{S}^{d-1}} \frac{2\pi^{\frac{d}{2}}}{\Gamma(\frac{d}{2})} r^{d-1} \cdot e^{-\frac{r^2}{2}} r^2 \cdot \frac{1}{(2\pi)^{\frac{d}{2}}} \boldsymbol{v}\boldsymbol{v}^\top \, d\sigma(\boldsymbol{v}) \, dr && (10.291) \\
&= \int_0^\infty \frac{2\pi^{\frac{d}{2}}}{\Gamma(\frac{d}{2})} r^{d-1} e^{-\frac{r^2}{2}} r^2 \cdot \frac{1}{(2\pi)^{\frac{d}{2}}} \, dr \int_{\mathbb{S}^{d-1}} \boldsymbol{v}\boldsymbol{v}^\top \, d\sigma(\boldsymbol{v}) && (10.292) \\
&= \int_0^\infty \frac{r^{d-1} e^{-\frac{r^2}{2}}}{2^{\frac{d}{2}-1}\Gamma(\frac{d}{2})} r^2 \, dr \int_{\mathbb{S}^{d-1}} \boldsymbol{v}\boldsymbol{v}^\top \, d\sigma(\boldsymbol{v}) && (10.293) \\
&= \mathbb{E}_{r\sim\chi(d)}[r^2] \int_{\mathbb{S}^{d-1}} \boldsymbol{v}\boldsymbol{v}^\top \, d\sigma(\boldsymbol{v}) && (10.294) \\
&= d \int_{\mathbb{S}^{d-1}} \boldsymbol{v}\boldsymbol{v}^\top \, d\sigma(\boldsymbol{v}) && (10.295) \\
&= \mathbb{E}_{\boldsymbol{w}\sim Uni[\sqrt{d}\mathbb{S}^{d-1}]}[\boldsymbol{w}\boldsymbol{w}^\top] && (10.296)
\end{aligned}
$$

where $\sigma(\cdot)$ denotes the normalized surface measure, $\chi(d)$ denotes the Chi distribution with degree d, $\Gamma(\cdot)$ denotes the gamma function. □

Lemma 10. *Let $f \in \mathcal{F}$ with $\mathcal{F} = \mathcal{L}_2$ or $\mathcal{F} = \overline{\mathcal{L}}_2$, then we have*

$$
\mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w})^2] - \|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2 = \mathbb{E}_{\boldsymbol{w}}[\big(f(\boldsymbol{w}) - \boldsymbol{w}^\top \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\big)^2] \geq 0 \tag{10.297}
$$

Proof. Let w_i denote the i^{th} component of $\boldsymbol{w}$, from Cauchy–Schwarz inequality, we know that

$$
(\mathbb{E}_{\boldsymbol{w}}[w_i f(\boldsymbol{w})])^2 \leq \mathbb{E}_{\boldsymbol{w}}[w_i^2]\mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w})^2] = \mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w})^2] < \infty \tag{10.298}
$$

Thus the expectation $\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]$ exits.

Since $\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\boldsymbol{w}^\top] = \boldsymbol{I}_d$, we have

$$
\begin{aligned}
\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2 &= (\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})])^\top (\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]) && (10.299) \\
&= (\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})])^\top \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\boldsymbol{w}^\top](\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]) && (10.300) \\
&= \mathbb{E}_{\boldsymbol{w}}[(\boldsymbol{w}^\top \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})])^2] && (10.301)
\end{aligned}
$$

It follows that

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w})^2] - \|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2 \\
&= \mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w})^2] - 2\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2 + \|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2 && (10.302)\\
&= \mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w})^2] - 2(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})])^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]) + \mathbb{E}_{\boldsymbol{w}}[(\boldsymbol{w}^\top\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})])^2] && (10.303)\\
&= \mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w})^2] - 2\mathbb{E}_{\boldsymbol{w}}[f(\boldsymbol{w})\boldsymbol{w}^\top\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]] + \mathbb{E}_{\boldsymbol{w}}[(\boldsymbol{w}^\top\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})])^2] && (10.304)\\
&= \mathbb{E}_{\boldsymbol{w}}[\big(f(\boldsymbol{w}) - \boldsymbol{w}^\top\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\big)^2] \geq 0 && (10.305)
\end{aligned}$$

□

Lemma 11. *Let $f \in \mathcal{F}$ with $\mathcal{F} = \mathcal{L}_2$ or $\mathcal{F} = \overline{\mathcal{L}}_2$, we have*

$$\|\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2 = \mathbb{E}_{\boldsymbol{w}}[(\boldsymbol{w}^\top(\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]))^2] \tag{10.306}$$

Proof. Since $\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\boldsymbol{w}^\top] = \boldsymbol{I}_d$, we have

$$\begin{aligned}
\|\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2 &= (\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})])^\top(\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]) && (10.307)\\
&= (\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})])^\top\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\boldsymbol{w}^\top](\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]) && (10.308)\\
&= \mathbb{E}_{\boldsymbol{w}}[(\boldsymbol{w}^\top(\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]))^2] && (10.309)
\end{aligned}$$

□

Lemma 12. *Denote $L(f) := \frac{1}{2}\|\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2$. For $\forall f, g \in \mathcal{F}$ with $\mathcal{F} = \mathcal{L}_2$ or $\mathcal{F} = \overline{\mathcal{L}}_2$, we have $L(f) = L(g) + \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})] - \boldsymbol{x})\big(f(\boldsymbol{w}) - g(\boldsymbol{w})\big)] + \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f(\boldsymbol{w}) - g(\boldsymbol{w})\big)]\|_2^2$.*

Proof.

$$\begin{aligned}
\frac{1}{2}\|\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2 &= \frac{1}{2}\|\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})] + \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})] - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2 && (10.310)\\
&= \frac{1}{2}\|\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})]\|_2^2 + \frac{1}{2}\big\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})] - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\big\|_2^2 \\
&\quad + \langle \boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})], \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})] - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\rangle && (10.311)
\end{aligned}$$

The inner product term can be rewritten as

$$\begin{aligned}
\langle \boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})], \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})] - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\rangle &= (\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})])^\top\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(g(\boldsymbol{w}) - f(\boldsymbol{w})\big)] && (10.312)\\
&= \mathbb{E}_{\boldsymbol{w}}[(\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})])^\top\boldsymbol{w}\big(g(\boldsymbol{w}) - f(\boldsymbol{w})\big)] && (10.313)
\end{aligned}$$

It follows that

$$\frac{1}{2}\|\boldsymbol{x}-\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2=\frac{1}{2}\|\boldsymbol{x}-\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})]\|_2^2+\frac{1}{2}\big\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(g(\boldsymbol{w})-f(\boldsymbol{w})\big)]\big\|_2^2 + \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}g(\boldsymbol{w})]-\boldsymbol{x})\big(f(\boldsymbol{w})-g(\boldsymbol{w})\big)] \tag{10.314}$$

□

Lemma 13. *For $\forall f_t, f_{t+1}, f^* \in \mathcal{F}$ with $\mathcal{F}=\mathcal{L}_2$ or $\mathcal{F}=\overline{\mathcal{L}}_2$, we have*

$$L(f_{t+1})=L(f^*)+\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f_t(\boldsymbol{w})]-\boldsymbol{x})\big(f_{t+1}(\boldsymbol{w})-f^*(\boldsymbol{w})\big)]+\frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w})-f_t(\boldsymbol{w})\big)]\|_2^2 - \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w})-f^*(\boldsymbol{w})\big)]\|_2^2 \tag{10.315}$$

Proof. From Lemma 12, we know that

$$L(f_{t+1}) = L(f_t) + \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f_t(\boldsymbol{w})]-\boldsymbol{x})\big(f_{t+1}(\boldsymbol{w})-f_t(\boldsymbol{w})\big)]+\frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w})-f_t(\boldsymbol{w})\big)]\|_2^2 \tag{10.316}$$

$$L(f^*) = L(f_t) + \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f_t(\boldsymbol{w})]-\boldsymbol{x})\big(f^*(\boldsymbol{w})-f_t(\boldsymbol{w})\big)]+\frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f^*(\boldsymbol{w})-f_t(\boldsymbol{w})\big)]\|_2^2 \tag{10.317}$$

Plug $L(f_t)$ into Eq.(10.316), we can obtain that

$$L(f_{t+1})=L(f^*)-\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f_t(\boldsymbol{w})]-\boldsymbol{x})\big(f^*(\boldsymbol{w})-f_t(\boldsymbol{w})\big)]-\frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f^*(\boldsymbol{w})-f_t(\boldsymbol{w})\big)]\|_2^2 + \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f_t(\boldsymbol{w})]-\boldsymbol{x})\big(f_{t+1}(\boldsymbol{w})-f_t(\boldsymbol{w})\big)]+\frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w})-f_t(\boldsymbol{w})\big)]\|_2^2 \tag{10.318}$$

$$=L(f^*)+\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f_t(\boldsymbol{w})]-\boldsymbol{x})\big(f_{t+1}(\boldsymbol{w})-f^*(\boldsymbol{w})\big)]+\frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w})-f_t(\boldsymbol{w})\big)]\|_2^2 - \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w})-f^*(\boldsymbol{w})\big)]\|_2^2 \tag{10.319}$$

□

Lemma 14. *For a convex function $\phi_\lambda(\cdot)$, denote $h(\cdot)$ as the proximal operator of $\phi_\lambda(\cdot)$, i.e., $h(z)=\arg\min_x \frac{1}{2}(x-z)^2+\phi_\lambda(x)$, let $f_{t+1}=h\circ g_{t+1}\in\mathcal{F}$ with $\mathcal{F}=\mathcal{L}_2$ or $\mathcal{F}=\overline{\mathcal{L}}_2$, then for $\forall f^*\in\mathcal{F}$, we have*

$$\mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f_{t+1}(\boldsymbol{w}))]\le\mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f^*(\boldsymbol{w}))]-\mathbb{E}_{\boldsymbol{w}}[\big(g_{t+1}(\boldsymbol{w})-f_{t+1}(\boldsymbol{w})\big)\big(f^*(\boldsymbol{w})-f_{t+1}(\boldsymbol{w})\big)] \tag{10.320}$$

Proof. Since $\phi_\lambda(\cdot)$ is convex function and $f_{t+1}(\boldsymbol{w})=\arg\min_x \phi_\lambda(x)+\frac{1}{2}\|x-g_{t+1}(\boldsymbol{w})\|_2^2$, we have

$$0\in\partial\phi_\lambda(f_{t+1}(\boldsymbol{w}))+(f_{t+1}(\boldsymbol{w})-g_{t+1}(\boldsymbol{w}))\implies(g_{t+1}(\boldsymbol{w})-f_{t+1}(\boldsymbol{w}))\in\partial\phi_\lambda(f_{t+1}(\boldsymbol{w})) \tag{10.321}$$

From the definition of subgradient and convex function $\phi_\lambda(\cdot)$, we have

$$\phi_\lambda(f_{t+1}(\boldsymbol{w})) \leq \phi_\lambda(f^*(\boldsymbol{w})) - (g_{t+1}(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w}))(f^*(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w})) \tag{10.322}$$

It follows that

$$\mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f_{t+1}(\boldsymbol{w}))] \leq \mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f^*(\boldsymbol{w}))] - \mathbb{E}_{\boldsymbol{w}}[\big(g_{t+1}(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w})\big)\big(f^*(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w})\big)] \tag{10.323}$$

□

Lemma 15. *Denote $h(\cdot)$ as the proximal operator of $\phi_\lambda(\cdot)$. Suppose $|h(x)| \leq c|x|$ (or $|h(x)| \leq c$) , $0 < c < \infty$. Given a bouned $\boldsymbol{x} \in \mathcal{R}^d$, set function $g_{t+1}(\boldsymbol{w}) = \boldsymbol{w}^\top \boldsymbol{x} + f_t(\boldsymbol{w}) - \boldsymbol{w}^\top \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f_t(\boldsymbol{w})]$ with $f_t \in \mathcal{L}_2$ and $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ (or $f_t \in \overline{\mathcal{L}}_2$ and $\boldsymbol{w} \sim Uni[\sqrt{d}\mathbb{S}^{d-1}]$). Set $f_{t+1} = h \circ g_{t+1}$, then, we know $f_{t+1} \in \mathcal{F}$ with $\mathcal{F} = \mathcal{L}_2$ or $\mathcal{F} = \overline{\mathcal{L}}_2$,respectively.*

Proof. **Case** $|h(x)| \leq c,\ 0 < c < \infty$**:** It is straightforward to know $\mathbb{E}_{\boldsymbol{w}}[h(g_{t+1}(\boldsymbol{w}))^2] \leq c^2 < \infty$, thus $f_{t+1} \in \mathcal{F}$.

Case $|h(x)| \leq c|x|,\ 0 < c < \infty$**:** Since $|h(x)| \leq c|x|$, we know that

$$h(g_{t+1}(\boldsymbol{w}))^2 \leq c^2 g_{t+1}(\boldsymbol{w})^2 = c^2\big(\boldsymbol{w}^\top \boldsymbol{x} + f_t(\boldsymbol{w}) - \boldsymbol{w}^\top \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f_t(\boldsymbol{w})]\big)^2 \tag{10.324}$$

$$\leq 2c^2(\boldsymbol{w}^\top(\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f_t(\boldsymbol{w})]))^2 + 2c^2 f_t(\boldsymbol{w})^2 \tag{10.325}$$

It follows that

$$\mathbb{E}_{\boldsymbol{w}}[h(g_{t+1}(\boldsymbol{w}))^2] \leq c^2 \mathbb{E}_{\boldsymbol{w}}[\big(\boldsymbol{w}^\top \boldsymbol{x} + f_t(\boldsymbol{w}) - \boldsymbol{w}^\top \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f_t(\boldsymbol{w})]\big)^2] \tag{10.326}$$

$$\leq 2c^2 \mathbb{E}_{\boldsymbol{w}}[(\boldsymbol{w}^\top(\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f_t(\boldsymbol{w})]))^2] + 2c^2 \mathbb{E}_{\boldsymbol{w}}[f_t(\boldsymbol{w})^2] \tag{10.327}$$

$$= 2c^2 \|\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f_t(\boldsymbol{w})]\|_2^2 + 2c^2 \mathbb{E}_{\boldsymbol{w}}[f_t(\boldsymbol{w})^2] \tag{10.328}$$

$$\leq 4c^2 \|\boldsymbol{x}\|_2^2 + 4c^2 \|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f_t(\boldsymbol{w})]\|_2^2 + 2c^2 \mathbb{E}_{\boldsymbol{w}}[f_t(\boldsymbol{w})^2] \tag{10.329}$$

From Lemma 10, we know $\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f_t(\boldsymbol{w})]\|_2^2 \leq \mathbb{E}_{\boldsymbol{w}}[f_t(\boldsymbol{w})^2]$ is bounded, together with $\|\boldsymbol{x}\|_2 < \infty$,it follows that $\mathbb{E}_{\boldsymbol{w}}[f_{t+1}(\boldsymbol{w}))^2] = \mathbb{E}_{\boldsymbol{w}}[h(g_{t+1}(\boldsymbol{w}))^2] < \infty$. Thus, $f_{t+1} \in \mathcal{F}$.

□

Lemma 16. *For a convex function $\phi_\lambda(\cdot)$, denote $h(\cdot)$ as the proximal operator of $\phi_\lambda(\cdot)$, i.e., $h(z) = \arg\min_x \frac{1}{2}(x - z)^2 + \phi_\lambda(x)$. Suppose $|h(x)| \leq c|x|$ (or $|h(x)| \leq c$), $0 < c < \infty$ (e.g., soft thresholding function). Given a bouned $\boldsymbol{x} \in \mathcal{R}^d$, set function $g_{t+1}(\boldsymbol{w}) = \boldsymbol{w}^\top \boldsymbol{x} + f_t(\boldsymbol{w}) - \boldsymbol{w}^\top \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f_t(\boldsymbol{w})]$ with $f_t \in \mathcal{L}_2$ and $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ (or $f_t \in \overline{\mathcal{L}}_2$*

and $\boldsymbol{w} \sim Uni[\sqrt{d}\mathbb{S}^{d-1}]$*). Set* $f_{t+1} = h \circ g_{t+1}$*. Denote* $Q(f) = L(f) + \mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f(\boldsymbol{w}))]$ *with* $L(f) := \frac{1}{2}\|\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2$*, for* $\forall f^* \in \mathcal{F}$ *with* $\mathcal{F} = \mathcal{L}_2$ *or* $\mathcal{F} = \overline{\mathcal{L}}_2$*, we have*

$$Q(f_{t+1}) \leq Q(f^*) + \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w}))^2] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f^*(\boldsymbol{w}))^2] - \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)]\|_2^2 \tag{10.330}$$

Proof. From Lemma 13, we know that

$$L(f_{t+1}) = L(f^*) + \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f_t(\boldsymbol{w})] - \boldsymbol{x})\big(f_{t+1}(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)] + \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)]\|_2^2 - \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)]\|_2^2 \tag{10.331}$$

Together with Lemma 14, it follows that

$$\begin{aligned}
&Q(f_{t+1}) && (10.332)\\
&\leq Q(f^*) - \mathbb{E}_{\boldsymbol{w}}[\big(g_{t+1}(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w})\big)\big(f^*(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w})\big)] + \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f_t(\boldsymbol{w})] - \boldsymbol{x})\big(f_{t+1}(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)]\\
&+ \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)]\|_2^2 - \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)]\|_2^2 && (10.333)\\
&= Q(f^*) + \mathbb{E}_{\boldsymbol{w}}[\big(g_{t+1}(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w}) + \boldsymbol{w}^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f_t(\boldsymbol{w})] - \boldsymbol{x})\big)\big(f_{t+1}(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)]\\
&+ \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)]\|_2^2 - \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)]\|_2^2 && (10.334)\\
&= Q(f^*) + \mathbb{E}_{\boldsymbol{w}}[\big(f_t(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w})\big)\big(f_{t+1}(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)]\\
&+ \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)]\|_2^2 - \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)]\|_2^2 && (10.335)
\end{aligned}$$

Note that

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{w}}[\big(f_t(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w})\big)\big(f_{t+1}(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)]\\
&= \mathbb{E}_{\boldsymbol{w}}[\big(f_t(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w})\big)\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}) + f_t(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)] && (10.336)\\
&= \mathbb{E}_{\boldsymbol{w}}[\big(f_t(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w})\big)\big(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)] - \mathbb{E}_{\boldsymbol{w}}[\big(f_t(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w})\big)^2] && (10.337)
\end{aligned}$$

Also note that $ab = \frac{a^2 + b^2 - (a-b)^2}{2}$, it follows that

$$\big(f_t(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w})\big)\big(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w})\big) = \frac{(f_t(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w}))^2 + (f_t(\boldsymbol{w}) - f^*(\boldsymbol{w}))^2 - (f_{t+1}(\boldsymbol{w}) - f^*(\boldsymbol{w}))^2}{2} \tag{10.338}$$

It follows that

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{w}}[\big(f_t(\boldsymbol{w}) - f_{t+1}(\boldsymbol{w})\big)\big(f_{t+1}(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)]\\
&= \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w}))^2] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f^*(\boldsymbol{w}))^2] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}))^2]
\end{aligned} \tag{10.339}$$

Plug Eq.(10.339) into Eq.(10.335), we can obtain that

$$Q(f_{t+1}) \leq Q(f^*) + \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w}))^2] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f^*(\boldsymbol{w}))^2] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}))^2] + \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)]\|_2^2 - \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)]\|_2^2 \tag{10.340}$$

From Lemma 10, we know $\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)]\|_2^2 \leq \mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}))^2]$. It follows that

$$Q(f_{t+1}) \leq Q(f^*) + \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w}))^2] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f^*(\boldsymbol{w}))^2] - \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w}) - f^*(\boldsymbol{w})\big)]\|_2^2 \tag{10.341}$$

□

Lemma 17. *(Strictly Monotonic Descent (a.s.)) Following the same condition of Lemma 16, we have*

$$Q(f_{t+1}) \leq Q(f_t) - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}))^2] \tag{10.342}$$

