Topics in Deep Learning Theory (Spring 2025)

Lecture 5: Learning RKHS with SGD

Instructor: Lei Wu Date: April 20, 2025

Abstract

In this lecture, we explore the random feature (RF) approximation of KRR, introducing a Monte Carlo perspective to understand high-dimensional function learning. Additionally, we analyze how stochastic gradient descent (SGD) performs in learning RKHS functions, discussing its optimality and limitations, as well as the phenomenon known as *spectral bias*. We stress, in this lecture, we choose to examine classical RKHS function learning from perspectives that are applicable to nonlinear neural networks, establishing a connection between classical learning theory and deep learning.

1 Introduction

An important implication of Lecture 4 is the **sample complexity** of kernel ridge regression (KRR), which refers to the number of samples necessary to achieve a specified generalization error. Specifically, for a reproducing kernel Hilbert space (RKHS) endowed with the kernel k with a power-law eigenvalue decay, $\lambda_j \asymp j^{-\beta}$ under the underlying distribution ρ, we obtain the following bound for the generalization error for $f^* \in \mathcal{H}_k^s$ as

$$\|\hat{f}_\lambda - f^*\|_{L^2(\rho)}^2 \lesssim n^{-\frac{\min(s,2)\beta}{\min(s,2)\beta+1}}. \tag{1}$$

WLOG, assuming $s \in [0, 2]$. Then, to achieve a precision of ϵ, the number of samples we need is at least

$$n_\epsilon \geq \left(\frac{1}{\epsilon}\right)^{\frac{s\beta+1}{s\beta}}.$$

Another significant aspect to consider is the **time complexity**, which represents the computational time required to complete a task. Discussions of time complexity typically include the following categories:

- **Information-theoretic:** The information-theoretic lower bound on time complexity refers to the minimal time required to solve a given task, irrespective of the algorithm employed. For instance, sorting a sequence of length n necessitates at least $O(n \log n)$ comparisons.
- **Commonly used algorithms:** Alternatively, one can study the time complexity of solving the given task using a specific learning algorithm (e.g., KRR or stochastic gradient descent (SGD)).

[0]Special thanks to Yuhao Liu and Zilin Wang for scribing the notes.

In this lecture, we primarily focus on the time complexity of learning RKHS functions. To begin, we provide a naive estimation of the time complexity for KRR. By the Representer theorem, the KRR estimator can be expressed as

$$\hat{f}_\lambda = \sum_{j=1}^{n} (\hat{a}_\lambda)_j k(x_j, \cdot)$$

where

$$\hat{a}_\lambda = (K + n\lambda I)^{-1} y \tag{2}$$

Computing the inverse of an $n \times n$ matrix typically requires $O(n^3)$ operations. Consequently, the time complexity is bounded by

$$C_\epsilon^{\text{krr}} \lesssim n_\epsilon^3 = \left(\frac{1}{\epsilon}\right)^{\frac{3(s\beta+1)}{s\beta}}. \tag{3}$$

Note that $s\beta$ represents the inherent smoothness of f^*.

Remark 1.1 (Periodic Sobolev spaces)**.** *Suppose $f^* \in H^\gamma(\mathbb{T})$, which is defined as*

$$\|f\|_{H^\gamma(\mathbb{T})}^2 = \sum_{m\in\mathbb{Z}} (1 + |m|^2)^\gamma |\hat{f}(m)|^2,$$

where $\{\hat{f}(m)\}_{m\in\mathbb{Z}}$ denotes the Fourier coefficients of f: $f^* = \sum_{m\in\mathbb{Z}} \hat{f}(m) e_j$ where $e_m(x) := e^{2\pi i m x}$ is the Fourier basis functions.

This regularity condition $\|f^*\|_{H^\gamma(\mathbb{T})} < \infty$ implies that $\hat{f}(m) \asymp m^{-\gamma-\frac{1}{2}}$. To ensure f^* lies in $\mathcal{H}_k^s$, we require

$$m^{-\gamma-\frac{1}{2}} \asymp m^{-\frac{1}{2}-\frac{s\beta}{2}},$$

which gives $2\gamma = s\beta$. Under these assumptions, the resulting time complexity can be bounded by Equation (3) as

$$C_\epsilon^{\text{krr}} \lesssim \left(\frac{1}{\epsilon}\right)^{3+\frac{3}{2\gamma}},$$

indicating that the time complexity depends directly on the inherent smoothness of f^*.