Proof. It follows directly from Lemma 16 by setting $f^* = f_t$. □

Theorem. *For a convex function $\phi_\lambda(\cdot)$, denote $h(\cdot)$ as the proximal operator of $\phi_\lambda(\cdot)$, i.e., $h(z) = \arg\min_x \frac{1}{2}(x-z)^2 + \phi_\lambda(x)$. Suppose $|h(x)| \leq c|x|$ (or $|h(x)| \leq c$), $0 < c < \infty$. Given a bouned $\boldsymbol{x} \in \mathcal{R}^d$, set function $g_{t+1}(\boldsymbol{w}) = \boldsymbol{w}^\top \boldsymbol{x} + f_t(\boldsymbol{w}) - \boldsymbol{w}^\top \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f_t(\boldsymbol{w})]$ with $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$ (or $\boldsymbol{w} \sim Uni[\sqrt{d}\mathbb{S}^{d-1}]$). Set $f_{t+1} = h \circ g_{t+1}$ and $f_0 \in \mathcal{F}$ with $\mathcal{F} = \mathcal{L}_2$ or $\mathcal{F} = \overline{\mathcal{L}}_2$ (e.g., $f_0 = 0$). Denote $Q(f) = L(f) + \mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f(\boldsymbol{w}))]$ with $L(f) := \frac{1}{2}\|\boldsymbol{x} - \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w} f(\boldsymbol{w})]\|_2^2$. Denote $f_* \in \mathcal{F}$ as an optimal of $Q(\cdot)$, we have*

$$T\big(Q(f_T) - Q(f_*)\big) \leq \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_0(\boldsymbol{w}) - f_*(\boldsymbol{w}))^2] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_T(\boldsymbol{w}) - f_*(\boldsymbol{w}))^2] - \frac{1}{2}\sum_{t=0}^{T-1}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w}) - f_*(\boldsymbol{w})\big)]\|_2^2 - \frac{1}{2}\sum_{t=0}^{T-1}(t+1)\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}))^2] \tag{10.343}$$

Proof. From Lemma 16, by setting $f^* = f_*$, we can obtain that

$$Q(f_{t+1}) \leq Q(f_*) + \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_t(\boldsymbol{w}) - f_*(\boldsymbol{w}))^2] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f_*(\boldsymbol{w}))^2] - \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w}) - f_*(\boldsymbol{w})\big)]\|_2^2 \tag{10.344}$$

Telescope the inequality (10.344) from $t=0$ to $t=T-1$, we can obtain that

$$\sum_{t=0}^{T-1} Q(f_{t+1}) - TQ(f_*) \leq \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_0(\boldsymbol{w})-f_*(\boldsymbol{w}))^2] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_T(\boldsymbol{w})-f_*(\boldsymbol{w}))^2] - \frac{1}{2}\sum_{t=0}^{T-1}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w})-f_*(\boldsymbol{w})\big)]\|_2^2 \tag{10.345}$$

In addition, from Lemma 17, we can obtain that

$$Q(f_T) \leq Q(f_t) - \frac{1}{2}\sum_{i=t}^{T-1}\mathbb{E}_{\boldsymbol{w}}[(f_{i+1}(\boldsymbol{w})-f_i(\boldsymbol{w}))^2] \tag{10.346}$$

It follows that

$$TQ(f_T) - TQ(f_*) \leq \sum_{t=0}^{T-1} Q(f_{t+1}) - TQ(f_*) - \frac{1}{2}\sum_{t=0}^{T-1}\sum_{i=t}^{T-1}\mathbb{E}_{\boldsymbol{w}}[(f_{i+1}(\boldsymbol{w})-f_i(\boldsymbol{w}))^2] \tag{10.347}$$

$$= \sum_{t=0}^{T-1} Q(f_{t+1}) - TQ(f_*) - \frac{1}{2}\sum_{t=0}^{T-1}(t+1)\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w})-f_t(\boldsymbol{w}))^2] \tag{10.348}$$

Plug inequality (10.345) into inequality (10.348), we can obtain that

$$TQ(f_T) - TQ(f_*) \leq \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_0(\boldsymbol{w})-f_*(\boldsymbol{w}))^2] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_T(\boldsymbol{w})-f_*(\boldsymbol{w}))^2] - \frac{1}{2}\sum_{t=0}^{T-1}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_t(\boldsymbol{w})-f_*(\boldsymbol{w})\big)]\|_2^2 - \frac{1}{2}\sum_{t=0}^{T-1}(t+1)\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w})-f_t(\boldsymbol{w}))^2] \tag{10.349}$$

□

10.22 Proof of Theorem 21

Non-convex ϕ-regularization:

Theorem. *For a (non-convex) regularization function $\phi_\lambda(\cdot)$, denote $h(\cdot)$ as the proximal operator of $\phi_\lambda(\cdot)$, i.e., $h(z)=\arg\min_x \frac{1}{2}(x-z)^2+\phi_\lambda(x)$. Suppose $|h(x)|\leq c|x|$ (or $|h(x)|\leq c$), $0<c<\infty$ (e.g., hard thresholding function). Given a bouned $\boldsymbol{x}\in\mathcal{R}^d$, set function $g_{t+1}(\boldsymbol{w})=\boldsymbol{w}^\top\boldsymbol{x}+f_t(\boldsymbol{w})-\boldsymbol{w}^\top\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f_t(\boldsymbol{w})]$ with $f_t\in\mathcal{L}_2$ and $\boldsymbol{w}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I}_d)$ (or $f_t\in\overline{\mathcal{L}}_2$, $\boldsymbol{w}\sim Uni[\sqrt{d}\mathbb{S}^{d-1}]$). Set $f_{t+1}=h\circ g_{t+1}$. Denote $Q(f)=L(f)+\mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f(\boldsymbol{w}))]$ with $L(f):=\frac{1}{2}\|\boldsymbol{x}-\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f(\boldsymbol{w})]\|_2^2$, we have*

$$Q(f_{t+1}) \leq Q(f_t) - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w})-f_t(\boldsymbol{w})-\boldsymbol{w}^\top\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}(f_{t+1}(\boldsymbol{w})-f_t(\boldsymbol{w}))])^2] \leq Q(f_t) \tag{10.350}$$

Proof. From Lemma 12, we know that

$$L(f_{t+1}) = L(f_t) + \mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}^\top(\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f_t(\boldsymbol{w})] - \boldsymbol{x})\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)] + \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)]\|_2^2 \tag{10.351}$$

Let $g_{t+1}(\boldsymbol{w}) = \boldsymbol{w}^\top\boldsymbol{x} + f_t(\boldsymbol{w}) - \boldsymbol{w}^\top\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}f_t(\boldsymbol{w})]$, together with Eq.(10.351), we can obtain that

$$L(f_{t+1}) = L(f_t) + \mathbb{E}_{\boldsymbol{w}}[\big(f_t(\boldsymbol{w}) - g_{t+1}(\boldsymbol{w})\big)\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)] + \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)]\|_2^2 \tag{10.352}$$

Note that $ab = \frac{(a+b)^2 - a^2 - b^2}{2}$, it follows that

$$\big(f_t(\boldsymbol{w}) - g_{t+1}(\boldsymbol{w})\big)\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big) = \frac{\big(f_{t+1}(\boldsymbol{w}) - g_{t+1}(\boldsymbol{w})\big)^2 - \big(f_t(\boldsymbol{w}) - g_{t+1}(\boldsymbol{w})\big)^2 - \big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)^2}{2} \tag{10.353}$$

Since $f_{t+1} = h \circ g_{t+1}$ is the solution of the proximal problem,
i.e., $f_{t+1}(\boldsymbol{w}) = \arg\min_x \frac{(x - g_{t+1}(\boldsymbol{w}))^2}{2} + \phi_\lambda(x)$, we know that

$$\frac{\big(f_{t+1}(\boldsymbol{w}) - g_{t+1}(\boldsymbol{w})\big)^2}{2} - \frac{\big(f_t(\boldsymbol{w}) - g_{t+1}(\boldsymbol{w})\big)^2}{2} \leq \phi_\lambda(f_t(\boldsymbol{w})) - \phi_\lambda(f_{t+1}(\boldsymbol{w})) \tag{10.354}$$

It follows that

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{w}}[\big(f_t(\boldsymbol{w}) - g_{t+1}(\boldsymbol{w})\big)\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)] \\
&= \mathbb{E}_{\boldsymbol{w}}[\frac{\big(f_{t+1}(\boldsymbol{w}) - g_{t+1}(\boldsymbol{w})\big)^2 - \big(f_t(\boldsymbol{w}) - g_{t+1}(\boldsymbol{w})\big)^2 - \big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)^2}{2}] && (10.355) \\
&\leq \mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f_t(\boldsymbol{w}))] - \mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f_{t+1}(\boldsymbol{w}))] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)^2] && (10.356)
\end{aligned}$$

Plug inequality (10.356) into Eq.(10.352), we can achieve that

$$\begin{aligned}
L(f_{t+1}) + \mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f_{t+1}(\boldsymbol{w}))] \leq L(f_t) + \mathbb{E}_{\boldsymbol{w}}[\phi_\lambda(f_t(\boldsymbol{w}))] - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)^2] \\
+ \frac{1}{2}\|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)]\|_2^2
\end{aligned} \tag{10.357}$$

From Lemma 10, we know that

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{w}}[\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)^2] - \|\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})\big)]\|_2^2 \\
&= \mathbb{E}_{\boldsymbol{w}}[\big(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}) - \boldsymbol{w}^\top\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}))]\big)^2]
\end{aligned} \tag{10.358}$$

It follows that

$$Q(f_{t+1}) \leq Q(f_t) - \frac{1}{2}\mathbb{E}_{\boldsymbol{w}}[(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}) - \boldsymbol{w}^\top\mathbb{E}_{\boldsymbol{w}}[\boldsymbol{w}(f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w}))])^2] \leq Q(f_t) \tag{10.359}$$

□

10.23 Proof of Theorem 23

To prove the Theorem 23, we first show some useful Lemmas.

Lemma 18. *Suppose* $\frac{1}{N}\boldsymbol{W}\boldsymbol{W}^\top = \boldsymbol{I}_d$, *for any bounded* $\boldsymbol{y} \in \mathcal{R}^N$, *we have* $\frac{1}{N}\|\boldsymbol{y}\|_2^2 - \|\frac{1}{N}\boldsymbol{W}\boldsymbol{y}\|_2^2 = \frac{1}{N}\|\boldsymbol{y} - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y}\|_2^2 \geq 0$.

Proof.

$$\frac{1}{N}\|\boldsymbol{y}\|_2^2 - \|\frac{1}{N}\boldsymbol{W}\boldsymbol{y}\|_2^2 = \|\boldsymbol{y}\|_2^2 - 2\|\frac{1}{N}\boldsymbol{W}\boldsymbol{y}\|_2^2 + \|\frac{1}{N}\boldsymbol{W}\boldsymbol{y}\|_2^2 \tag{10.360}$$

$$= \frac{1}{N}\|\boldsymbol{y}\|_2^2 - \frac{2}{N^2}\boldsymbol{y}^\top\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y} + \frac{1}{N^2}\boldsymbol{y}^\top\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y} \tag{10.361}$$

$$= \frac{1}{N}\|\boldsymbol{y}\|_2^2 - \frac{2}{N^2}\boldsymbol{y}^\top\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y} + \frac{1}{N^2}\boldsymbol{y}^\top\boldsymbol{W}^\top\frac{1}{N}\boldsymbol{W}\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y} \tag{10.362}$$

$$= \frac{1}{N}\|\boldsymbol{y} - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y}\|_2^2 \geq 0 \tag{10.363}$$

□

Lemma 19. *Denote* $L(\boldsymbol{y}) := \frac{1}{2}\|\boldsymbol{x} - \frac{1}{N}\boldsymbol{W}\boldsymbol{y}\|_2^2$. *For* $\forall \boldsymbol{y}, \boldsymbol{z} \in \mathcal{R}^N$, *we have* $L(\boldsymbol{z}) = L(\boldsymbol{y}) + < \frac{1}{N^2}\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y} - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{x}, \boldsymbol{z} - \boldsymbol{y} > + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y})\|_2^2$

Proof.

$$\frac{1}{2}\|\boldsymbol{x} - \frac{1}{N}\boldsymbol{W}\boldsymbol{z}\|_2^2 = \frac{1}{2}\|\boldsymbol{x} - \frac{1}{N}\boldsymbol{W}\boldsymbol{y} + \frac{1}{N}\boldsymbol{W}\boldsymbol{y} - \frac{1}{N}\boldsymbol{W}\boldsymbol{z}\|_2^2 \tag{10.364}$$

$$= L(\boldsymbol{y}) + < \frac{1}{N}\boldsymbol{W}\boldsymbol{y} - \boldsymbol{x}, \frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y}) > + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y})\|_2^2 \tag{10.365}$$

$$= L(\boldsymbol{y}) + < \frac{1}{N^2}\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y} - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{x}, \boldsymbol{z} - \boldsymbol{y} > + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y})\|_2^2 \tag{10.366}$$

□

Theorem. *(Monotonic Descent) For a function* $\phi_\lambda(\cdot)$, *denote* $h(\cdot)$ *as the proximal operator of* $\phi_\lambda(\cdot)$. *Given a bouned* $\boldsymbol{x} \in \mathcal{R}^d$, *set* $\boldsymbol{y}_{t+1} = h\big(\boldsymbol{W}^\top\boldsymbol{x} + (\boldsymbol{I} - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{W})\boldsymbol{y}_t\big)$ *with* $\frac{1}{N}\boldsymbol{W}\boldsymbol{W}^\top = \boldsymbol{I}_d$. *Denote* $\widehat{Q}(\boldsymbol{y}) := \frac{1}{2}\|\boldsymbol{x} - \frac{1}{N}\boldsymbol{W}\boldsymbol{y}\|_2^2 + \frac{1}{N}\phi_\lambda(\boldsymbol{y})$. *For* $t \geq 0$, *we have*

$$\widehat{Q}(\boldsymbol{y}_{t+1}) \leq \widehat{Q}(\boldsymbol{y}_t) - \frac{1}{2N}\|(\boldsymbol{I}_d - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{W})(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \leq \widehat{Q}(\boldsymbol{y}_t) \tag{10.367}$$

Proof. Denote $L(\boldsymbol{y}) := \frac{1}{2}\|\boldsymbol{x} - \frac{1}{N}\boldsymbol{W}\boldsymbol{y}\|_2^2$, from Lemma 19, we know that

$$L(\boldsymbol{y}_{t+1}) = L(\boldsymbol{y}_t) + < \frac{1}{N^2}\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y}_t - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{x}, \boldsymbol{y}_{t+1} - \boldsymbol{y}_t > + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \tag{10.368}$$

Let $\boldsymbol{a}_{t+1} = \boldsymbol{W}^\top \boldsymbol{x} + (\boldsymbol{I} - \frac{1}{N}\boldsymbol{W}^\top \boldsymbol{W})\boldsymbol{y}_t$. Together with Eq.(10.368), we can obtain that

$$L(\boldsymbol{y}_{t+1}) = L(\boldsymbol{y}_t) + <\frac{1}{N^2}\boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{y}_t - \frac{1}{N}\boldsymbol{W}^\top \boldsymbol{x}, \boldsymbol{y}_{t+1} - \boldsymbol{y}_t> + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \tag{10.369}$$

$$= L(\boldsymbol{y}_t) + \frac{1}{N} <\boldsymbol{y}_t - \boldsymbol{a}_{t+1}, \boldsymbol{y}_{t+1} - \boldsymbol{y}_t> + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \tag{10.370}$$

Note that $\boldsymbol{a}^\top \boldsymbol{b} = \frac{\|\boldsymbol{a}+\boldsymbol{b}\|_2^2 - \|\boldsymbol{a}\|_2^2 - \|\boldsymbol{b}\|_2^2}{2}$, it follows that

$$<\boldsymbol{y}_t - \boldsymbol{a}_{t+1}, \boldsymbol{y}_{t+1} - \boldsymbol{y}_t> = \frac{\|\boldsymbol{y}_{t+1} - \boldsymbol{a}_{t+1}\|_2^2 - \|\boldsymbol{y}_t - \boldsymbol{a}_{t+1}\|_2^2 - \|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2}{2} \tag{10.371}$$

Since $\boldsymbol{y}_{t+1} = h(\boldsymbol{a}_{t+1})$ is the solution of the proximal problem,
i.e., $\boldsymbol{y}_{t+1} = \arg\min_{\boldsymbol{y}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{a}_{t+1}\|_2^2 + \phi_\lambda(\boldsymbol{y})$, we can achieve that

$$\frac{1}{2}\|\boldsymbol{y}_{t+1} - \boldsymbol{a}_{t+1}\|_2^2 + \phi_\lambda(\boldsymbol{y}_{t+1}) \leq \frac{1}{2}\|\boldsymbol{y}_t - \boldsymbol{a}_{t+1}\|_2^2 + \phi_\lambda(\boldsymbol{y}_t) \tag{10.372}$$

It can be rewritten as

$$\frac{1}{2}\|\boldsymbol{y}_{t+1} - \boldsymbol{a}_{t+1}\|_2^2 - \frac{1}{2}\|\boldsymbol{y}_t - \boldsymbol{a}_{t+1}\|_2^2 \leq \phi_\lambda(\boldsymbol{y}_t) - \phi_\lambda(\boldsymbol{y}_{t+1}) \tag{10.373}$$

Together with Eq.(10.370), Eq.(10.371) and inequality (10.373), it follows that

$$L(\boldsymbol{y}_{t+1}) + \frac{1}{N}\phi_\lambda(\boldsymbol{y}_{t+1}) \leq L(\boldsymbol{y}_t) + \frac{1}{N}\phi_\lambda(\boldsymbol{y}_t) - \frac{1}{2N}\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \tag{10.374}$$

Together with Lemma 18, we can achieve that

$$\widehat{Q}(\boldsymbol{y}_{t+1}) \leq \widehat{Q}(\boldsymbol{y}_t) - \frac{1}{2N}\|(\boldsymbol{I}_d - \frac{1}{N}\boldsymbol{W}^\top \boldsymbol{W})(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \leq \widehat{Q}(\boldsymbol{y}_t) \tag{10.375}$$

□

10.24 Proof of Theorem 24

Before proving Theorem 24, we first show some useful Lemmas.

Lemma 20. *Denote* $L(\boldsymbol{y}) := \frac{1}{2}\|\boldsymbol{x} - \frac{1}{N}\boldsymbol{W}\boldsymbol{y}\|_2^2$. *For any bounded* $\boldsymbol{y}_t, \boldsymbol{y}_{t+1}, \boldsymbol{z} \in \mathcal{R}^N$, *we have*

$$L(\boldsymbol{y}_{t+1}) = L(\boldsymbol{z}) + \left\langle \frac{1}{N^2}\boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{y}_t - \frac{1}{N}\boldsymbol{W}^\top \boldsymbol{x}, \boldsymbol{y}_{t+1} - \boldsymbol{z} \right\rangle + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 - \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y}_t)\|_2^2 \tag{10.376}$$

Proof. Denote $L(\boldsymbol{y}) := \frac{1}{2}\|\boldsymbol{x} - \frac{1}{N}\boldsymbol{W}\boldsymbol{y}\|_2^2$. From Lemma 19, we can achieve that

$$L(\boldsymbol{z}) = L(\boldsymbol{y}_t) + <\frac{1}{N^2}\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y}_t - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{x}, \boldsymbol{z} - \boldsymbol{y}_t> + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y}_t)\|_2^2 \quad (10.377)$$

$$L(\boldsymbol{y}_{t+1}) = L(\boldsymbol{y}_t) + <\frac{1}{N^2}\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y}_t - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{x}, \boldsymbol{y}_{t+1} - \boldsymbol{y}_t> + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \quad (10.378)$$

It follows that

$$\begin{aligned} L(\boldsymbol{y}_{t+1}) &= L(\boldsymbol{z}) - \left\langle \frac{1}{N^2}\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y}_t - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{x}, \boldsymbol{z} - \boldsymbol{y}_t \right\rangle + \left\langle \frac{1}{N^2}\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y}_t - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{x}, \boldsymbol{y}_{t+1} - \boldsymbol{y}_t \right\rangle \\ &\quad + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 - \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y}_t)\|_2^2 \quad (10.379) \\ &= L(\boldsymbol{z}) + \left\langle \frac{1}{N^2}\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y}_t - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{x}, \boldsymbol{y}_{t+1} - \boldsymbol{z} \right\rangle + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \\ &\quad - \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y}_t)\|_2^2 \quad (10.380) \end{aligned}$$

□

Lemma 21. *For a convex function $\phi_\lambda(\cdot)$, let $h(\cdot)$ be the proximal operator w.r.t $\phi_\lambda(\cdot)$. Denote $\widehat{Q}(\boldsymbol{y}) := \frac{1}{2}\|\boldsymbol{x} - \frac{1}{N}\boldsymbol{W}\boldsymbol{y}\|_2^2 + \frac{1}{N}\phi_\lambda(\boldsymbol{y})$, for any bounded $\boldsymbol{y}_t, \boldsymbol{z} \in \mathcal{R}^N$, set $\boldsymbol{a}_{t+1} = \boldsymbol{W}^\top\boldsymbol{x} + (\boldsymbol{I} - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{W})\boldsymbol{y}_t$ and $\boldsymbol{y}_{t+1} = h(\boldsymbol{a_{t+1}})$, then we have*

$$\widehat{Q}(\boldsymbol{y}_{t+1}) \leq \widehat{Q}(\boldsymbol{z}) + \frac{1}{2N}\big(\|\boldsymbol{y}_t - \boldsymbol{z}\|_2^2 - \|\boldsymbol{y}_{t+1} - \boldsymbol{z}\|_2^2\big) - \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y}_t)\|_2^2 \quad (10.381)$$

Proof. Since $\boldsymbol{y}_{t+1} = \arg\min_{\boldsymbol{y}} \phi_\lambda(\boldsymbol{y}) + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{a}_{t+1}\|_2^2$, we have

$$\boldsymbol{0} \in \partial\phi(\boldsymbol{y}_{t+1}) + (\boldsymbol{y}_{t+1} - \boldsymbol{a}_{t+1}) \implies (\boldsymbol{a}_{t+1} - \boldsymbol{y}_{t+1}) \in \partial\phi(\boldsymbol{y}_{t+1}) \quad (10.382)$$

For a convex function $\phi_\lambda(\boldsymbol{y})$ and subgradient $\boldsymbol{g} \in \partial\phi_\lambda(\boldsymbol{y})$, we know $\phi_\lambda(\boldsymbol{z}) \geq \phi_\lambda(\boldsymbol{y}) + \langle \boldsymbol{g}, \boldsymbol{z} - \boldsymbol{y}\rangle$, it follows that

$$\phi_\lambda(\boldsymbol{z}) \geq \phi_\lambda(\boldsymbol{y}_{t+1}) + \left\langle \boldsymbol{a}_{t+1} - \boldsymbol{y}_{t+1}, \boldsymbol{z} - \boldsymbol{y}_{t+1} \right\rangle \quad (10.383)$$

Together with Lemma 20, we can obtain that

$$\begin{aligned} L(\boldsymbol{y}_{t+1}) + \frac{1}{N}\phi_\lambda(\boldsymbol{y}_{t+1}) &\leq L(\boldsymbol{z}) + \frac{1}{N}\phi_\lambda(\boldsymbol{z}) - \frac{1}{N}\left\langle \boldsymbol{a}_{t+1} - \boldsymbol{y}_{t+1}, \boldsymbol{z} - \boldsymbol{y}_{t+1} \right\rangle \\ &\quad + \left\langle \frac{1}{N^2}\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{y}_t - \frac{1}{N}\boldsymbol{W}^\top\boldsymbol{x}, \boldsymbol{y}_{t+1} - \boldsymbol{z} \right\rangle + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \\ &\quad - \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y}_t)\|_2^2 \quad (10.384) \end{aligned}$$

It follows that

$$\widehat{Q}(\boldsymbol{y}_{t+1}) \leq \widehat{Q}(\boldsymbol{z}) + \frac{1}{N}\langle \boldsymbol{y}_t - \boldsymbol{y}_{t+1}, \boldsymbol{y}_{t+1} - \boldsymbol{z}\rangle + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 - \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y}_t)\|_2^2 \tag{10.385}$$

Note that $\boldsymbol{a}^\top \boldsymbol{b} = \frac{\|\boldsymbol{a}+\boldsymbol{b}\|_2^2 - \|\boldsymbol{a}\|_2^2 - \|\boldsymbol{b}\|_2^2}{2}$, it follows that

$$\langle \boldsymbol{y}_t - \boldsymbol{y}_{t+1}, \boldsymbol{y}_{t+1} - \boldsymbol{z}\rangle = \frac{1}{2}\|\boldsymbol{y}_t - \boldsymbol{z}\|_2^2 - \frac{1}{2}\|\boldsymbol{y}_{t+1} - \boldsymbol{z}\|_2^2 - \frac{1}{2}\|\boldsymbol{y}_t - \boldsymbol{y}_{t+1}\|_2^2 \tag{10.386}$$

Together with inequality (10.385), we can achieve that

$$\begin{aligned}\widehat{Q}(\boldsymbol{y}_{t+1}) \leq \widehat{Q}(\boldsymbol{z}) + \frac{1}{2N}\big(\|\boldsymbol{y}_t - \boldsymbol{z}\|_2^2 - \|\boldsymbol{y}_{t+1} - \boldsymbol{z}\|_2^2 - \|\boldsymbol{y}_t - \boldsymbol{y}_{t+1}\|_2^2\big) + \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \\ - \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y}_t)\|_2^2 \end{aligned} \tag{10.387}$$

From Lemma 18, we know $\frac{1}{N}\|\boldsymbol{y}_t - \boldsymbol{y}_{t+1}\|_2^2 \geq \|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2$, it follows that

$$\widehat{Q}(\boldsymbol{y}_{t+1}) \leq \widehat{Q}(\boldsymbol{z}) + \frac{1}{2N}\big(\|\boldsymbol{y}_t - \boldsymbol{z}\|_2^2 - \|\boldsymbol{y}_{t+1} - \boldsymbol{z}\|_2^2\big) - \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{z} - \boldsymbol{y}_t)\|_2^2 \tag{10.388}$$

□

Lemma 22. *(Strictly Monotonic Descent) For a convex function $\phi_\lambda(\cdot)$, let $h(\cdot)$ be the proximal operator w.r.t $\phi_\lambda(\cdot)$. Denote $\widehat{Q}(\boldsymbol{y}) := \frac{1}{2}\|\boldsymbol{x} - \frac{1}{N}\boldsymbol{W}\boldsymbol{y}\|_2^2 + \frac{1}{N}\phi_\lambda(\boldsymbol{y})$, for any bounded $\boldsymbol{y}_t \in \mathcal{R}^N$, set $\boldsymbol{a}_{t+1} = \boldsymbol{W}^\top \boldsymbol{x} + (\boldsymbol{I} - \frac{1}{N}\boldsymbol{W}^\top \boldsymbol{W})\boldsymbol{y}_t$ and $\boldsymbol{y}_{t+1} = h(\boldsymbol{a_{t+1}})$, then we have*

$$\widehat{Q}(\boldsymbol{y}_{t+1}) \leq \widehat{Q}(\boldsymbol{y}_t) - \frac{1}{2N}\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 \tag{10.389}$$

Proof. From Lemma 21, setting $\boldsymbol{z} = \boldsymbol{y}_t$, we can directly get the result. □

Theorem. *For a convex function $\phi_\lambda(\cdot)$, denote $h(\cdot)$ as the proximal operator of $\phi_\lambda(\cdot)$. Given a bounded $\boldsymbol{x} \in \mathcal{R}^d$, set $\boldsymbol{y}_{t+1} = h\big(\boldsymbol{W}^\top \boldsymbol{x} + (\boldsymbol{I} - \frac{1}{N}\boldsymbol{W}^\top \boldsymbol{W})\boldsymbol{y}_t\big)$ with $\frac{1}{N}\boldsymbol{W}\boldsymbol{W}^\top = \boldsymbol{I}_d$. Denote $\widehat{Q}(\boldsymbol{y}) := \frac{1}{2}\|\boldsymbol{x} - \widehat{\mathcal{A}}(\boldsymbol{y})\|_2^2 + \frac{1}{N}\phi_\lambda(\boldsymbol{y})$ and $\boldsymbol{y}^*$ as an optimal of $\widehat{Q}(\cdot)$, for $T \geq 1$, we have*

$$\begin{aligned} T\big(\widehat{Q}(\boldsymbol{y}_T) - \widehat{Q}(\boldsymbol{y}^*)\big) \leq \frac{1}{2N}\|\boldsymbol{y}_0 - \boldsymbol{y}^*\|_2^2 - \frac{1}{2N}\|\boldsymbol{y}_T - \boldsymbol{y}^*\|_2^2 - \frac{1}{2}\sum_{t=0}^{T-1}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_t - \boldsymbol{y}^*)\|_2^2 \\ - \frac{1}{2}\sum_{t=0}^{T-1}\frac{t+1}{N}\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 \end{aligned} \tag{10.390}$$

Proof. From Lemma 21, setting $\boldsymbol{z} = \boldsymbol{y}^*$, we can achieve that

$$\widehat{Q}(\boldsymbol{y}_{t+1}) \leq \widehat{Q}(\boldsymbol{y}^*) + \frac{1}{2N}\big(\|\boldsymbol{y}_t - \boldsymbol{y}^*\|_2^2 - \|\boldsymbol{y}_{t+1} - \boldsymbol{y}^*\|_2^2\big) - \frac{1}{2}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}^* - \boldsymbol{y}_t)\|_2^2 \tag{10.391}$$

Telescope the inequality (10.391) from $t = 0$ to $t = T - 1$, we can obtain that

$$\sum_{t=0}^{T-1} \widehat{Q}(\boldsymbol{y}_{t+1}) - T\widehat{Q}(\boldsymbol{y}^*) \leq \frac{1}{2N}\|\boldsymbol{y}_0 - \boldsymbol{y}^*\|_2^2 - \frac{1}{2N}\|\boldsymbol{y}_T - \boldsymbol{y}^*\|_2^2 - \frac{1}{2}\sum_{t=0}^{T-1}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_t - \boldsymbol{y}^*)\|_2^2 \tag{10.392}$$

From Lemma 22, we know that

$$\widehat{Q}(\boldsymbol{y}_{t+1}) \leq \widehat{Q}(\boldsymbol{y}_t) - \frac{1}{2N}\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 \tag{10.393}$$

It follows that

$$\widehat{Q}(\boldsymbol{y}_T) \leq \widehat{Q}(\boldsymbol{y}_t) - \frac{1}{2N}\sum_{i=t}^{T-1}\|\boldsymbol{y}_{i+1} - \boldsymbol{y}_i\|_2^2 \tag{10.394}$$

Then, we can achieve that

$$T\widehat{Q}(\boldsymbol{y}_T) - T\widehat{Q}(\boldsymbol{y}^*) \leq \sum_{t=0}^{T-1}\widehat{Q}(\boldsymbol{y}_{t+1}) - T\widehat{Q}(\boldsymbol{y}^*) - \frac{1}{2N}\sum_{t=0}^{T-1}\sum_{i=t}^{T-1}\|\boldsymbol{y}_{i+1} - \boldsymbol{y}_i\|_2^2 \tag{10.395}$$

$$= \sum_{t=0}^{T-1}\widehat{Q}(\boldsymbol{y}_{t+1}) - T\widehat{Q}(\boldsymbol{y}^*) - \frac{1}{2N}\sum_{t=0}^{T-1}(t+1)\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 \tag{10.396}$$

Plug inequality (10.392) into inequality (10.396), we obtain that

$$T\big(\widehat{Q}(\boldsymbol{y}_T) - \widehat{Q}(\boldsymbol{y}^*)\big) \leq \frac{1}{2N}\|\boldsymbol{y}_0 - \boldsymbol{y}^*\|_2^2 - \frac{1}{2N}\|\boldsymbol{y}_T - \boldsymbol{y}^*\|_2^2 - \frac{1}{2}\sum_{t=0}^{T-1}\|\frac{1}{N}\boldsymbol{W}(\boldsymbol{y}_t - \boldsymbol{y}^*)\|_2^2$$

$$- \frac{1}{2}\sum_{t=0}^{T-1}\frac{t+1}{N}\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 \tag{10.397}$$

□

10.25 Proof of Theorem 25

We first show the structured samples $\boldsymbol{B}$ constructed in [117, 119].

Without loss of generality, we assume that $d = 2m, N = 2n$. Let $\boldsymbol{F} \in \mathbb{C}^{n\times n}$ be an $n \times n$ discrete Fourier matrix. $\boldsymbol{F}_{k,j} = e^{\frac{2\pi \boldsymbol{i} kj}{n}}$ is the $(k,j)^{th}$entry of $\boldsymbol{F}$, where $\boldsymbol{i} = \sqrt{-1}$. Let $\Lambda = \{k_1, k_2, ..., k_m\} \subset \{1, ..., n-1\}$ be a subset of indexes.

The structured matrix $\boldsymbol{B}$ can be constructed as Eq.(10.398).

$$\boldsymbol{B} = \frac{\sqrt{n}}{\sqrt{m}} \begin{bmatrix} \mathrm{Re}\boldsymbol{F}_\Lambda & -\mathrm{Im}\boldsymbol{F}_\Lambda \\ \mathrm{Im}\boldsymbol{F}_\Lambda & \mathrm{Re}\boldsymbol{F}_\Lambda \end{bmatrix} \in \mathbb{R}^{d\times N} \tag{10.398}$$

where Re and Im denote the real and imaginary parts of a complex number, and $\boldsymbol{F}_\Lambda$ in Eq. (10.399) is the matrix constructed by m rows of $\boldsymbol{F}$

$$\boldsymbol{F}_\Lambda = \frac{1}{\sqrt{n}} \begin{bmatrix} e^{\frac{2\pi \boldsymbol{i} k_1 1}{n}} & \cdots & e^{\frac{2\pi \boldsymbol{i} k_1 n}{n}} \\ \vdots & \ddots & \vdots \\ e^{\frac{2\pi \boldsymbol{i} k_m 1}{n}} & \cdots & e^{\frac{2\pi \boldsymbol{i} k_m n}{n}} \end{bmatrix} \in \mathbb{C}^{m\times n}. \tag{10.399}$$

The index set can be constructed by a closed-form solution [119] or by a coordinate descent method [117].

Specifically, for a prime number n such that m divides $n-1$, i.e., $m|(n-1)$, we can employ a closed-form construction as in [119]. Let g denote a primitive root modulo n. We can construct the index $\Lambda = \{k_1, k_2, ..., k_m\}$ as

$$\Lambda = \{g^0, g^{\frac{n-1}{m}}, g^{\frac{2(n-1)}{m}}, \cdots, g^{\frac{(m-1)(n-1)}{m}}\} \bmod n. \tag{10.400}$$

The resulted structured matrix $\boldsymbol{B}$ has a bounded mutual coherence, which is shown in Theorem 32.

Theorem 32. *[119] Suppose $d = 2m, N = 2n$, and n is a prime such that $m|(n-1)$. Construct matrix $\boldsymbol{B}$ as in Eq.(10.398) with index set Λ as Eq.(10.400). Let mutual coherence $\mu(\boldsymbol{B}) := \max_{i\neq j} \frac{|\boldsymbol{b}_i^\top \boldsymbol{b}_j|}{\|\boldsymbol{b}_i\|_2 \|\boldsymbol{b}_j\|_2}$. Then $\mu(\boldsymbol{B}) \leq \frac{\sqrt{n}}{m}$.*

Remark: The bound of mutual coherence in Theorem 32 is non-trivial when $n < m^2$. For the case $n \geq m^2$, we can use the coordinate descent method in [117] to minimize the mutual coherence.

We now show the orthogonal property of our data-dependent structured samples $\boldsymbol{D} = \frac{\sqrt{d}}{\sqrt{N}} \boldsymbol{R}^\top \boldsymbol{B}$

Proposition 2. *Suppose $d = 2m, N = 2n$. Let $\boldsymbol{D} = \frac{\sqrt{d}}{\sqrt{N}} \boldsymbol{R}^\top \boldsymbol{B}$ with $\boldsymbol{B}$ constructed as in Eq.(10.398). Then $\boldsymbol{D}\boldsymbol{D}^\top = \boldsymbol{I}_d$ and column vector has constant norm, i.e., $\|\boldsymbol{d}_j\|_2 = \sqrt{\frac{m}{n}}$, $\forall j \in \{1, \cdots, N\}$.*

Proof. Since $\boldsymbol{D}\boldsymbol{D}^\top = \frac{d}{N}\boldsymbol{B}\boldsymbol{B}^\top = \frac{m}{n}\boldsymbol{B}\boldsymbol{B}^\top = \widetilde{\boldsymbol{B}}\widetilde{\boldsymbol{B}}^\top$, where $\widetilde{\boldsymbol{B}} = \frac{\sqrt{m}}{\sqrt{n}}\boldsymbol{B}$. It follows that

$$\widetilde{\boldsymbol{B}} = \begin{bmatrix} \mathrm{Re}\boldsymbol{F}_\Lambda & -\mathrm{Im}\boldsymbol{F}_\Lambda \\ \mathrm{Im}\boldsymbol{F}_\Lambda & \mathrm{Re}\boldsymbol{F}_\Lambda \end{bmatrix} \in \mathbb{R}^{d\times N} \tag{10.401}$$

Let $\boldsymbol{c}_i \in \mathbb{C}^{1\times n}$ be the i^{th} row of matrix $\boldsymbol{F}_\Lambda \in \mathbb{C}^{m\times n}$ in Eq.(10.399). Let $\boldsymbol{v}_i \in \mathbb{R}^{1\times 2n}$ be the i^{th} row of matrix $\widetilde{\boldsymbol{B}} \in \mathbb{R}^{2m\times 2n}$ in Eq.(10.401). For $1 \le i, j \le m$, $i \ne j$, we know that

$$\boldsymbol{v}_i \boldsymbol{v}_{i+m}^\top = 0, \tag{10.402}$$
$$\boldsymbol{v}_{i+m} \boldsymbol{v}_{j+m}^\top = \boldsymbol{v}_i \boldsymbol{v}_j^\top = \mathrm{Re}(\boldsymbol{c}_i \boldsymbol{c}_j^*), \tag{10.403}$$
$$\boldsymbol{v}_{i+m} \boldsymbol{v}_j^\top = -\boldsymbol{v}_i \boldsymbol{v}_{j+m}^\top = \mathrm{Im}(\boldsymbol{c}_i \boldsymbol{c}_j^*), \tag{10.404}$$

where $*$ denotes the complex conjugate, $\mathrm{Re}(\cdot)$ and $\mathrm{Im}(\cdot)$ denote the real and imaginary parts of the input complex number.

For a discrete Fourier matrix $\boldsymbol{F}$, we know that

$$\boldsymbol{c}_i \boldsymbol{c}_j^* = \frac{1}{n} \sum_{k=0}^{n-1} e^{\frac{2\pi(i-j)k\boldsymbol{i}}{n}} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \tag{10.405}$$

When $i \ne j$, from Eq.(10.405), we know $\boldsymbol{c}_i \boldsymbol{c}_j^* = 0$. Thus, we have

$$\boldsymbol{v}_{i+m} \boldsymbol{v}_{j+m}^\top = \boldsymbol{v}_i \boldsymbol{v}_j^\top = \mathrm{Re}(\boldsymbol{c}_i \boldsymbol{c}_j^*) = 0, \tag{10.406}$$
$$\boldsymbol{v}_{i+m} \boldsymbol{v}_j^\top = -\boldsymbol{v}_i \boldsymbol{v}_{j+m}^\top = \mathrm{Im}(\boldsymbol{c}_i \boldsymbol{c}_j^*) = 0, \tag{10.407}$$

When $i = j$, we know that $\boldsymbol{v}_{i+m} \boldsymbol{v}_{i+m}^\top = \boldsymbol{v}_i \boldsymbol{v}_i^\top = \boldsymbol{c}_i \boldsymbol{c}_i^* = 1$.

Put two cases together, also note that $d = 2m$, we have $\boldsymbol{D}\boldsymbol{D}^\top = \widetilde{\boldsymbol{B}} \widetilde{\boldsymbol{B}}^\top = \boldsymbol{I}_d$.