2 Preliminaries

Equation (3) provides an upper bound on the time complexity when $\hat{f}_\lambda$ is computed analytically by inverting the regularized Gram matrix (see Equation (2)). In practice, however, one often prefer gradient-based algorithms and expect improved time complexity.

2.1 The model and target functions

Firstly, let us translate the KRR into a linear regression problem. Assume the kernel k admits the following decomposition

$$k(x, x') = \sum_{j=1}^{\infty} \lambda_j e_j(x) e_j(x'),$$

where $\{e_j\}_{j=1}^{\infty}$ forms an orthogonal basis for $L^2(\rho)$. The model is defined as

$$f_\theta(x) = \varphi(x)^\top \theta \quad \text{where} \quad \varphi(x) = (\sqrt{\lambda_1} e_1(x), \cdots, \sqrt{\lambda_p} e_p(x))^\top,$$

and let $p = \infty$ in principle. Then,

$$\Sigma = \mathbb{E}_x[\varphi(x)\varphi(x)^\top] = \mathrm{diag}(\lambda_1, \ldots, \lambda_p). \tag{4}$$

Define

$$\mathcal{R}(\theta) = \frac{1}{2}\mathbb{E}_{x,y}\left[\left(\varphi(x)^\top \theta - y\right)^2\right].$$

Suppose that $x \sim \rho$ and $y = \varphi(x)^\top \theta^* + \xi$, where the noise ξ is assumed to satisfy $\mathbb{E}\,\xi = 0$ and $\mathbb{E}\,\xi^2 = \sigma^2$. Then, we have the decomposition:

$$\mathcal{R}(\theta) = \mathcal{E}(\theta) + \frac{\sigma^2}{2}, \tag{5}$$

where $\mathcal{E}(\cdot)$ is the excess risk given by

$$\mathcal{E}(\theta) = \frac{1}{2}\mathbb{E}_x\left[(f_\theta(x) - f^*(x))^2\right]. \tag{6}$$

Suppose $f^*(x) = \varphi(x)^\top \theta^*$ for some θ^*, the excess risk is given by

$$\mathcal{E}(\theta) = \frac{1}{2}(\theta - \theta^*)^\top \Sigma (\theta - \theta^*) = \frac{1}{2}\|\theta - \theta^*\|_\Sigma^2. \tag{7}$$

For target function we make the following assumption:

Assumption 2.1 (Source condition)**.** *Suppose $f^* \in \mathcal{H}^s$. This is equivalent to assume $\theta^* = \left(a_j^* \lambda_j^{(s-1)/2}\right)_{j\geq 1}$ with $a \in \ell^2$.*

Here, the index s denotes the smoothness of f^* relative to the model space $\mathcal{H}_k$. For the model, we make the power-law decay assumption:

Assumption 2.2 (Capacity condition)**.** *Suppose $\lambda_j \asymp j^{-\beta}$ for some $\beta > 1$.*

2.2 Stochastic Gradient Descent

We consider the algorithm of online stochastic gradient descent (SGD) to minimize $\mathcal{R}(\cdot)$. At the t-th iteration, SGD draws a batch of i.i.d. samples (denoted as S_t) and updates the θ as follows

$$\begin{aligned}
\theta_{t+1} &= \theta_t - \eta_t g_t \\
&= \theta_t - \eta_t \nabla_\theta \left(\frac{1}{2}\sum_{(x,y)\in S_t} (\varphi(x)^\top \theta_t - y)^2\right) \\
&= \theta_t - \frac{\eta_t}{|S_t|}\sum_{(x,y)\in S_t} \varphi(x)(\varphi(x)^\top \theta_t - y),
\end{aligned} \tag{8}$$

where $g_t \in \mathbb{R}^p$ denotes the minibatch gradient and η_t is the learning rate (LR). Substituting $y = \varphi(x)^\top \theta^* + \xi$ into (8) yields

$$\begin{aligned} g_t &= \left(\frac{1}{|S_t|} \sum_{(x,y)\in S_t} \varphi(x)\varphi(x)^\top \right) (\theta_t - \theta^*) + \frac{\eta_t}{|S_t|} \sum_{(x,y)\in S_t} \varphi(x)\xi \\ &= \widehat{\Sigma}_t(\theta_t - \theta^*) + \frac{\eta_t}{|S_t|} \sum_{(x,y)\in S_t} \varphi(x)\xi, \end{aligned} \tag{9}$$

where

$$\hat{\Sigma}_t = \frac{1}{|S_t|} \sum_{(x,y)\in S_t} \varphi(x)\varphi(x)^\top.$$