The l_2-norm of the column vector of $\widetilde{\boldsymbol{B}}$ is given as

$$\|\widetilde{\boldsymbol{b}}_j\|_2^2 = \frac{1}{n} \sum_{i=1}^{m} \big(\sin^2 \frac{2\pi k_i j}{n} + \cos^2 \frac{2\pi k_i j}{n} \big) = \frac{m}{n} \tag{10.408}$$

Thus, we have $\|\boldsymbol{d}_j\|_2 = \|\widetilde{\boldsymbol{b}}_j\|_2^2 = \sqrt{\frac{m}{n}}$ for $j \in \{1, \cdots, M\}$

□

Lemma 23. *Let $\boldsymbol{D} = \frac{\sqrt{d}}{\sqrt{N}} \boldsymbol{R}^\top \boldsymbol{B}$, where $\boldsymbol{B}$ is constructed as as in Eq.(10.398) with index set Λ as Eq.(10.400) [119] with $N = 2n, d = 2m$. $\forall \boldsymbol{y} \in \mathcal{R}^N, \|\boldsymbol{y}\|_0 \le 2k$, we have $\|\boldsymbol{D}\boldsymbol{y}\|_2^2 - \|\boldsymbol{y}\|_2^2 \le -\frac{n-(2k-1)\sqrt{n}-m}{n} \|\boldsymbol{y}\|_2^2$*

Proof. Denote $\boldsymbol{M} = \boldsymbol{D}^\top \boldsymbol{D}$. Since the column vector of $\boldsymbol{D}$ has constant norm, i.e.,

$\|\boldsymbol{d}_j\|_2^2 = \frac{m}{n}$, it follows that

$$\|\boldsymbol{D}\boldsymbol{y}\|_2^2 = \boldsymbol{y}^\top \boldsymbol{M}\boldsymbol{y} = \|\boldsymbol{d}_j\|_2^2\big(\sum_{i=1}^N y_i^2 + \sum_{i=1}^N \sum_{j=1,j\neq i}^N y_i y_j M_{ij}\big) \tag{10.409}$$

$$= \frac{m}{n}\|\boldsymbol{y}\|_2^2 + \frac{m}{n}\sum_{i=1}^N \sum_{j=1,j\neq i}^N y_i y_j M_{ij} \tag{10.410}$$

$$\leq \frac{m}{n}\|\boldsymbol{y}\|_2^2 + \frac{m}{n}\mu(\boldsymbol{D})\big(\sum_{i=1}^N \sum_{j=1,j\neq i}^N |y_i||y_j|\big) \tag{10.411}$$

$$= \frac{m}{n}\|\boldsymbol{y}\|_2^2 + \frac{m}{n}\mu(\boldsymbol{D})\left(\big(\sum_{i=1}^N |y_i|\big)^2 - \sum_{i=1}^N y_i^2\right) \tag{10.412}$$

Since $\|\boldsymbol{y}\|_0 \leq 2k$, we know there is at most $2k$ non-zero elements among $\boldsymbol{y}$. Thus, we know that

$$\|\boldsymbol{D}\boldsymbol{y}\|_2^2 \leq \frac{m}{n}\|\boldsymbol{y}\|_2^2 + \frac{m}{n}\mu(\boldsymbol{D})\left(\big(\sum_{i=1}^N |y_i|\big)^2 - \sum_{i=1}^N y_i^2\right) \tag{10.413}$$

$$\leq \frac{m}{n}\|\boldsymbol{y}\|_2^2 + \frac{m}{n}\mu(\boldsymbol{D})\big(2k\sum_{i=1}^N y_i^2 - \sum_{i=1}^N y_i^2\big) \tag{10.414}$$

$$= \frac{m}{n}\|\boldsymbol{y}\|_2^2 + \frac{m}{n}\mu(\boldsymbol{D})(2k-1)\|\boldsymbol{y}\|_2^2 \tag{10.415}$$

Since $\mu(\boldsymbol{D}) = \boldsymbol{B}$, from Theorem 32, we know $\mu(\boldsymbol{D}) \leq \frac{\sqrt{n}}{m}$. It follows that

$$\|\boldsymbol{D}\boldsymbol{y}\|_2^2 \leq \frac{m}{n}\|\boldsymbol{y}\|_2^2 + \frac{m}{n}\mu(\boldsymbol{D})(2k-1)\|\boldsymbol{y}\|_2^2 \tag{10.416}$$

$$\leq \frac{m}{n}\|\boldsymbol{y}\|_2^2 + \frac{m}{n}\frac{(2k-1)\sqrt{n}}{m}\|\boldsymbol{y}\|_2^2 \tag{10.417}$$

$$= \frac{(2k-1)\sqrt{n}+m}{n}\|\boldsymbol{y}\|_2^2 \tag{10.418}$$

It follows that $\|\boldsymbol{D}\boldsymbol{y}\|_2^2 - \|\boldsymbol{y}\|_2^2 \leq \frac{(2k-1)\sqrt{n}+m-n}{n}\|\boldsymbol{y}\|_2^2$. □

Theorem. *(Strictly Monotonic Descent of k-sparse problem) Let* $L(\boldsymbol{y}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{y}\|_2^2$, *s.t.* $\|\boldsymbol{y}\|_0 \leq k$ *with* $\boldsymbol{D} = \frac{\sqrt{d}}{\sqrt{N}}\boldsymbol{R}^\top\boldsymbol{B}$, *where* $\boldsymbol{B}$ *is constructed as as in Eq.(10.398) with index set* Λ *as Eq.(10.400) [119] with* $N = 2n, d = 2m$. *Set* $\boldsymbol{y}_{t+1} = h(\boldsymbol{a}_{t+1})$ *with sparity* k *and* $\boldsymbol{a}_{t+1} = \boldsymbol{D}^\top\boldsymbol{x} + (\boldsymbol{I} - \boldsymbol{D}^\top\boldsymbol{D})\boldsymbol{y}_t$, *we have*

$$L(\boldsymbol{y}_{t+1}) \leq L(\boldsymbol{y}_t) + \frac{1}{2}\|\boldsymbol{y}_{t+1} - \boldsymbol{a}_{t+1}\|_2^2 - \frac{1}{2}\|\boldsymbol{y}_t - \boldsymbol{a}_{t+1}\|_2^2 - \frac{n-(2k-1)\sqrt{n}-m}{2n}\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 \leq L(\boldsymbol{y}_t) \tag{10.419}$$

where $h(\cdot)$ *is defined as*

$$h(z_j) = \begin{cases} z_j & \text{if } |z_j| \text{ is one of the } k\text{-highest values of } |\boldsymbol{z}| \in \mathcal{R}^N \\ 0 & \text{otherwise} \end{cases}. \tag{10.420}$$

Proof. Denote $L(\boldsymbol{y}) := \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{y}\|_2^2$. It follows that

$$L(\boldsymbol{y}_{t+1}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{y}_{t+1}\|_2^2 = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{y}_t + \boldsymbol{D}\boldsymbol{y}_t - \boldsymbol{D}\boldsymbol{y}_{t+1}\|_2^2 \tag{10.421}$$

$$= L(\boldsymbol{y}_t) + \left\langle \boldsymbol{x} - \boldsymbol{D}\boldsymbol{y}_t, \boldsymbol{D}(\boldsymbol{y}_t - \boldsymbol{y}_{t+1}) \right\rangle + \|\boldsymbol{D}(\boldsymbol{y}_t - \boldsymbol{y}_{t+1})\|_2^2 \tag{10.422}$$

$$= L(\boldsymbol{y}_t) + \left\langle \boldsymbol{D}^\top\boldsymbol{x} - \boldsymbol{D}^\top\boldsymbol{D}\boldsymbol{y}_t, \boldsymbol{y}_t - \boldsymbol{y}_{t+1} \right\rangle + \|\boldsymbol{D}(\boldsymbol{y}_t - \boldsymbol{y}_{t+1})\|_2^2 \tag{10.423}$$

$$= L(\boldsymbol{y}_t) + \left\langle \boldsymbol{D}^\top\boldsymbol{D}\boldsymbol{y}_t - \boldsymbol{D}^\top\boldsymbol{x}, \boldsymbol{y}_{t+1} - \boldsymbol{y}_t \right\rangle + \|\boldsymbol{D}(\boldsymbol{y}_t - \boldsymbol{y}_{t+1})\|_2^2 \tag{10.424}$$

Let $\boldsymbol{a}_{t+1} = \boldsymbol{D}^\top\boldsymbol{x} + (\boldsymbol{I} - \boldsymbol{D}^\top\boldsymbol{D})\boldsymbol{y}_t$, together with Eq.(10.424), we can obtain that

$$L(\boldsymbol{y}_{t+1}) = L(\boldsymbol{y}_t) + \left\langle \boldsymbol{y}_t - \boldsymbol{a}_{t+1}, \boldsymbol{y}_{t+1} - \boldsymbol{y}_t \right\rangle + \frac{1}{2}\|\boldsymbol{D}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \tag{10.425}$$

$$= L(\boldsymbol{y}_t) + \frac{\|\boldsymbol{y}_{t+1} - \boldsymbol{a}_{t+1}\|_2^2 - \|\boldsymbol{y}_t - \boldsymbol{a}_{t+1}\|_2^2 - \|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2}{2} + \frac{1}{2}\|\boldsymbol{D}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 \tag{10.426}$$

From Lemma 23, we know that

$$\frac{1}{2}\|\boldsymbol{D}(\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)\|_2^2 - \frac{1}{2}\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 \leq -\frac{n - (2k-1)\sqrt{n} - m}{2n}\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 \tag{10.427}$$

It follows that

$$L(\boldsymbol{y}_{t+1}) \leq L(\boldsymbol{y}_t) + \frac{1}{2}\|\boldsymbol{y}_{t+1} - \boldsymbol{a}_{t+1}\|_2^2 - \frac{1}{2}\|\boldsymbol{y}_t - \boldsymbol{a}_{t+1}\|_2^2 - \frac{n - (2k-1)\sqrt{n} - m}{2n}\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 \tag{10.428}$$

Note that $\boldsymbol{y}_{t+1} := \arg\min_{\boldsymbol{y}, \|\boldsymbol{y}\|_0 \leq k} \|\boldsymbol{y} - \boldsymbol{a}_{t+1}\|_2^2$, we know $\|\boldsymbol{y}_{t+1} - \boldsymbol{a}_{t+1}\|_2^2 \leq \|\boldsymbol{y}_t - \boldsymbol{a}_{t+1}\|_2^2$. It follows that $L(\boldsymbol{y}_{t+1}) \leq L(\boldsymbol{y}_t)$, in which the equality holds true when $\|\boldsymbol{y}_{t+1} - \boldsymbol{a}_{t+1}\|_2^2 = \|\boldsymbol{y}_t - \boldsymbol{a}_{t+1}\|_2^2$ and $\|\boldsymbol{y}_{t+1} - \boldsymbol{y}_t\|_2^2 = 0$

□

10.26 A Better Diagonal Random Rotation for SSF

In [117], a diagonal rotation matrix $\boldsymbol{D}$ is constructed by sampling its diagonal elements uniformly from $\{-1, +1\}$. In this section, we propose a better diagonal random rotation. Without loss of generality, we assume that $d = 2m, N = 2n$.

We first generate a diagonal complex matrix $\boldsymbol{D} \in \mathbb{C}^{m \times m}$, in which the diagonal elements are constructed as

$$\boldsymbol{D}_{jj} = \cos\theta_j + \boldsymbol{i}\,\sin\theta_j\ ,\ \forall j \in \{1, \cdots, m\} \tag{10.429}$$

where $\theta_j, \forall j \in \{1, \cdots, m\}$ are i.i.d. samples from the uniform distribution $Uni[0, 2\pi)$, and $\boldsymbol{i} = \sqrt{-1}$.

We then generate a uniformly random permutation $\Pi : \{1, \cdots, d\} \to \{1, \cdots, d\}$. The SSF samples can be constructed as $\boldsymbol{H} = \Pi \circ \widetilde{\boldsymbol{B}}$ with $\widetilde{\boldsymbol{B}}$:

$$\widetilde{\boldsymbol{B}} = \frac{\sqrt{n}}{\sqrt{m}} \begin{bmatrix} \mathrm{Re}\widetilde{\boldsymbol{F}}_\Lambda & -\mathrm{Im}\widetilde{\boldsymbol{F}}_\Lambda \\ \mathrm{Im}\widetilde{\boldsymbol{F}}_\Lambda & \mathrm{Re}\widetilde{\boldsymbol{F}}_\Lambda \end{bmatrix} \in \mathbb{R}^{d \times N} \tag{10.430}$$

where $\widetilde{\boldsymbol{F}}_\Lambda = \boldsymbol{D}\boldsymbol{F}_\Lambda$.

It is worth noting that $\boldsymbol{H}^\top \boldsymbol{H} = \boldsymbol{B}^\top \boldsymbol{B}$, which means that the proposed the diagonal rotation scheme preserved the pairwise inner product of SSF [117]. Moreover, the SSF with the proposed random rotation maintains $O(d)$ space complexity and $O(n \log n)$ (matrix-vector product) time complexity by FFT.

10.27 Rademacher Complexity

Neural Network Structure: For structured approximated NOK networks (SNOK), the 1-T layers are given as

$$\boldsymbol{y}_{t+1} = h(\boldsymbol{D}^\top \boldsymbol{R}_t \boldsymbol{x} + (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D})\boldsymbol{y}_t) \tag{10.431}$$

where $\boldsymbol{R}_t$ are free parameters such that $\boldsymbol{R}_t^\top \boldsymbol{R}_t = \boldsymbol{R}_t^\top \boldsymbol{R}_t = \boldsymbol{I}_d$. And $\boldsymbol{D}$ is the scaled structured spherical samples such that $\boldsymbol{D}\boldsymbol{D}^\top = \boldsymbol{I}_d$, and $\boldsymbol{y}_0 = \boldsymbol{0}$.

The last layer ($(T{+}1)^{th}$ layer) is given by $z = \boldsymbol{w}^\top \boldsymbol{y}_{T+1}$. Consider a L-Lipschitz continuous loss function $\ell(z, y) : \mathcal{Z} \times \mathcal{Y} \to [0, 1]$ with Lipschitz constant L w.r.t the input z.

Rademacher Complexity: Rademacher complexity of a function class $\mathcal{G}$ is defined as

$$\mathfrak{R}_N(\mathcal{G}) := \frac{1}{N}\mathbb{E}\left[\sup_{g \in \mathcal{G}} \sum_{i=1}^{N} \epsilon_i g(\boldsymbol{x}_i)\right] \tag{10.432}$$

where $\epsilon_i, i \in \{1, \cdots, N\}$ are i.i.d. samples drawn uniformly from $\{+1, -1\}$ with probality $\mathrm{P}[\epsilon_i = +1] = \mathrm{P}[\epsilon_i = -1] = 1/2$. And $\boldsymbol{x}_i, i \in \{1, \cdots, N\}$ are i.i.d. samples from $\mathcal{X}$.

Theorem. *(Rademacher Complexity Bound) Consider a Lipschitz continuous loss function $\ell(z, y) : \mathcal{Z} \times \mathcal{Y} \to [0, 1]$ with Lipschitz constant L w.r.t the input z. Let $\widetilde{\ell}(z, y) := \ell(z, y) - \ell(0, y)$. Let $\widehat{G}$ be the function class of our $(T{+}1)$-layer SNOK mapping from $\mathcal{X}$ to $\mathcal{Z}$. Suppose the activation function $|h(\boldsymbol{y})| \leq |\boldsymbol{y}|$ (element-wise), and the l_2-norm of last layer weight is bounded, i.e., $\|\boldsymbol{w}\|_2 \leq \mathcal{B}_w$. Let $(\boldsymbol{x}_i, y_i)_{i=1}^N$ be*

i.i.d. samples drawn from $\mathcal{X}\times\mathcal{Y}$. *Let* $\boldsymbol{Y}_{T+1}=[\boldsymbol{y}_{T+1}^{(1)},\cdots,\boldsymbol{y}_{T+1}^{(N)}]$ *be the* T^{th} *layer output with input* $\boldsymbol{X}$. *Denote the mutual coherence of* $\boldsymbol{Y}_{T+1}$ *as* μ^*, *i.e.,* $\mu^*=\mu(\boldsymbol{Y}_{T+1})=\max_{i\neq j}\frac{\boldsymbol{y}_{T+1}^{(i)\top}\boldsymbol{y}_{T+1}^{(j)}}{\|\boldsymbol{y}_{T+1}^{(i)}\|_2\|\boldsymbol{y}_{T+1}^{(j)}\|_2}\leq 1$. *Then, we have*

$$\mathfrak{R}_N(\widetilde{\ell}\circ\widehat{G})=\frac{1}{N}\mathbb{E}\left[\sup_{g\in\widehat{\mathcal{G}}}\sum_{i=1}^{N}\epsilon_i\widetilde{\ell}(g(\boldsymbol{x}_i),y_i)\right]\leq\frac{L\mathcal{B}_w\sqrt{\big((N-1)\mu^*+1\big)}}{N}\sqrt{\sum_{i=0}^{T-1}\beta^i}\|\boldsymbol{X}\|_F \tag{10.433}$$

where $\beta=\|\boldsymbol{I}-\boldsymbol{D}^\top\boldsymbol{D}\|_2^2\leq 1$, $\boldsymbol{X}=[\boldsymbol{x}_1,\cdots,\boldsymbol{x}_N]$. $\|\cdot\|_2$ *and* $\|\cdot\|_F$ *denote the spectral norm and the Frobenius norm of input matrix, respectively.*

Remark: When the width of NN $N_D>d$, we have $\beta=1$, and $\sqrt{\sum_{i=0}^{T-1}\beta^i}=\sqrt{T}$. In this case, the Rademacher complexity bound has a complexity $O(\sqrt{T})$ w.r.t. the depth of NN (SNOK).

Proof. Since $\widetilde{\ell}$ is L-Lipschitz continuous function, from the composition rule of Rademacher complexity, we know that

$$\mathfrak{R}_N(\widetilde{\ell}\circ\widehat{G})\leq L\,\mathfrak{R}_N(\widehat{G}) \tag{10.434}$$

It follows that

$$\mathfrak{R}_N(\widehat{G}) = \frac{1}{N}\mathbb{E}\left[\sup_{g\in\widehat{\mathcal{G}}}\sum_{i=1}^{N}\epsilon_i f(\boldsymbol{x}_i)\right] \tag{10.435}$$

$$= \frac{1}{N}\mathbb{E}\left[\sup_{\boldsymbol{w},\{\boldsymbol{R}_t\in\mathrm{SO}(d)\}_{t=1}^{T}}\sum_{i=1}^{N}\epsilon_i\big\langle\boldsymbol{w},\boldsymbol{y}_{T+1}^{(i)}\big\rangle\right] \tag{10.436}$$

$$= \frac{1}{N}\mathbb{E}\left[\sup_{\boldsymbol{w},\{\boldsymbol{R}_t\in\mathrm{SO}(d)\}_{t=1}^{T}}\big\langle\boldsymbol{w},\sum_{i=1}^{N}\epsilon_i\boldsymbol{y}_{T+1}^{(i)}\big\rangle\right] \tag{10.437}$$

$$\leq \frac{1}{N}\mathbb{E}\left[\sup_{\boldsymbol{w},\{\boldsymbol{R}_t\in\mathrm{SO}(d)\}_{t=1}^{T}}\|\boldsymbol{w}\|_2\,\big\|\sum_{i=1}^{N}\epsilon_i\boldsymbol{y}_{T+1}^{(i)}\big\|_2\right] \quad \text{(Cauchy-Schwarz inequality)} \tag{10.438}$$

$$\leq \frac{\mathcal{B}_w}{N}\mathbb{E}\left[\sup_{\{\boldsymbol{R}_t\in\mathrm{SO}(d)\}_{t=1}^{T}}\big\|\sum_{i=1}^{N}\epsilon_i\boldsymbol{y}_{T+1}^{(i)}\big\|_2\right] \tag{10.439}$$

$$= \frac{\mathcal{B}_w}{N}\mathbb{E}\left[\sup_{\{\boldsymbol{R}_t\in\mathrm{SO}(d)\}_{t=1}^{T}}\sqrt{\sum_{i=1}^{N}\big\|\epsilon_i\boldsymbol{y}_{T+1}^{(i)}\big\|_2^2+\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N}\epsilon_i\epsilon_j\boldsymbol{y}_{T+1}^{(i)\top}\boldsymbol{y}_{T+1}^{(j)}}\right] \tag{10.440}$$

$$= \frac{\mathcal{B}_w}{N}\mathbb{E}\left[\sup_{\{\boldsymbol{R}_t\in\mathrm{SO}(d)\}_{t=1}^{T}}\sqrt{\sum_{i=1}^{N}\big\|\boldsymbol{y}_{T+1}^{(i)}\big\|_2^2+\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N}\epsilon_i\epsilon_j\boldsymbol{y}_{T+1}^{(i)\top}\boldsymbol{y}_{T+1}^{(j)}}\right] \tag{10.441}$$

$$= \frac{\mathcal{B}_w}{N}\mathbb{E}\left[\sqrt{\sup_{\{\boldsymbol{R}_t\in\mathrm{SO}(d)\}_{t=1}^{T}}\sum_{i=1}^{N}\big\|\boldsymbol{y}_{T+1}^{(i)}\big\|_2^2+\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N}\epsilon_i\epsilon_j\boldsymbol{y}_{T+1}^{(i)\top}\boldsymbol{y}_{T+1}^{(j)}}\right] \tag{10.442}$$

$$\leq \frac{\mathcal{B}_w}{N}\sqrt{\mathbb{E}\left[\sup_{\{\boldsymbol{R}_t\in\mathrm{SO}(d)\}_{t=1}^{T}}\sum_{i=1}^{N}\big\|\boldsymbol{y}_{T+1}^{(i)}\big\|_2^2+\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N}\epsilon_i\epsilon_j\boldsymbol{y}_{T+1}^{(i)\top}\boldsymbol{y}_{T+1}^{(j)}\right]} \tag{10.443}$$

Inequality (10.443) is because of the Jensen inequality and concavity of the square root function.

Note that $|\epsilon_i| = 1, \forall i \in \{1, \cdots, N\}$, and the mutual coherence of $\boldsymbol{Y}_{T+1}$ is μ^*, i.e.,

$\mu^* = \mu(\boldsymbol{Y}_{T+1}) \leq 1$, it follows that

$$\mathfrak{R}_N(\widehat{F}) \leq \frac{\mathcal{B}_w}{N} \sqrt{\mathbb{E}\left[\sup_{\{\boldsymbol{R}_t \in \mathrm{SO}(d)\}_{t=1}^T} \sum_{i=1}^N \left\|\boldsymbol{y}_{T+1}^{(i)}\right\|_2^2 + \sum_{i=1}^N \sum_{j=1, j\neq i}^N \epsilon_i \epsilon_j \boldsymbol{y}_{T+1}^{(i)\top} \boldsymbol{y}_{T+1}^{(j)}\right]} \tag{10.444}$$

$$\leq \frac{\mathcal{B}_w}{N} \sqrt{\sup_{\{\boldsymbol{R}_t \in \mathrm{SO}(d)\}_{t=1}^T} \sum_{i=1}^N \left\|\boldsymbol{y}_{T+1}^{(i)}\right\|_2^2 + \sum_{i=1}^N \sum_{j=1, j\neq i}^N \left\|\boldsymbol{y}_{T+1}^{(i)}\right\|_2 \left\|\boldsymbol{y}_{T+1}^{(j)}\right\|_2 \mu^*} \tag{10.445}$$

$$= \frac{\mathcal{B}_w}{N} \sqrt{\sup_{\{\boldsymbol{R}_t \in \mathrm{SO}(d)\}_{t=1}^T} (1-\mu^*) \sum_{i=1}^N \left\|\boldsymbol{y}_{T+1}^{(i)}\right\|_2^2 + \mu^* \big(\sum_{i=1}^N \left\|\boldsymbol{y}_{T+1}^{(i)}\right\|_2\big)^2} \tag{10.446}$$

$$\leq \frac{\mathcal{B}_w}{N} \sqrt{\sup_{\{\boldsymbol{R}_t \in \mathrm{SO}(d)\}_{t=1}^T} (1-\mu^*) \|\boldsymbol{Y}_{T+1}\|_F^2 + N\mu^* \|\boldsymbol{Y}_{T+1}\|_F^2} \text{ Cauchy-Schwarz} \tag{10.447}$$

$$\leq \frac{\mathcal{B}_w}{N} \sqrt{\sup_{\{\boldsymbol{R}_t \in \mathrm{SO}(d)\}_{t=1}^T} \big((N-1)\mu^* + 1\big) \|\boldsymbol{Y}_{T+1}\|_F^2} \tag{10.448}$$

where $\boldsymbol{Y}_{T+1} = [\boldsymbol{y}_{T+1}^{(1)}, \cdots, \boldsymbol{y}_{T+1}^{(N)}]$ and $\|\cdot\|_F$ denotes the Frobenius norm.

Since $|h(\boldsymbol{Y})| \leq |\boldsymbol{Y}|$ (element-wise), (e.g., ReLU, max-pooling, soft-thresholding), it follows that

$$\|\boldsymbol{Y}_{T+1}\|_F^2 = \|h\big(\boldsymbol{D}^\top \boldsymbol{R}_T \boldsymbol{X} + (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{Y}_T\big)\|_F^2 \tag{10.449}$$

$$\leq \|\boldsymbol{D}^\top \boldsymbol{R}_T \boldsymbol{X} + (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{Y}_T\|_F^2 \tag{10.450}$$

In addition, we have

$$\|\boldsymbol{D}^\top \boldsymbol{R}_T \boldsymbol{X} + (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{Y}_T\|_F^2 = \|\boldsymbol{D}^\top \boldsymbol{R}_T \boldsymbol{X}\|_F^2 + \|(\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{Y}_T\|_F^2 + 2\big\langle \boldsymbol{D}^\top \boldsymbol{R}_T \boldsymbol{X}, (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{Y}_T \big\rangle \tag{10.451}$$

Note that $\boldsymbol{D}\boldsymbol{D}^\top = \boldsymbol{I}_d$ and $\boldsymbol{R}_T^\top \boldsymbol{R}_T = \boldsymbol{R}_T \boldsymbol{R}_T^\top = \boldsymbol{I}_d$, we have

$$\|\boldsymbol{D}^\top \boldsymbol{R}_T \boldsymbol{X}\|_F^2 = \|\boldsymbol{X}\|_F^2 \tag{10.452}$$

$$\big\langle \boldsymbol{D}^\top \boldsymbol{R}_T \boldsymbol{X}, (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{Y}_T \big\rangle = \mathrm{tr}\left(\boldsymbol{X}^\top \boldsymbol{R}_T^\top \boldsymbol{D} (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{Y}_T\right) = 0 \tag{10.453}$$

It follows that

$$\|\boldsymbol{D}^\top \boldsymbol{R}_T \boldsymbol{X} + (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{Y}_T\|_F^2 = \|\boldsymbol{D}^\top \boldsymbol{R}_T \boldsymbol{X}\|_F^2 + \|(\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{Y}_T\|_F^2 + 2\big\langle \boldsymbol{D}^\top \boldsymbol{R}_T \boldsymbol{X}, (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{Y}_T \big\rangle \tag{10.454}$$

$$= \|\boldsymbol{X}\|_F^2 + \|(\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{Y}_T\|_F^2 \tag{10.455}$$

$$\leq \|\boldsymbol{X}\|_F^2 + \|\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}\|_2^2 \|\boldsymbol{Y}_T\|_F^2 = \|\boldsymbol{X}\|_F^2 + \beta \|\boldsymbol{y}_T\|_F^2 \tag{10.456}$$

Recursively apply the above procedure from $t = T$ to $t = 1$, together with $\boldsymbol{Y}_0 = \boldsymbol{0}$, we can achieve that

$$\|\boldsymbol{Y}_{T+1}\|_F^2 \leq \|\boldsymbol{X}\|_F^2 \big(\sum_{i=0}^{T-1} \beta^i \big) \tag{10.457}$$

Together with inequality (10.448), it follows that

$$\mathfrak{R}_N(\widehat{G}) \leq \frac{\mathcal{B}_w}{N} \sqrt{\sup_{\{\boldsymbol{R}_t \in \mathrm{SO}(d)\}_{t=1}^T} \big((N-1)\mu^* + 1\big) \|\boldsymbol{Y}_{T+1}\|_F^2} \tag{10.458}$$

$$\leq \frac{\mathcal{B}_w \sqrt{\big((N-1)\mu^* + 1\big)}}{N} \sqrt{\sum_{i=0}^{T-1} \beta^i} \|\boldsymbol{X}\|_F \tag{10.459}$$

Finally, we obtain that

$$\mathfrak{R}_N(\widetilde{\ell} \circ \widehat{G}) = \frac{1}{N} \mathbb{E} \left[\sup_{g \in \widehat{\mathcal{G}}} \sum_{i=1}^N \epsilon_i \widetilde{\ell}(g(\boldsymbol{x}_i), y_i) \right] \leq \frac{L\mathcal{B}_w \sqrt{\big((N-1)\mu^* + 1\big)}}{N} \sqrt{\sum_{i=0}^{T-1} \beta^i} \|\boldsymbol{X}\|_F \tag{10.460}$$

Now, we show that $\beta = \|\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}\|_2^2 \leq 1$. From the definition of spectral norm, we have that

$$\beta = \|\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}\|_2^2 = \sup_{\|\boldsymbol{y}\|_2 = 1} \|(\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D})\boldsymbol{y}\|_2^2 \tag{10.461}$$

$$= \sup_{\|\boldsymbol{y}\|_2 = 1} \boldsymbol{y}^\top (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D})^\top (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{y} \tag{10.462}$$

$$= \sup_{\|\boldsymbol{y}\|_2 = 1} \boldsymbol{y}^\top \big(\boldsymbol{I} - 2\boldsymbol{D}^\top \boldsymbol{D} + \boldsymbol{D}^\top \boldsymbol{D} \boldsymbol{D}^\top \boldsymbol{D}\big) \boldsymbol{y} \tag{10.463}$$

$$= \sup_{\|\boldsymbol{y}\|_2 = 1} \boldsymbol{y}^\top (\boldsymbol{I} - \boldsymbol{D}^\top \boldsymbol{D}) \boldsymbol{y} \tag{10.464}$$

$$= 1 - \min_{\|\boldsymbol{y}\|_2 = 1} \|\boldsymbol{D}\boldsymbol{y}\|_2^2 \leq 1 \tag{10.465}$$

□

10.28 Generalization Bound

Theorem. *Consider a Lipschitz continuous loss function $\ell(z, y) : \mathcal{Z} \times \mathcal{Y} \to [0, 1]$ with Lipschitz constant L w.r.t the input z. Let $\widetilde{\ell}(z, y) := \ell(z, y) - \ell(0, y)$. Let $\widehat{G}$ be the function class of our (T+1)-layer SNOK mapping from $\mathcal{X}$ to $\mathcal{Z}$. Suppose the activation function $|h(\boldsymbol{y})| \leq |\boldsymbol{y}|$ (element-wise), and the l_2-norm of last layer weight is bounded, i.e., $\|\boldsymbol{w}\|_2 \leq \mathcal{B}_w$. Let $(\boldsymbol{x}_i, y_i)_{i=1}^N$ be i.i.d. samples drawn from $\mathcal{X} \times \mathcal{Y}$. Let*

$\boldsymbol{Y}_{T+1}$ be the T^{th} layer output with input $\boldsymbol{X}$. Denote the mutual coherence of $\boldsymbol{Y}_{T+1}$ as μ^*, i.e., $\mu^* = \mu(\boldsymbol{Y}_{T+1}) \leq 1$. Then, for $\forall N$ and $\forall \delta, 0 < \delta < 1$, with a probability at least $1-\delta$, $\forall g \in \widehat{G}$, we have

$$\mathbb{E}\big[\ell(g(X), Y)\big] \leq \frac{1}{N}\sum_{i=1}^{N} \ell(g(\boldsymbol{x}_i), y_i) + \frac{L\mathcal{B}_w\sqrt{\big((N-1)\mu^*+1\big)}}{N}\sqrt{\sum_{i=0}^{T-1}\beta^i}\|\boldsymbol{X}\|_F + \sqrt{\frac{8\ln(2/\delta)}{N}} \tag{10.466}$$

where $\beta = \|\boldsymbol{I} - \boldsymbol{D}^\top\boldsymbol{D}\|_2^2 \leq 1$, $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N]$, and $\|\cdot\|_F$ denotes the Frobenius norm.

Proof. Plug the Rademacher complexity bound of SNOK (our Theorem 26) into the Theorem 8 in [21], we can obtain the bound. □

10.29 Rademacher Complexity and Generalization Bound for General Structured Neural Network Family

Neural Network Structure: For a more general structured neural network family that includes SNOK, the 1-T layers are given as

$$\boldsymbol{y}_{t+1} = h(\boldsymbol{D}_t^\top\boldsymbol{x} + (\boldsymbol{I} - \boldsymbol{D}_t^\top\boldsymbol{D}_t)\boldsymbol{y}_t) \tag{10.467}$$

where $\boldsymbol{D}_t \in \mathcal{R}^{d_D \times d}$ are free parameters such that $\boldsymbol{D}_t\boldsymbol{D}_t^\top = \boldsymbol{I}_d$ and $d_D > d$, and $\boldsymbol{y}_0 = \boldsymbol{0}$.

The last layer ($(T+1)^{th}$ layer) is given by $z = \boldsymbol{w}^\top\boldsymbol{y}_{T+1}$. Consider a L-Lipschitz continuous loss function $\ell(z, y) : \mathcal{Z} \times \mathcal{Y} \to [0, 1]$ with Lipschitz constant L w.r.t the input z.

Theorem 33. *(Rademacher Complexity Bound) Consider a Lipschitz continuous loss function $\ell(z, y) : \mathcal{Z} \times \mathcal{Y} \to [0, 1]$ with Lipschitz constant L w.r.t the input z. Let $\widetilde{\ell}(z, y) := \ell(z, y) - \ell(0, y)$. Let $\widehat{G}$ be the function class of the above $(T+1)$-layer structured NN mapping from $\mathcal{X}$ to $\mathcal{Z}$. Suppose the activation function $|h(\boldsymbol{y})| \leq |\boldsymbol{y}|$ (element-wise), and the l_2-norm of last layer weight is bounded, i.e., $\|\boldsymbol{w}\|_2 \leq \mathcal{B}_w$. Let $(\boldsymbol{x}_i, y_i)_{i=1}^N$ be i.i.d. samples drawn from $\mathcal{X} \times \mathcal{Y}$. Let $\boldsymbol{Y}_{T+1} = [\boldsymbol{y}_{T+1}^{(1)}, \cdots, \boldsymbol{y}_{T+1}^{(N)}]$ be the T^{th} layer output with input $\boldsymbol{X}$. Denote the mutual coherence of $\boldsymbol{Y}_{T+1}$ as μ^*, i.e.,*

$\mu^* = \mu(\boldsymbol{Y}_{T+1}) = \max_{i \neq j} \frac{\boldsymbol{y}_{T+1}^{(i)\top} \boldsymbol{y}_{T+1}^{(j)}}{\|\boldsymbol{y}_{T+1}^{(i)}\|_2 \|\boldsymbol{y}_{T+1}^{(j)}\|_2} \leq 1$. *Then, we have*

$$\mathfrak{R}_N(\widetilde{\ell} \circ \widehat{G}) = \frac{1}{N} \mathbb{E}\left[\sup_{g \in \widehat{\mathcal{G}}} \sum_{i=1}^{N} \epsilon_i \widetilde{\ell}(g(\boldsymbol{x}_i), y_i)\right] \leq \frac{L\mathcal{B}_w \sqrt{T\big((N-1)\mu^* + 1\big)}}{N} \|\boldsymbol{X}\|_F \tag{10.468}$$

where $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N]$. $\|\cdot\|_2$ *and* $\|\cdot\|_F$ *denote the spectral norm and the Frobenius norm of input matrix, respectively.*

Remark: The Rademacher complexity bound has a complexity $O(\sqrt{T})$ w.r.t. the depth of NN.

Proof. Since $\widetilde{\ell}$ is L-Lipschitz continuous function, from the composition rule of Rademacher complexity, we know that

$$\mathfrak{R}_N(\widetilde{\ell} \circ \widehat{G}) \leq L\, \mathfrak{R}_N(\widehat{G}) \tag{10.469}$$

It follows that

$$\mathfrak{R}_N(\widehat{G}) = \frac{1}{N}\mathbb{E}\left[\sup_{g\in\widehat{\mathcal{G}}}\sum_{i=1}^{N}\epsilon_i g(\boldsymbol{x}_i)\right] \tag{10.470}$$

$$= \frac{1}{N}\mathbb{E}\left[\sup_{\boldsymbol{w},\{\boldsymbol{D}_t\in\mathcal{M}\}_{t=1}^{T}}\sum_{i=1}^{N}\epsilon_i\big\langle \boldsymbol{w}, \boldsymbol{y}_{T+1}^{(i)}\big\rangle\right] \tag{10.471}$$

$$= \frac{1}{N}\mathbb{E}\left[\sup_{\boldsymbol{w},\{\boldsymbol{D}_t\in\mathcal{M}\}_{t=1}^{T}}\Big\langle \boldsymbol{w}, \sum_{i=1}^{N}\epsilon_i\boldsymbol{y}_{T+1}^{(i)}\Big\rangle\right] \tag{10.472}$$

$$\leq \frac{1}{N}\mathbb{E}\left[\sup_{\boldsymbol{w},\{\boldsymbol{D}_t\in\mathcal{M}\}_{t=1}^{T}}\|\boldsymbol{w}\|_2\,\Big\|\sum_{i=1}^{N}\epsilon_i\boldsymbol{y}_{T+1}^{(i)}\Big\|_2\right] \quad \text{(Cauchy-Schwarz inequality)} \tag{10.473}$$

$$\leq \frac{\mathcal{B}_w}{N}\mathbb{E}\left[\sup_{\{\boldsymbol{D}_t\in\mathcal{M}\}_{t=1}^{T}}\Big\|\sum_{i=1}^{N}\epsilon_i\boldsymbol{y}_{T+1}^{(i)}\Big\|_2\right] \tag{10.474}$$

$$= \frac{\mathcal{B}_w}{N}\mathbb{E}\left[\sup_{\{\boldsymbol{D}_t\in\mathcal{M}\}_{t=1}^{T}}\sqrt{\sum_{i=1}^{N}\big\|\epsilon_i\boldsymbol{y}_{T+1}^{(i)}\big\|_2^2+\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N}\epsilon_i\epsilon_j\boldsymbol{y}_{T+1}^{(i)\top}\boldsymbol{y}_{T+1}^{(j)}}\right] \tag{10.475}$$

$$= \frac{\mathcal{B}_w}{N}\mathbb{E}\left[\sup_{\{\boldsymbol{D}_t\in\mathcal{M}\}_{t=1}^{T}}\sqrt{\sum_{i=1}^{N}\big\|\boldsymbol{y}_{T+1}^{(i)}\big\|_2^2+\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N}\epsilon_i\epsilon_j\boldsymbol{y}_{T+1}^{(i)\top}\boldsymbol{y}_{T+1}^{(j)}}\right] \tag{10.476}$$

$$= \frac{\mathcal{B}_w}{N}\mathbb{E}\left[\sqrt{\sup_{\{\boldsymbol{D}_t\in\mathcal{M}\}_{t=1}^{T}}\sum_{i=1}^{N}\big\|\boldsymbol{y}_{T+1}^{(i)}\big\|_2^2+\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N}\epsilon_i\epsilon_j\boldsymbol{y}_{T+1}^{(i)\top}\boldsymbol{y}_{T+1}^{(j)}}\right] \tag{10.477}$$

$$\leq \frac{\mathcal{B}_w}{N}\sqrt{\mathbb{E}\left[\sup_{\{\boldsymbol{D}_t\in\mathcal{M}\}_{t=1}^{T}}\sum_{i=1}^{N}\big\|\boldsymbol{y}_{T+1}^{(i)}\big\|_2^2+\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N}\epsilon_i\epsilon_j\boldsymbol{y}_{T+1}^{(i)\top}\boldsymbol{y}_{T+1}^{(j)}\right]} \tag{10.478}$$

Inequality (10.478) is because of the Jensen inequality and concavity of the square root function.

Note that $|\epsilon_i| = 1, \forall i \in \{1, \cdots, N\}$, and the mutual coherence of $\boldsymbol{Y}_{T+1}$ is μ^*, i.e.,

$\mu^* = \mu(\boldsymbol{Y}_{T+1}) \leq 1$, it follows that

$$\mathfrak{R}_N(\widehat{G}) \leq \frac{\mathcal{B}_w}{N} \sqrt{\mathbb{E}\left[\sup_{\{\boldsymbol{D}_t \in \mathcal{M}\}_{t=1}^T} \sum_{i=1}^N \left\|\boldsymbol{y}_{T+1}^{(i)}\right\|_2^2 + \sum_{i=1}^N \sum_{j=1, j\neq i}^N \epsilon_i \epsilon_j \boldsymbol{y}_{T+1}^{(i)\top} \boldsymbol{y}_{T+1}^{(j)}\right]} \tag{10.479}$$

$$\leq \frac{\mathcal{B}_w}{N} \sqrt{\sup_{\{\boldsymbol{D}_t \in \mathcal{M}\}_{t=1}^T} \sum_{i=1}^N \left\|\boldsymbol{y}_{T+1}^{(i)}\right\|_2^2 + \sum_{i=1}^N \sum_{j=1, j\neq i}^N \left\|\boldsymbol{y}_{T+1}^{(i)}\right\|_2 \left\|\boldsymbol{y}_{T+1}^{(j)}\right\|_2 \mu^*} \tag{10.480}$$

$$= \frac{\mathcal{B}_w}{N} \sqrt{\sup_{\{\boldsymbol{D}_t \in \mathcal{M}\}_{t=1}^T} (1-\mu^*) \sum_{i=1}^N \left\|\boldsymbol{y}_{T+1}^{(i)}\right\|_2^2 + \mu^* \Big(\sum_{i=1}^N \left\|\boldsymbol{y}_{T+1}^{(i)}\right\|_2\Big)^2} \tag{10.481}$$

$$\leq \frac{\mathcal{B}_w}{N} \sqrt{\sup_{\{\boldsymbol{D}_t \in \mathcal{M}\}_{t=1}^T} (1-\mu^*) \|\boldsymbol{Y}_{T+1}\|_F^2 + N\mu^* \|\boldsymbol{Y}_{T+1}\|_F^2} \text{ Cauchy-Schwarz} \tag{10.482}$$

$$\leq \frac{\mathcal{B}_w}{N} \sqrt{\sup_{\{\boldsymbol{D}_t \in \mathcal{M}\}_{t=1}^T} \big((N-1)\mu^* + 1\big) \|\boldsymbol{Y}_{T+1}\|_F^2} \tag{10.483}$$

where $\boldsymbol{Y}_{T+1} = [\boldsymbol{y}_{T+1}^{(1)}, \cdots, \boldsymbol{y}_{T+1}^{(N)}]$ and $\|\cdot\|_F$ denotes the Frobenius norm.

Since $|h(\boldsymbol{Y})| \leq |\boldsymbol{Y}|$ (element-wise), (e.g., ReLU, max-pooling, soft-thresholding), it follows that

$$\|\boldsymbol{Y}_{T+1}\|_F^2 = \|h\big(\boldsymbol{D}_T^\top \boldsymbol{X} + (\boldsymbol{I} - \boldsymbol{D}_T^\top \boldsymbol{D}_T)\boldsymbol{Y}_T\big)\|_F^2 \tag{10.484}$$

$$\leq \|\boldsymbol{D}_T^\top \boldsymbol{X} + (\boldsymbol{I} - \boldsymbol{D}_T^\top \boldsymbol{D}_T)\boldsymbol{Y}_T\|_F^2 \tag{10.485}$$

In addition, we have

$$\|\boldsymbol{D}_T^\top \boldsymbol{X} + (\boldsymbol{I} - \boldsymbol{D}_T^\top \boldsymbol{D}_T)\boldsymbol{Y}_T\|_F^2 = \|\boldsymbol{D}_T^\top \boldsymbol{X}\|_F^2 + \|(\boldsymbol{I} - \boldsymbol{D}_T^\top \boldsymbol{D}_T)\boldsymbol{Y}_T\|_F^2 + 2\big\langle \boldsymbol{D}_T^\top \boldsymbol{X}, (\boldsymbol{I} - \boldsymbol{D}_T^\top \boldsymbol{D}_T)\boldsymbol{Y}_T \big\rangle \tag{10.486}$$

Note that $\boldsymbol{D}_T \boldsymbol{D}_T^\top = \boldsymbol{I}_d$, we have

$$\|\boldsymbol{D}_T^\top \boldsymbol{X}\|_F^2 = \|\boldsymbol{X}\|_F^2 \tag{10.487}$$

$$\big\langle \boldsymbol{D}_T^\top \boldsymbol{X}, (\boldsymbol{I} - \boldsymbol{D}_T^\top \boldsymbol{D}_T)\boldsymbol{Y}_T \big\rangle = \operatorname{tr}\left(\boldsymbol{X}_T^\top \boldsymbol{D}_T (\boldsymbol{I} - \boldsymbol{D}_T^\top \boldsymbol{D}_T)\boldsymbol{Y}_T\right) = 0 \tag{10.488}$$

It follows that

$$\|\boldsymbol{D}_T^\top \boldsymbol{X} + (\boldsymbol{I} - \boldsymbol{D}_T^\top \boldsymbol{D}_T)\boldsymbol{Y}_T\|_F^2 = \|\boldsymbol{D}_T^\top \boldsymbol{X}\|_F^2 + \|(\boldsymbol{I} - \boldsymbol{D}_T^\top \boldsymbol{D}_T)\boldsymbol{Y}_T\|_F^2 + 2\big\langle \boldsymbol{D}_T^\top \boldsymbol{X}, (\boldsymbol{I} - \boldsymbol{D}_T^\top \boldsymbol{D}_T)\boldsymbol{Y}_T \big\rangle \tag{10.489}$$

$$= \|\boldsymbol{X}\|_F^2 + \|(\boldsymbol{I} - \boldsymbol{D}_T^\top \boldsymbol{D}_T)\boldsymbol{Y}_T\|_F^2 \tag{10.490}$$

$$\leq \|\boldsymbol{X}\|_F^2 + \|\boldsymbol{I} - \boldsymbol{D}_T^\top \boldsymbol{D}_T\|_2^2 \|\boldsymbol{Y}_T\|_F^2 = \|\boldsymbol{X}\|_F^2 + \|\boldsymbol{y}_T\|_F^2 \tag{10.491}$$

Recursively apply the above procedure from $t = T$ to $t = 1$, together with $\boldsymbol{Y}_0 = \boldsymbol{0}$, we can achieve that

$$\|\boldsymbol{Y}_{T+1}\|_F^2 \leq T\|\boldsymbol{X}\|_F^2 \tag{10.492}$$

Together with inequality (10.483), it follows that

$$\mathfrak{R}_N(\widehat{G}) \leq \frac{\mathcal{B}_w}{N}\sqrt{\sup_{\{\boldsymbol{D}_t\in\mathcal{M}\}_{t=1}^T}\big((N-1)\mu^*+1\big)\|\boldsymbol{Y}_{T+1}\|_F^2} \tag{10.493}$$

$$\leq \frac{\mathcal{B}_w\sqrt{T\big((N-1)\mu^*+1\big)}}{N}\|\boldsymbol{X}\|_F \tag{10.494}$$

Finally, we obtain that

$$\mathfrak{R}_N(\widetilde{\ell}\circ\widehat{G}) = \frac{1}{N}\mathbb{E}\left[\sup_{g\in\widehat{\mathcal{G}}}\sum_{i=1}^N \epsilon_i\widetilde{\ell}(g(\boldsymbol{x}_i),y_i)\right] \leq \frac{L\mathcal{B}_w\sqrt{T\big((N-1)\mu^*+1\big)}}{N}\|\boldsymbol{X}\|_F \tag{10.495}$$

□

Theorem 34. *Consider a Lipschitz continuous loss function $\ell(z,y):\mathcal{Z}\times\mathcal{Y}\to[0,1]$ with Lipschitz constant L w.r.t the input z. Let $\widetilde{\ell}(z,y) := \ell(z,y)-\ell(0,y)$. Let $\widehat{G}$ be the function class of our general (T+1)-layer structured NN mapping from $\mathcal{X}$ to $\mathcal{Z}$. Suppose the activation function $|h(\boldsymbol{y})|\leq|\boldsymbol{y}|$ (element-wise), and the l_2-norm of last layer weight is bounded, i.e., $\|\boldsymbol{w}\|_2\leq\mathcal{B}_w$. Let $(\boldsymbol{x}_i,y_i)_{i=1}^N$ be i.i.d. samples drawn from $\mathcal{X}\times\mathcal{Y}$. Let $\boldsymbol{Y}_{T+1}$ be the T^{th} layer output with input $\boldsymbol{X}$. Denote the mutual coherence of $\boldsymbol{Y}_{T+1}$ as μ^*, i.e., $\mu^*=\mu(\boldsymbol{Y}_{T+1})\leq 1$. Then, for $\forall N$ and $\forall\delta, 0<\delta<1$, with a probability at least $1-\delta$, $\forall g\in\widehat{G}$, we have*

$$\mathbb{E}\big[\ell(g(X),Y)\big] \leq \frac{1}{N}\sum_{i=1}^N\ell(g(\boldsymbol{x}_i),y_i) + \frac{L\mathcal{B}_w\sqrt{T\big((N-1)\mu^*+1\big)}}{N}\|\boldsymbol{X}\|_F + \sqrt{\frac{8\ln(2/\delta)}{N}} \tag{10.496}$$

where $\boldsymbol{X}=[\boldsymbol{x}_1,\cdots,\boldsymbol{x}_N]$, and $\|\cdot\|_F$ denotes the Frobenius norm.

Proof. Plug the Rademacher complexity bound of general structured NN (our Theorem 33) into the Theorem 8 in [21], we can obtain the bound. □

10.30 Explanation of Theorem 28 for robust learning

Theorem. ***(Monotonic Relationship)*** *([77] Let $p(x,y)$ and $q(x,y)$ be the training and test density,respectively. Define $r(x,y) = q(x,y)/p(x,y)$ and $r_i = r(x_i, y_i)$. Let $l(\widehat{y}, y) = \mathbf{1}\big(sign(\widehat{y}) \neq y\big)$ and $l(\widehat{y}, y) = \mathbf{1}\big(argmax_k(\widehat{y}_k) \neq y\big)$ be 0-1 loss for binary classification and multi-class classification, respectively. Let $f(\cdot)$ be convex with $f(1) = 0$. Define risk $\mathcal{R}(\theta)$, empirical risk $\widehat{\mathcal{R}}(\theta)$, adversarial risk $\mathcal{R}_{adv}(\theta)$ and empirical adversarial risk $\widehat{\mathcal{R}}_{adv}(\theta)$ as*

$$\mathcal{R}(\theta) = \mathbb{E}_{p(x,y)}\left[l(g_\theta(x), y)\right] \tag{10.497}$$

$$\widehat{\mathcal{R}}(\theta) = \frac{1}{n}\sum\nolimits_{i=1}^{n} l(g_\theta(x_i), y_i) \tag{10.498}$$

$$\mathcal{R}_{adv}(\theta) = \sup_{r \in \mathcal{U}_f} \mathbb{E}_{p(x,y)}\left[r(x,y) l(g_\theta(x), y)\right] \tag{10.499}$$

$$\widehat{\mathcal{R}}_{adv}(\theta) = \sup_{\mathbf{r} \in \widehat{\mathcal{U}}_f} \frac{1}{n}\sum\nolimits_{i=1}^{n} r_i l(g_\theta(x_i), y_i), \tag{10.500}$$

where $\mathcal{U}_f = \left\{r(x,y) \,\middle|\, \mathbb{E}_{p(x,y)}\left[f\left(r(x,y)\right)\right] \leq \delta, \mathbb{E}_{p(x,y)}\left[r(x,y)\right] = 1, r(x,y) \geq 0, \forall (x,y) \in \mathcal{X} \times \mathcal{Y}\right\}$ and $\widehat{\mathcal{U}}_f = \left\{\mathbf{r} \,\middle|\, \frac{1}{n}\sum_{i=1}^{n} f(r_i) \leq \delta, \frac{1}{n}\sum_{i=1}^{n} r_i = 1, \mathbf{r} \geq 0\right\}$. Then we have that

$$\textbf{\textit{If}}\ \mathcal{R}_{adv}(\theta_1) < 1, \quad \textbf{\textit{then}}\ \mathcal{R}(\theta_1) < \mathcal{R}(\theta_2) \iff \mathcal{R}_{adv}(\theta_1) < \mathcal{R}_{adv}(\theta_2). \tag{10.501}$$

$$\textbf{\textit{If}}\ \mathcal{R}_{adv}(\theta_1) = 1, \quad \textbf{\textit{then}}\ \mathcal{R}(\theta_1) \leq \mathcal{R}(\theta_2) \iff \mathcal{R}_{adv}(\theta_2) = 1. \tag{10.502}$$

The same monotonic relationship holds between their empirical approximation: $\widehat{\mathcal{R}}(\theta)$ and $\widehat{\mathcal{R}}_{adv}$.

[77] show that minimizing (empirical) risk is equivalent to minimize the (empirical) adversarial risk (worst-case risk) for 0-1 loss. Thus, we can directly optimize the risk instead of the worst-case risk. Specifically, suppose we have an observable training distribution $p(x,y)$. The observable distribution $p(x,y)$ may be corrupted from an underlying clean distribution $q(x,y)$. We train a model based on the training distribution $p(x,y)$, and we want our model to perform well on the clean distribution $q(x,y)$. Since we do not know the clean distribution $q(x,y)$, we want our model to perform well for the worst-case estimate of the clean distribution, with the assumption that the f-divergence between the corrupted distribution p and the clean distribution q is bounded by δ. Note that the underlying clean distribution is fixed but unknown, given the corrupted training distribution, the smallest δ that bounds the divergence between the corrupted distribution and clean distribution measures

the intrinsic difficulty of the corruption, and it is also fixed and unknown. The corresponding worst-case distribution w.r.t the smallest δ is an estimate of the true clean distribution, and this worst-case risk upper bounds the risk of the true clean distribution. In addition, this bound is tighter than the other worst-case risks w.r.t larger δ. Formally, the upper bound w.r.t the smallest δ is given as

$$G(\theta) := \sup_{q \in \widetilde{\mathcal{U}}_f} \mathbb{E}_{q(x,y)} \left[l(g_\theta(x), y) \right] \tag{10.503}$$

where $\widetilde{\mathcal{U}}_f$ is an equivalent constrainted set w.r.t $\mathcal{U}_f$ for $q(x, y)$. Then, we have

$$G(\theta) := \sup_{q \in \widetilde{\mathcal{U}}_f} \mathbb{E}_{q(x,y)} \left[l(g_\theta(x), y) \right] = \sup_{r \in \mathcal{U}_f} \mathbb{E}_{p(x,y)} \left[r(x, y) l(g_\theta(x), y) \right] \tag{10.504}$$

When $l(\cdot)$ is 0-1 loss, from Theorem 1, we know that minimize $G(\theta)$ is equivalent to minimize $\widetilde{G}(\theta)$. Thus, we can minimize $\widetilde{G}(\theta)$ instead of $G(\theta)$.

$$\widetilde{G}(\theta) := \mathbb{E}_{p(x,y)} \left[l(g_\theta(x), y) \right] \tag{10.505}$$

Minimize the Eq.(10.505) enables us to minimize the Eq.(10.503) without knowing the true divergence parameter δ beforehand. Usually, minimizing the upper bound can decrease the true risk under clean distribution. Particularly, when the clean distribution coincides with the worst-case estimate w.r.t the smallest δ, minimizing the risk under the corrupted training distribution leads to the same minimizer as minimizing the risk under the clean distribution.

Relationship between label corruption and general corruption

Label corruption is a special case of general corruption. Label corruption restricts the corruption in the space $\mathcal{Y}$ instead of the space $\mathcal{X} \times \mathcal{Y}$. That is to say, the training distribution $p(x)$ is same as the clean distribution $q(x)$ over $\mathcal{X}$. Then, we have the robust risk for label corruption as

$$G_y(\theta) := \sup_{q \in \widetilde{\mathcal{U}}_f \cap H} \mathbb{E}_{q(x,y)} \left[l(g_\theta(x), y) \right] \tag{10.506}$$

where $H := \{q(x, y) \,|q(x) = p(x), \forall (x, y) \in \mathcal{X} \times \mathcal{Y}\}$. The supremum in $G_y(\theta)$ is taken over $\widetilde{\mathcal{U}}_f \cap H$, while the supremum in $G(\theta)$ is taken over $\widetilde{\mathcal{U}}_f$. Due to the additional constrain $q(x) = p(x), \forall (x, y) \in \mathcal{X} \times \mathcal{Y}$, we thus know that the robust risk $G_y(\theta)$ is bounded by $G(\theta)$, i.e., $G_y(\theta) \leq G(\theta)$. Moreover, it is more piratical and important to be robust for both label corruption and feature corruption.

10.31 Proof of Theorem 29

Proof. Because $\mathbf{1}\big(u<0\big)\leq l(u)$, we have $\sum_{i=1}^{n} l(u_i)\geq\sum_{i=1}^{n}\mathbf{1}\big(u_i<0\big)$. Then

$$
\begin{aligned}
Q\left(\mathbf{u}\right) &= \min_{\mathbf{v}\in\{0,1\}^n}\max\big(\sum\nolimits_{i=1}^{n} v_i l(u_i), n-\sum\nolimits_{i=1}^{n} v_i+\sum\nolimits_{i=1}^{n}\mathbf{1}\big(u_i<0\big)\big) && (10.507)\\
&\leq \max\big(\sum\nolimits_{i=1}^{n} l(u_i), n-\sum\nolimits_{i=1}^{n} 1+\sum\nolimits_{i=1}^{n}\mathbf{1}\big(u_i<0\big)\big) && (10.508)\\
&= \max\big(\sum\nolimits_{i=1}^{n} l(u_i), \sum\nolimits_{i=1}^{n}\mathbf{1}\big(u_i<0\big)\big) && (10.509)\\
&= \sum\nolimits_{i=1}^{n} l(u_i) && (10.510)
\end{aligned}
$$

Since loss $\widehat{J}(\mathbf{u})=\sum_{i=1}^{n} l(u_i)$, we obtain $Q\left(\mathbf{u}\right)\leq\widehat{J}\left(\mathbf{u}\right)$.

On the other hand, we have that

$$
\begin{aligned}
Q\left(\mathbf{u}\right) &= \min_{\mathbf{v}\in\{0,1\}^n}\max\big(\sum\nolimits_{i=1}^{n} v_i l(u_i), n-\sum\nolimits_{i=1}^{n} v_i+\sum\nolimits_{i=1}^{n}\mathbf{1}\big(u_i<0\big)\big)\\
&\geq \min_{\mathbf{v}\in\{0,1\}^n} n-\sum\nolimits_{i=1}^{n} v_i+\sum\nolimits_{i=1}^{n}\mathbf{1}\big(u_i<0\big) && (10.511)\\
&= \sum\nolimits_{i=1}^{n}\mathbf{1}\big(u_i<0\big) && (10.512)
\end{aligned}
$$

Since $J(\mathbf{u})=\sum_{i=1}^{n}\mathbf{1}\big(u_i<0\big)$, we obtain $Q\left(\mathbf{u}\right)\geq J\left(\mathbf{u}\right)$

□

10.32 Proof of Corollary 4

Proof. Since $n=mb$, similar to the proof of $Q\left(\mathbf{u}\right)\leq\widehat{J}\left(\mathbf{u}\right)$, we have

$$
\begin{aligned}
\widehat{Q}\left(\mathbf{u}\right) =& \sum\nolimits_{j=1}^{b}\min_{\mathbf{v}\in\{0,1\}^m}\max\big(\sum\nolimits_{i=1}^{m} v_{ij} l(u_{ij}), m-\sum\nolimits_{i=1}^{m} v_{ij}+\sum\nolimits_{i=1}^{m}\mathbf{1}\big(u_{ij}<0\big)\big)\\
\leq& \sum\nolimits_{j=1}^{b}\max\big(\sum\nolimits_{i=1}^{m} l(u_{ij}), m-\sum\nolimits_{i=1}^{m} 1+\sum\nolimits_{i=1}^{m}\mathbf{1}\big(u_{ij}<0\big)\big) && (10.513)\\
=& \sum\nolimits_{j=1}^{b}\max\big(\sum\nolimits_{i=1}^{m} l(u_{ij}), \sum\nolimits_{i=1}^{m}\mathbf{1}\big(u_{ij}<0\big)\big) && (10.514)\\
=& \sum\nolimits_{j=1}^{b}\sum\nolimits_{i=1}^{m} l(u_{ij})=\widehat{J}\left(\mathbf{u}\right) && (10.515)
\end{aligned}
$$

On the other hand, since the group (batch) separable sum structure, we have that

$$\begin{aligned}\widehat{Q}\left(\mathbf{u}\right) =& \sum_{j=1}^{b} \min_{\mathbf{v}\in\{0,1\}^m} \max\left(\sum_{i=1}^{m} v_{ij} l(u_{ij}), m - \sum_{i=1}^{m} v_{ij} + \sum_{i=1}^{m} \mathbf{1}\left(u_{ij} < 0\right)\right) \\ =& \min_{\mathbf{v}\in\{0,1\}^n} \sum_{j=1}^{b} \max\left(\sum_{i=1}^{m} v_{ij} l(u_{ij}), m - \sum_{i=1}^{m} v_{ij} + \sum_{i=1}^{m} \mathbf{1}\left(u_{ij} < 0\right)\right) \end{aligned} \tag{10.516}$$

$$\geq \min_{\mathbf{v}\in\{0,1\}^n} \max\left(\sum_{j=1}^{b}\sum_{i=1}^{m} v_{ij} l(u_{ij}), n - \sum_{j=1}^{b}\sum_{i=1}^{m} v_{ij} + \sum_{j=1}^{b}\sum_{i=1}^{m} \mathbf{1}\left(u_{ij} < 0\right)\right) \tag{10.517}$$

$$= Q\left(\mathbf{u}\right) \geq J\left(\mathbf{u}\right) \tag{10.518}$$

□

10.33 Proof of Partial Optimization Theorem (Theorem 31)

Proof. For simplicity, let $l_i = l(u_i)$, $i \in \{1, ..., n\}$. Without loss of generality, assume $l_1 \leq l_2 \cdots \leq l_n$. Let $\mathbf{v}^*$ be the solution obtained by Algorithm 13. Assume there exits a $\mathbf{v}$ such that

$$\max\left(\sum_{i=1}^{n} v_i l_i, C - \sum_{i=1}^{n} v_i\right) < \max\left(\sum_{i=1}^{n} v_i^* l_i, C - \sum_{i=1}^{n} v_i^*\right). \tag{10.519}$$

Let $T = \sum_{i=1}^{n} v_i$ and $T^* = \sum_{i=1}^{n} v_i^*$.

Case 1: If $T = T^*$, then there exists an $v_k = 1$ and $v_k^* = 0$. From Algorithm 13, we know $k > T^*$ ($v_k^* = 0 \Rightarrow k > T^*$) and $l_k \geq l_j, j \in \{1, ..., T^*\}$. Then we know $\sum_{i=1}^{n} v_i^* l_i \leq \sum_{i=1}^{n} v_i l_i$. Thus, we can achieve that

$$\max\left(\sum_{i=1}^{n} v_i^* l_i, C - \sum_{i=1}^{n} v_i^*\right) = \max(\sum_{i=1}^{n} v_i^* l_i, C - \sum_{i=1}^{n} v_i) \tag{10.520}$$

$$\leq \max\left(\sum_{i=1}^{n} v_i l_i, C - \sum_{i=1}^{n} v_i\right). \tag{10.521}$$

This contradicts the assumption in Eq.(10.519)

Case 2: If $T > T^*$, then there exists an $v_k = 1$ and $v_k^* = 0$. Let $\mathrm{L}_{T^*} = \sum_{i=1}^{T^*} l_i$. Since $l_k \geq 0$, we have $\mathrm{L}_{T^*} + l_k \geq \mathrm{L}_{T^*}$. From Algorithm 13, we know that $\mathrm{L}_{T^*} + l_k > C - T^*$.

Thus we obtain that

$$\max\big(\sum_{i=1}^{n} v_i l_i, C - \sum_{i=1}^{n} v_i\big) \geq \mathrm{L}_{T^*} + l_k \tag{10.522}$$

$$\geq \max\big(\mathrm{L}_{T^*}, C - T^*\big) \tag{10.523}$$

$$= \max\big(\sum_{i=1}^{n} v_i^* l_i, C - \sum_{i=1}^{n} v_i^*\big) \tag{10.524}$$

This contradicts the assumption in Eq.(10.519)

Case 3: If $T < T^*$, we obtain $C - T \geq C - T^* + 1$. Then we can achieve that

$$\max\big(\sum_{i=1}^{n} v_i^* l_i, C - \sum_{i=1}^{n} v_i^*\big) = \max\big(\mathrm{L}_{T^*}, C - T^*\big) \tag{10.525}$$

$$\leq C + 1 - T^* \tag{10.526}$$

$$\leq C - T \tag{10.527}$$

$$= C - \sum\nolimits_{i=1}^{n} v_i \tag{10.528}$$

$$\leq \max\big(\sum_{i=1}^{n} v_i l_i, C - \sum_{i=1}^{n} v_i\big). \tag{10.529}$$

This contradicts the assumption in Eq.(10.519).

Finally, we conclude that $\mathbf{v}^*$ obtained by Algorithm 13 is the minimum of the optimization problem given in (8.13). □

10.34 Proof of Proposition 2

Proof. Note that $T^* = \sum_{i=1}^{n} v_i^*$, from the condition of $v_i^* = 1$ in Algorithm 13, we know that $L_{T^*} \leq C + 1 - T^*$. From the condition of $v_k^* = 0$ in Algorithm 13, we know that $L_{T^*+1} > C - T^*$. Because $l(u_i) \geq \mathbf{1}\big(u_i < 0\big) \geq 0$ for $i \in \{1, ..., n\}$, we have $L_{T^*+1} = L_{T^*} + l(u_{T^*+1}) \geq L_{T^*}$. Thus, we obtain $L_{T^*+1} > \max(L_{T^*}, C - T^*)$. By substitute the optimum $\mathbf{v}^*$ into the optimization function, we obtain that

$$\min_{\mathbf{v}\in\{0,1\}^n} \max\big(\sum\nolimits_{i=1}^{n} v_i l(u_i), C - \sum\nolimits_{i=1}^{n} v_i\big) \tag{10.530}$$

$$= \max\big(\sum\nolimits_{i=1}^{n} v_i^* l(u_i), C - \sum\nolimits_{i=1}^{n} v_i^*\big) \tag{10.531}$$

$$= \max(L_{T^*}, C - T^*) \tag{10.532}$$

□

10.35 Proof of Theorem 30

Proof. We first prove that objective (8.11) is tighter than the loss objective $\widehat{J}(\mathbf{u})$ in Eq.(8.8). After this, we prove that objective (8.11) is an upper bound of the 0/1 loss defined in equation (8.7).

For simplicity, let $l_i = l(u_i)$, we obtain that

$$E\left(\mathbf{u}\right) = \min_{\mathbf{v}\in\{0,1\}^n} \max(\sum_{i=1}^{n} v_i l(u_i), n - \sum_{i=1}^{n} v_i) \tag{10.533}$$

$$\leq \max(\sum_{i=1}^{n} l(u_i), (n - \sum_{i=1}^{n} 1)) \tag{10.534}$$

$$= \sum_{i=1}^{n} l(u_i). \tag{10.535}$$

Note that $\widehat{J}\left(\mathbf{u}\right) = \sum_{i=1}^{n} l(u_i)$, thus, we have $E\left(\mathbf{u}\right) \leq \widehat{J}\left(\mathbf{u}\right)$.

Without loss of generality, assume $l_1 \leq l_2 \cdots \leq l_n$. Let $\mathrm{L}_i = \sum_{j=1}^{i} l_j$, $\mathrm{T} = \sum_{i=1}^{n} v_i^*$, where $\mathbf{v}^*{=}[v_1^*, v_2^* \cdots v_n^*]^T$ is the optimum of v for fixed $\mathbf{u}$. Let $k = \sum_{i=1}^{n} \mathbf{1}(u_i \geq 0)$. Then we achieve that the 0/1 loss $J(\mathbf{u})$ is as follows:

$$J(\mathbf{u}) = \sum_{i=1}^{n} \mathbf{1}(u_i < 0){=}n - k. \tag{10.536}$$

From Algorithm 1 with $C = n$, we achieve that $\mathrm{L_T} \leq n - T + 1$ and $\mathrm{L_{T+1}} > n - T$.

Case 1: If $k \geq T$, we can achieve that

$$2E\left(\mathbf{u}\right) - J(\mathbf{u}) = 2\max(\mathrm{L}_T, n - T) - (n - k) \tag{10.537}$$

$$\geq 2(n - T) - (n - k) \tag{10.538}$$

$$= n + k - 2T \geq 0.$$

Case 2: If $k < T, n - T \geq \mathrm{L}_T$, we can obtain that

$$2E\left(\mathbf{u}\right) - J(\mathbf{u}) = 2(n - T) - (n - k) = n + k - 2T. \tag{10.539}$$

Since $k < T$, if follows that

$$\mathrm{L_T} = \mathrm{L}_k + \sum_{j=k+1}^{T} l_j \geq \mathrm{L}_k + \sum_{j=k+1}^{T} 1 \tag{10.540}$$

$$= \mathrm{L}_k + T - k \tag{10.541}$$

$$\geq T - k. \tag{10.542}$$

Together with $n-T \geq \mathrm{L}_T$, we can obtain that

$$n-T \geq \mathrm{L}_T \geq T-k \Rightarrow n+k-2T \geq 0. \tag{10.543}$$

Thus, we can achieve that

$$2E\left(\mathbf{u}\right)-J(\mathbf{u})=n+k-2T \geq 0. \tag{10.544}$$

Case 3: If $k<T, n-T<\mathrm{L}_T$, we can obtain that

$$\begin{aligned} 2E\left(\mathbf{u}\right)-J(\mathbf{u}) &= 2\max(\mathrm{L}_T, n-T)-(n-k) && (10.545)\\ &= 2\mathrm{L}_T-(n-k) && (10.546)\\ &> (n-T)+\mathrm{L}_T-n+k. && (10.547) \end{aligned}$$

From (10.542), we have $\mathrm{L_T} \geq T-k$. Together with (10.547), it follows that

$$2E\left(\mathbf{u}\right)-J(\mathbf{u})>(n-T)+(T-k)-n+k \geq 0. \tag{10.548}$$

Finally, we can achieve that $J(\mathbf{u}) \leq 2E\left(\mathbf{u}\right) \leq 2\widehat{J}\left(\mathbf{u}\right)$. □

10.36 Proof of Corollary 5

Proof. Since $n=mb$, similar to the proof of $\widehat{Q}\left(\mathbf{u}\right) \leq \widehat{J}\left(\mathbf{u}\right)$, we have

$$\begin{aligned} \widehat{E}\left(\mathbf{u}\right) =& \sum_{j=1}^{b} \min_{\mathbf{v}\in\{0,1\}^m} \max\left(\sum_{i=1}^{m} v_{ij}l(u_{ij}), m-\sum_{i=1}^{m} v_{ij}\right)\\ \leq& \sum_{j=1}^{b} \max\left(\sum_{i=1}^{m} l(u_{ij}), m-\sum_{i=1}^{m} 1\right) && (10.549)\\ =& \sum_{j=1}^{b} \max\left(\sum_{i=1}^{m} l(u_{ij}),\ 0\ \right) && (10.550)\\ =& \sum_{j=1}^{b} \sum_{i=1}^{m} l(u_{ij}) = \widehat{J}\left(\mathbf{u}\right) && (10.551) \end{aligned}$$

On the other hand, since the group (batch) separable sum structure, we have that

$$\begin{aligned} \widehat{E}\left(\mathbf{u}\right) =& \sum_{j=1}^{b} \min_{\mathbf{v}\in\{0,1\}^m} \max\left(\sum_{i=1}^{m} v_{ij}l(u_{ij}), m-\sum_{i=1}^{m} v_{ij}\right)\\ =& \min_{\mathbf{v}\in\{0,1\}^n} \sum_{j=1}^{b} \max\left(\sum_{i=1}^{m} v_{ij}l(u_{ij}), m-\sum_{i=1}^{m} v_{ij}\right) && (10.552)\\ \geq& \min_{\mathbf{v}\in\{0,1\}^n} \max\left(\sum_{j=1}^{b}\sum_{i=1}^{m} v_{ij}l(u_{ij}), n-\sum_{j=1}^{b}\sum_{i=1}^{m} v_{ij}\right) && (10.553)\\ =& E\left(\mathbf{u}\right) && (10.554) \end{aligned}$$

Together with Theorem 30, we obtain that $J(\mathbf{u}) \leq 2E\left(\mathbf{u}\right) \leq 2\widehat{E}\left(\mathbf{u}\right) \leq 2\widehat{\mathrm{J}}\left(\mathbf{u}\right)$

□

10.37 Multi-Class Extension

For multi-class classification, denote the groudtruth label as $y \in \{1, ..., K\}$. Denote the classification prediction (the last layer output of networks before loss function) as $t_i, i \in \{1, ..., K\}$. Then, the classification margin for multi-class classification can be defined as follows

$$u = t_y - \max_{i \neq y} t_i. \tag{10.555}$$

We can see that $\mathbf{1}\big(u < 0\big) = \mathbf{1}\big(t_y - \max_{i \neq y} t_i < 0\big)$ is indeed the 0-1 loss for multi-class classification.

With the classification margin u, we can compute the base loss $l(u) \geq \mathbf{1}\big(u < 0\big)$. In this work, we employ the hinge loss. As we need the upper bound of 0-1 loss, the multi-class hard hinge loss function [127] can be defined as

$$H(\mathbf{t}, y) = \max(1 - u, 0) = \max(1 - t_y + \max_{i \neq y} t_i, 0). \tag{10.556}$$

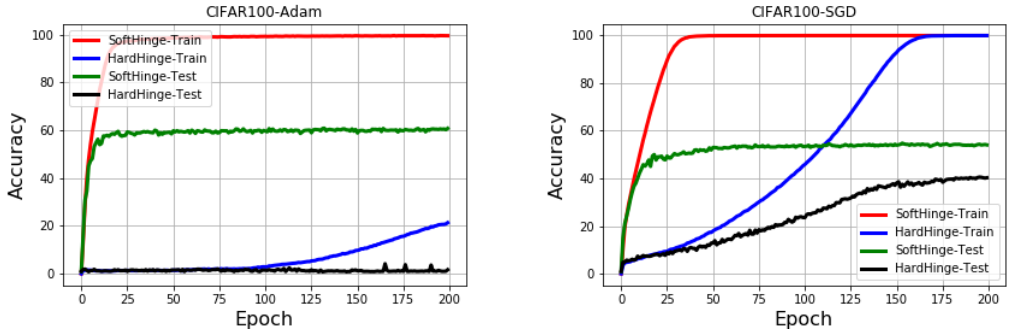
The multi-class hard hinge loss in Eq.(10.556) is not easy to optimize for deep networks. We propose a novel soft multi-class hinge loss function as follows:

$$S(\mathbf{t}, y) = \begin{cases} \max(1 - t_y + \max_{i \neq y} t_i, 0) & , \ t_y - \max_{i \neq y} t_i \geq 0 \\ \max(1 - t_y + \text{LogSumExp}(\mathbf{t}), 0) & , \ t_y - \max_{i \neq y} t_i < 0. \end{cases} \tag{10.557}$$

The soft hinge loss employs the LogSumExp function to approximate the max function when the classification margin is less than zero, i.e., misclassification case. Intuitively, when the sample is misclassified, it is far away from being correctly separate by a positive margin (e.g. margin $u \geq 1$). In this situation, a smooth loss function can help speed up gradient update. Because $\text{LogSumExp}(\mathbf{t}) > \max_{i \in \{1, \cdots K\}} t_i$ we know that the soft hinge loss is an upper bound of the hard hinge loss, i.e., $S(\mathbf{t}, y) \geq H(\mathbf{t}, y)$. Moreover, we can obtain a new weighted loss $F(\mathbf{t}, y; \beta) = \beta S(\mathbf{t}, y) + (1 - \beta) H(\mathbf{t}, y), \beta \in [0, 1]$ that is also an upper bound of 0-1 loss.

10.38 Evaluation of Efficiency of the Proposed Soft-hinge Loss

We compare our soft multi-class hinge loss with hard multi-class hinge loss [127] on CIFAR100 dataset training with Adam and SGD optimizer, respectively. We keep both the network architecture and hyperparameters same. We employ the default



(a) Training with Adam optimizer

(b) Training with SGD optimizer

Figure 10.1: Training/Test accuracy for soft and hard hinge loss with different optimizer on CIFAR100

learning rate and momentums of Adam optimizer in PyTorch toolbox, i.e. $lr = 10^{-3}, \beta_1 = 0.9, \beta_2 = 0.999$. For SGD optimizer, the learning rate (lr) and momentum (ρ) are set to $lr = 10^{-2}$ and $\rho = 0.9$ respectively.

The pictures of training/test accuracy v.s number of epochs are presented in Figure 10.1. We can observe that both the training accuracy and the test accuracy of our soft hinge loss increase greatly fast as the number of epochs increase. In contrast, the training and test accuracy of hard hinge loss grow very slowly. The training accuracy of soft hinge loss can arrive 100% trained with both optimizers. Both training and test accuracy of soft hinge loss are consistently better than hard hinge loss. In addition, training accuracy of hard hinge loss can also reach 100% when SGD optimizer is used. However, its test accuracy is lowever than that of soft hinge loss.

References

[1] Dirk Nuyens Adrian Ebert, Hernan Leövey. Successive coordinate search and component-by-component construction of rank-1 lattice rules. *arXiv preprint arXiv:1703.06334*, 2018.

[2] Youhei Akimoto and Nikolaus Hansen. Diagonal acceleration for covariance matrix adaptation evolution strategies. *Evolutionary computation*, pages 1–30, 2019.

[3] Youhei Akimoto, Yuichi Nagata, Isao Ono, and Shigenobu Kobayashi. Bidirectional relation between cma evolution strategies and natural evolution strategies. In *International Conference on Parallel Problem Solving from Nature*, pages 154–163. Springer, 2010.

[4] Zeyuan Allen-Zhu and Yuanzhi Li. What can resnet learn efficiently, going beyond kernels? *arXiv preprint arXiv:1905.10337*, 2019.

[5] Zeyuan Allen-Zhu and Yuanzhi Li. Backward feature correction: How deep learning performs deep learning. *arXiv preprint arXiv:2001.04413*, 2020.

[6] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

[7] Shun-ichi Amari. *Information geometry and its applications*, volume 194. Springer, 2016.

[8] Senjian An, Farid Boussaid, and Mohammed Bennamoun. How can deep rectifier networks achieve linear separability and preserve distances? In *ICML*, pages 514–523, 2015.

[9] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. *arXiv preprint arXiv:1904.11955*, 2019.

[10] Bouhari Arouna. Adaptative monte carlo method, a variance reduction technique. *Monte Carlo Methods and Applications*, 10(1):1–24, 2004.

[11] Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *ICML*, pages 233–242, 2017.

[12] Charles Audet and Warren Hare. *Derivative-free and blackbox optimization*. Springer, 2017.

[13] Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

[14] Haim Avron, Vikas Sindhwani, Jiyan Yang, and Michael W Mahoney. Quasi-monte carlo feature maps for shift-invariant kernels. *Journal of Machine Learning Research*, 17(120):1–38, 2016.

[15] Haim Avron, Vikas Sindhwani, Jiyan Yang, and Michael W Mahoney. Quasi-monte carlo feature maps for shift-invariant kernels. *The Journal of Machine Learning Research*, 17(1):4096–4133, 2016.

[16] Katy S Azoury and Manfred K Warmuth. Relative loss bounds for on-line density estimation with the exponential family of distributions. *Machine Learning*, 43(3):211–246, 2001.

[17] Thomas Back, Frank Hoffmeister, and Hans-Paul Schwefel. A survey of evolution strategies. In *Proceedings of the fourth international conference on genetic algorithms*, volume 2. Morgan Kaufmann Publishers San Mateo, CA, 1991.

[18] Ainesh Bakshi, Rajesh Jayaram, and David P Woodruff. Learning two layer rectified neural networks in polynomial time. In *Conference on Learning Theory*, pages 195–268. PMLR, 2019.

[19] Krishnakumar Balasubramanian and Saeed Ghadimi. Zeroth-order (non)-convex stochastic optimization via conditional gradient and gradient updates. In *Advances in Neural Information Processing Systems*, pages 3455–3464, 2018.

[20] Juan Cruz Barsce, Jorge A Palombarini, and Ernesto C Martínez. Towards autonomous reinforcement learning: Automatic setting of hyper-parameters using bayesian optimization. In *Computer Conference (CLEI), 2017 XLIII Latin American*, pages 1–9. IEEE, 2017.

[21] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[22] Matthew James Beal et al. *Variational algorithms for approximate Bayesian inference*. university of London London, 2003.

[23] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *ICML*, pages 41–48. ACM, 2009.

[24] Felix Berkenkamp, Angela P Schoellig, and Andreas Krause. No-regret bayesian optimization with unknown hyperparameters. *Journal of Machine Learning Research*, 20:50, 2019.

[25] Ilija Bogunovic, Jonathan Scarlett, Stefanie Jegelka, and Volkan Cevher. Adversarially robust optimization with gaussian processes. In *Advances in Neural Information Processing Systems*, pages 5765–5775, 2018.

[26] Ilija Bogunovic, Jonathan Scarlett, Andreas Krause, and Volkan Cevher. Truncated variance reduction: A unified approach to bayesian optimization and level-set estimation. In *Advances in Neural Information Processing Systems*, pages 1507–1515, 2016.

[27] Digvijay Boob and Guanghui Lan. Theoretical properties of the global optimizer of two layer neural network. *arXiv preprint arXiv:1710.11241*, 2017.

[28] J Brauchart, E Saff, I Sloan, and R Womersley. Qmc designs: optimal order quasi monte carlo integration schemes on the sphere. *Mathematics of computation*, 83(290):2821–2851, 2014.

[29] Johann S Brauchart and Peter J Grabner. Distributing many points on spheres: minimal energy and designs. *Journal of Complexity*, 31(3):293–326, 2015.

[30] Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in multi-armed bandits problems. In *International conference on Algorithmic learning theory*, pages 23–37. Springer, 2009.

[31] Alexander Buchholz, Florian Wenzel, and Stephan Mandt. Quasi-monte carlo variational inference. *arXiv preprint arXiv:1807.01604*, 2018.

[32] Adam D Bull. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research (JMLR)*, 12(Oct):2879–2904, 2011.

[33] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[34] Youngmin Cho and Lawrence K Saul. Kernel methods for deep learning. In *Advances in neural information processing systems*, pages 342–350, 2009.

[35] Youngmin Cho and Lawrence K. Saul. Kernel methods for deep learning. In *NIPS*, 2009.

[36] Krzysztof Choromanski, Aldo Pacchiano, Jack Parker-Holder, and Yunhao Tang. From complexity to simplicity: Adaptive es-active subspaces for blackbox optimization. *arXiv:1903.04268*, 2019.

[37] Krzysztof Choromanski, Aldo Pacchiano, Jack Parker-Holder, Yunhao Tang, and Vikas Sindhwani. From complexity to simplicity: Adaptive es-active subspaces for blackbox optimization, 2019.

[38] Krzysztof Choromanski, Mark Rowland, Vikas Sindhwani, Richard E Turner, and Adrian Weller. Structured evolution with compact architectures for scalable policy optimization. In *ICML*, pages 969–977, 2018.

[39] Krzysztof Choromanski and Vikas Sindhwani. Recycling randomness with structure for sublinear time kernel expansions. 2016.

[40] Emile Contal, David Buffoni, Alexandre Robicquet, and Nicolas Vayatis. Parallel gaussian process optimization with upper confidence bound and pure exploration. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 225–240. Springer, 2013.

[41] Kurt Cutajar, Edwin V Bonilla, Pietro Michiardi, and Maurizio Filippone. Practical learning of deep gaussian processes via random fourier features. *arXiv preprint arXiv:1610.04386*, 2016.

[42] Sabrina Dammertz and Alexander Keller. Image synthesis by rank-1 lattices. In *Monte Carlo and Quasi-Monte Carlo Methods 2006*, pages 217–236. Springer, 2008.

[43] Sabrina Dammertz and Alexander Keller. Image synthesis by rank-1 lattices. In *Monte Carlo and Quasi-Monte Carlo Methods 2006*, 2008.

[44] Amit Daniely, Roy Frostig, and Yoram Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. *arXiv preprint arXiv:1602.05897*, 2016.

[45] Thomas Desautels, Andreas Krause, and Joel W Burdick. Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. *Journal of Machine Learning Research*, 15(1):3873–3923, 2014.

[46] Josef Dick, Frances Y Kuo, and Ian H Sloan. High-dimensional integration: the quasi-monte carlo way. *Acta Numerica*, 22:133–288, 2013.

[47] Josef Dick, Frances Y Kuo, and Ian H Sloan. High-dimensional integration: the quasi-monte carlo way. *Acta Numerica*, 22:133–288, 2013.

[48] Carola Doerr and François-Michel De Rainville. Constructing low star discrepancy point sets with genetic algorithms. In *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, pages 789–796, 2013.

[49] Justin Domke. Provable smoothness guarantees for black-box variational inference. *arXiv preprint arXiv:1901.08431*, 2019.

[50] David L Donoho, Iain M Johnstone, Jeffrey C Hoch, and Alan S Stern. Maximum entropy and the nearly black object. *Journal of the Royal Statistical Society: Series B (Methodological)*, 54(1):41–67, 1992.

[51] David L Donoho and Jain M Johnstone. Ideal spatial adaptation by wavelet shrinkage. *biometrika*, 81(3):425–455, 1994.

[52] Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018.

[53] David Steven Dummit and Richard M Foote. *Abstract algebra*, volume 3. Wiley Hoboken, 2004.

[54] Ronen Eldan and Ohad Shamir. The power of depth for feedforward neural networks. In *Conference on learning theory*, pages 907–940. PMLR, 2016.

[55] Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

[56] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874, 2008.

[57] Chang Feng, Qinghua Hu, and Shizhong Liao. Random feature mapping with signed circulant matrix projection. In *IJCAI*, pages 3490–3496, 2015.

[58] Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. Linearized two-layers neural networks in high dimension. *The Annals of Statistics*, 49(2):1029–1054, 2021.

[59] Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. In *ICLR*, 2017.

[60] Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2916–2929, 2013.

[61] Javier González, Zhenwen Dai, Philipp Hennig, and Neil Lawrence. Batch bayesian optimization via local penalization. In *Artificial intelligence and statistics*, pages 648–657, 2016.

[62] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *ICLR*, 2015.

[63] Mario Götz. On the riesz energy of measures. *Journal of Approximation Theory*, 122(1):62–78, 2003.

[64] Leonhard Grünschloß, Johannes Hanika, Ronnie Schwede, and Alexander Keller. (t, m, s)-nets and maximized minimum distance. In *Monte Carlo and Quasi-Monte Carlo Methods 2006*, pages 397–412. Springer, 2008.

[65] John Hammersley. *Monte carlo methods*. Springer Science & Business Media, 2013.

[66] Bo Han, Jiangchao Yao, Gang Niu, Mingyuan Zhou, Ivor Tsang, Ya Zhang, and Masashi Sugiyama. Masking: A new perspective of noisy supervision. In *Advances in Neural Information Processing Systems*, pages 5836–5846, 2018.

[67] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems*, pages 8527–8537, 2018.

[68] Boris Hanin and Mihai Nica. Finite depth and width corrections to the neural tangent kernel. *arXiv preprint arXiv:1909.05989*, 2019.

[69] Nikolaus Hansen. The cma evolution strategy: a comparing review. In *Towards a new evolutionary computation*, pages 75–102. Springer, 2006.

[70] Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary computation*, 11(1):1–18, 2003.

[71] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[72] Kyle Helfrich, Devin Willmott, and Qiang Ye. Orthogonal recurrent neural networks with scaled cayley transform. In *International Conference on Machine Learning*, pages 1969–1978. PMLR, 2018.

[73] Philipp Hennig and Christian J Schuler. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 13(Jun):1809–1837, 2012.

[74] José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *NIPS*, pages 918–926, 2014.

[75] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[76] Mengqiu Hu, Yang Yang, Fumin Shen, Luming Zhang, Heng Tao Shen, and Xuelong Li. Robust web image annotation via exploring multi-facet and structural knowledge. *IEEE Transactions on Image Processing*, 26(10):4871–4884, 2017.

[77] Weihua Hu, Gang Niu, Issei Sato, and Masashi Sugiyama. Does distributionally robust supervised learning give robust classifiers? 2018.

[78] L-K Hua and Yuan Wang. *Applications of number theory to numerical analysis*. Springer Science & Business Media, 2012.

[79] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[80] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *arXiv preprint arXiv:1806.07572*, 2018.

[81] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128, 2010.

[82] Lu Jiang, Deyu Meng, Shoou-I Yu, Zhenzhong Lan, Shiguang Shan, and Alexander Hauptmann. Self-paced learning with diversity. In *Advances in Neural Information Processing Systems*, pages 2078–2086, 2014.

[83] Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. Self-paced curriculum learning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[84] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. 2018.

[85] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

[86] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

[87] Motonobu Kanagawa, Philipp Hennig, Dino Sejdinovic, and Bharath K Sriperumbudur. Gaussian processes and kernel methods: A review on connections and equivalences. *arXiv preprint arXiv:1807.02582*, 2018.

[88] Kenji Kawaguchi. Deep learning without poor local minima. *Advances in Neural Information Processing Systems*, 2016.

[89] Alexander Keller, Stefan Heinrich, and Harald Niederreiter. *Monte Carlo and Quasi-Monte Carlo Methods 2006*. Springer, 2007.

[90] Mohammad Emtiyaz Khan and Wu Lin. Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. *arXiv preprint arXiv:1703.04265*, 2017.

[91] Mohammad Emtiyaz Khan and Didrik Nielsen. Fast yet simple natural-gradient descent for variational inference in complex models. In *2018 International Symposium on Information Theory and Its Applications (ISITA)*, pages 31–35. IEEE, 2018.

[92] Mohammad Emtiyaz Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable bayesian deep learning by weight-perturbation in adam. In *ICML*, 2018.

[93] Ashish Khetan, Zachary C Lipton, and Anima Anandkumar. Learning from noisy singly-labeled data. *ICLR*, 2018.

[94] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[95] AN Korobov. The approximate computation of multiple integrals. In *Dokl. Akad. Nauk SSSR*, volume 124, pages 1207–1210, 1959.

[96] N. M. Korobov. Properties and calculation of optimal coefficients. *Dokl. Akad. Nauk SSSR*, 132:1009–1012, 1960.

[97] Nikolai Mikhailovich Korobov. Properties and calculation of optimal coefficients. In *Doklady Akademii Nauk*, volume 132, pages 1009–1012. Russian Academy of Sciences, 1960.

[98] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[99] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[100] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[101] M Pawan Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*, pages 1189–1197, 2010.

[102] Harold J Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86(1):97–106, 1964.

[103] Helene Laimer. On combined component-by-component constructions of lattice point sets. *Journal of Complexity*, 38:22–30, 2017.

[104] Jeffrey Larson, Matt Menickelly, and Stefan M. Wild. Derivative-free optimization methods. *arXiv:1904.11585*, 2019.

[105] Quoc Le, Tamás Sarlós, and Alex Smola. Fastfood-approximating kernel expansions in loglinear time. In *Proceedings of the international conference on machine learning*, 2013.

[106] Zichao Yang Alexander J Smola Le and Song Andrew Gordon Wilson. A la carte—learning fast kernels. 38, 2015.

[107] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[108] Pierre L'ecuyer and David Munger. Algorithm 958: Lattice builder: A general software tool for constructing rank-1 lattice rules. *ACM Transactions on Mathematical Software (TOMS)*, 42(2):1–30, 2016.

[109] Jaehoon Lee, Lechao Xiao, Samuel S Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *Neural Information Processing Systems*, 2019.

[110] Kimin Lee, Sukmin Yun, Kibok Lee, Honglak Lee, Bo Li, and Jinwoo Shin. Robust inference via generative classifiers for handling noisy labels. In *International Conference on Machine Learning*, 2019.

[111] Gunther Leobacher and Friedrich Pillichshammer. *Introduction to quasi-Monte Carlo integration and applications*. Springer, 2014.

[112] Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, 6(6):861–867, 1993.

[113] Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. *arXiv preprint arXiv:1808.01204*, 2018.

[114] Guoqing Liu, Li Zhao, Feidiao Yang, Jiang Bian, Tao Qin, Nenghai Yu, and Tie-Yan Liu. Trust region evolution strategies. In *AAAI*, 2019.

[115] Daniel J Lizotte, Tao Wang, Michael H Bowling, and Dale Schuurmans. Automatic gait optimization with gaussian process regression. In *IJCAI*, volume 7, pages 944–949, 2007.

[116] James N Lyness. Notes on lattice rules. *Journal of Complexity*, 19(3):321–331, 2003.

[117] Yueming Lyu. Spherical structured feature maps for kernel approximation. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2256–2264. JMLR. org, 2017.

[118] Yueming Lyu, Yuan Yuan, and Ivor W Tsang. Efficient batch black-box optimization with deterministic regret bounds. *arXiv preprint arXiv:1905.10041*, 2019.

[119] Yueming Lyu, Yuan Yuan, and Ivor W Tsang. Subgroup-based rank-1 lattice quasi-monte carlo. In *NeurIPS*, 2020.

[120] Xingjun Ma, Yisen Wang, Michael E Houle, Shuo Zhou, Sarah M Erfani, Shu-Tao Xia, Sudanthi Wijewickrema, and James Bailey. Dimensionality-driven learning with noisy labels. In *International Conference on Machine Learning*, 2018.

[121] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

[122] Hamed Masnadi-Shirazi and Nuno Vasconcelos. On the design of loss functions for classification: theory, robustness to outliers, and savageboost. In *Advances in neural information processing systems*, pages 1049–1056, 2009.

[123] Takeru Miyato, Andrew M Dai, and Ian Goodfellow. Virtual adversarial training for semi-supervised text classification. In *ICLR*, 2016.

[124] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[125] Jonas Močkus. On bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference*, pages 400–404. Springer, 1975.

[126] Jonas Močkus. On bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference*, pages 400–404. Springer, 1975.

[127] Robert Moore and John DeNero. L1 and l2 regularization for multiclass hinge loss models. In *Symposium on Machine Learning in Speech and Language Processing*, 2011.

[128] Diana M Negoescu, Peter I Frazier, and Warren B Powell. The knowledge-gradient algorithm for sequencing experiments in drug discovery. *INFORMS Journal on Computing*, 23(3):346–363, 2011.

[129] Yurii Nesterov and Vladimir Spokoiny. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17(2):527–566, 2017.

[130] Harald Niederreiter. *Random number generation and quasi-Monte Carlo methods*, volume 63. Siam, 1992.

[131] Frank Nielsen and Vincent Garcia. Statistical exponential families: A digest with flash cards. *Arxiv*, abs/0911.4863, 2009.

[132] Atsushi Nitanda and Taiji Suzuki. Optimal rates for averaged stochastic gradient descent under neural tangent kernel regime. *ICLR*, 2021.

[133] Dirk Nuyens and Ronald Cools. Fast algorithms for component-by-component construction of rank-1 lattice rules in shift-invariant reproducing kernel hilbert spaces. *Mathematics of Computation*, 75(254):903–920, 2006.

[134] Junier B Oliva, Avinava Dubey, Barnabas Poczos, Jeff Schneider, and Eric P Xing. Bayesian nonparametric kernel-learning. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 1078–1086, 2016.

[135] Yann Ollivier, Ludovic Arnold, Anne Auger, and Nikolaus Hansen. Information-geometric optimization algorithms: A unifying picture via invariance principles. *The Journal of Machine Learning Research (JMLR)*, 18(1):564–628, 2017.

[136] Carl Olsson, Marcus Carlsson, Fredrik Andersson, and Viktor Larsson. Non-convex rank/sparsity regularization and local minima. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 332–340, 2017.

[137] Art B Owen. Monte carlo book: the quasi-monte carlo parts. 2019.

[138] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1944–1952, 2017.

[139] Farhad Pourkamali-Anaraki, Stephen Becker, and Michael B Wakin. Randomized clustered nystrom for large-scale kernel machines. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[140] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, 2007.

[141] Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in neural information processing systems*, pages 1313–1320, 2009.

[142] Ali Rahimi, Benjamin Recht, et al. Random features for large-scale kernel machines. In *NIPS*, volume 3, page 5, 2007.

[143] Garvesh Raskutti and Sayan Mukherjee. The information geometry of mirror descent. *IEEE Transactions on Information Theory*, 61(3):1451–1457, 2015.

[144] Carl Edward Rasmussen. Gaussian processes for machine learning. 2006.

[145] Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. Data programming: Creating large training sets, quickly. In *Advances in neural information processing systems*, pages 3567–3575, 2016.

[146] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014.

[147] Luis Miguel Rios and Nikolaos V Sahinidis. Derivative-free optimization: a review of algorithms and comparison of software implementations. *Journal of Global Optimization*, 56(3):1247–1293, 2013.

[148] Walter Rudin. *Fourier analysis on groups*. John Wiley & Sons, 2011.

[149] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

[150] Jonathan Scarlett. Tight regret bounds for bayesian optimization in one dimension. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 4500–4508, 2018.

[151] Peter H Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, 1966.

[152] Amar Shah and Zoubin Ghahramani. Parallel predictive entropy search for batch global optimization of expensive objective functions. In *NIPS*, pages 3330–3338, 2015.

[153] Vaishaal Shankar, Alex Fang, Wenshuo Guo, Sara Fridovich-Keil, Jonathan Ragan-Kelley, Ludwig Schmidt, and Benjamin Recht. Neural kernels without tangents. In *International Conference on Machine Learning*, pages 8614–8623. PMLR, 2020.

[154] I Sloan and A Reztsov. Component-by-component construction of good lattice rules. *Mathematics of Computation*, 71(237):263–273, 2002.

[155] Ian H Sloan and Henryk Woźniakowski. Tractability of multivariate integration for weighted korobov classes. *Journal of Complexity*, 17(4):697–721, 2001.

[156] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *NeurIPS*, pages 2951–2959, 2012.

[157] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

[158] Mandavilli Srinivas and Lalit M Patnaik. Genetic algorithms: A survey. *computer*, 27(6):17–26, 1994.

[159] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

[160] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *ICML*, 2010.

[161] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. *NeurIPS*, 2015.

[162] Hao Su, Jia Deng, and Li Fei-Fei. Crowdsourcing annotations for visual object detection. In *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[163] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014.

[164] Dougal J Sutherland and Jeff Schneider. On the error of random fourier features. *arXiv preprint arXiv:1506.02785*, 2015.

[165] Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5552–5560, 2018.

[166] Anthony Tompkins, Ransalu Senanayake, Philippe Morere, and Fabio Ramos. Black box quantiles for kernel learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1427–1437, 2019.

[167] Brendan Van Rooyen, Aditya Menon, and Robert C Williamson. Learning with symmetric label noise: The importance of being unhinged. In *Advances in Neural Information Processing Systems*, pages 10–18, 2015.

[168] G Gary Wang and Songqing Shan. Review of metamodeling techniques in support of engineering design optimization. *Journal of Mechanical design*, 129(4):370–380, 2007.

[169] Zi Wang, Clement Gehring, Pushmeet Kohli, and Stefanie Jegelka. Batched large-scale bayesian optimization in high-dimensional spaces. In *Artificial intelligence and statistics*, 2018.

[170] Zi Wang and Stefanie Jegelka. Max-value entropy search for efficient bayesian optimization. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3627–3635. JMLR. org, 2017.

[171] Zi Wang and Stefanie Jegelka. Max-value entropy search for efficient bayesian optimization. In *International Conference on Machine Learning (ICML)*, page 3627–3635, 2017.

[172] Holger Wendland. *Scattered data approximation*, volume 17. Cambridge university press, 2004.

[173] Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. Natural evolution strategies. *The Journal of Machine Learning Research (JMLR)*, 15(1):949–980, 2014.

[174] Aaron Wilson, Alan Fern, and Prasad Tadepalli. Using trajectory data to improve bayesian optimization for reinforcement learning. *The Journal of Machine Learning Research*, 15(1):253–282, 2014.

[175] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[176] Jian Wu and Peter Frazier. The parallel knowledge gradient method for batch bayesian optimization. In *NIPS*, pages 3126–3134, 2016.

[177] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 501–509, 2019.

[178] J. Yang, Sindhwani. V., H. Avron, and M. Mahoney. Quasi-monte carlo feature maps for shift-invariant kernels. *ICML*, 32, 2014.

[179] Jiyan Yang, Vikas Sindhwani, Haim Avron, and Michael Mahoney. Quasi-monte carlo feature maps for shift-invariant kernels. In *International Conference on Machine Learning*, pages 485–493, 2014.

[180] Gilad Yehudai and Ohad Shamir. On the power and limitations of random features for understanding neural networks. *arXiv preprint arXiv:1904.00687*, 2019.

[181] Felix Xinnan X Yu, Ananda Theertha Suresh, Krzysztof M Choromanski, Daniel N Holtmann-Rice, and Sanjiv Kumar. Orthogonal random features. In *Advances in Neural Information Processing Systems*, pages 1975–1983, 2016.

[182] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *International Conference on Machine Learning*, pages 7164–7173, 2019.

[183] Yuan Yuan, Yueming Lyu, Xi Shen, Ivor W. Tsang, and Dit-Yan Yeung. Marginalized average attentional network for weakly-supervised learning. In *ICLR*, 2019.

[184] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[185] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *ICLR*, 2017.

[186] Cun-Hui Zhang et al. Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics*, 38(2):894–942, 2010.

[187] Tong Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *Journal of Machine Learning Research*, 11(3), 2010.

[188] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in neural information processing systems*, pages 8778–8788, 2018.

[189] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[190] Difan Zou and Quanquan Gu. An improved analysis of training over-parameterized deep neural networks. *Advances in Neural Information Processing Systems*, 2019.