For better understanding and analysis, we often rewrite the SGD iteration as:

$$\theta_{t+1} = \theta_t - \eta_t g_t = \theta_t - \eta_t(\mathbb{E}[g_t] + \epsilon_t),$$

where $\mathbb{E}[g_t] = \nabla\mathcal{R}(\theta_t)$ is the full-batch gradient and $\epsilon_t = g_t - \mathbb{E}[g_t]$ is the gradient noise. Specifically, we have

$$\nabla\mathcal{R}(\theta_t) = \Sigma(\theta_t - \theta^*)$$

and

$$\epsilon_t = (\widehat{\Sigma}_t - \Sigma)(\theta_t - \theta^*) + \frac{1}{|S_t|} \sum_{(x,y)\in S_t} \varphi(x)\xi.$$

Note that there are two distinct sources of gradient noise:

- **Label noise** ξ, which directly perturbs the observed outputs.
- **Estimation noise** arising from the discrepancy between the empirical covariance $\hat{\Sigma}_t$ and the true covariance Σ. This source of noise persists even in the *noiseless* setting where $\sigma = 0$.

We shall make the following assumption for the SGD.

Assumption 2.3 (Batch size)**.** *Assume $|S_t| = B$ for all $t \in \mathbb{N}$.*

Assumption 2.4 (Initialization)**.** *We shall focus on the zero initialization: $\theta_0 = 0$.*

3 Full-batch Gradient Descent

We first consider the regime with $|S_t| = \infty$ and $\eta_t = \eta$ (constant learning rate). Then, the SGD iteration reduces to the full-batch GD:

$$\theta_{t+1} = \theta_t - \eta\Sigma(\theta_t - \theta^*). \tag{10}$$

For the simplicity of analysis, we further assume that the learning rate η is infinitesimally small ($\eta \to 0$). In this regime, GD reduces to the gradient flow (GF):

$$\frac{\mathrm{d}}{\mathrm{d}t}\theta_t = \Sigma(\theta_t - \theta^*).$$

Let $\delta(t) = \theta - \theta^*$. Then, GF reduces to

$$\dot{\delta}(t) = -\Sigma\delta(t), \tag{11}$$

with the initialization

$$\delta(0) = -\theta^*.$$

Moreover, the excess risk becomes

$$\mathcal{E}(\theta_t) = \|\delta(t)\|_\Sigma^2$$

Because $\{e_j\}_{j=1}^\infty$ constitutes an orthogonal basis of $L^2(\rho)$, the matrix Σ is diagonalizable, with

$$\Sigma = \text{diag}(\lambda_1, \cdots, \lambda_p)$$

Hence, the solution to Equation (11) can be written component-wise as

$$\delta_j(t) = \delta_j(0)e^{-\lambda_j t} \quad \text{for } j = 1, 2, \ldots, p. \tag{12}$$

The fitting error evolution. Using the solution $\delta_j(t)$ from (12), we have

$$f_{\theta_t} - f^* = \sum_{j=1}^p \delta_j(t)\lambda_j^{1/2} e_j = \sum_{j=1}^p \delta_j(0)\lambda_j^{1/2} e^{-\lambda_j t} e_j. \tag{13}$$

As $t \to \infty$, we see that $f_{\theta_t} \to f^*$.

The following lemma further characterizes the evolution of the average generalization error:

Lemma 3.1. $\mathcal{E}_t := \mathcal{E}(\theta_t) = \sum_{j=1}^p |\theta_j^*|^2 \lambda_j e^{-2\lambda_j t}$.

Proof. Note that

$$\mathcal{E}(\theta_t) = \|\theta_t - \theta^*\|_\Sigma^2 = \sum_{j=1}^p \lambda_j \delta_j^2(t).$$

Since $\delta_j(t) = -\theta_j^* e^{-\lambda_j t}$, we obtain $\mathcal{E}_t = \sum_{j=1}^p \lambda_j (\theta_j^*)^2 e^{-2\lambda_j t}$. □

By lemma 3.1, the convergence for the j-th component is governed by both the coefficient θ_j^* and the eigenvalue λ_j. Specifically, components with smaller $|\theta_j^*|^2$ and larger λ_j converge more rapidly.

3.1 Convergence Rate

By Lemma 3.1, when the dimension p is finite (assuming $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$), the error $\mathcal{E}(\theta_t) = \mathcal{E}_t$ decays exponentially as

$$\mathcal{E}_t = O\big(e^{-\lambda_p t}\big)$$

in the limit $t \to \infty$. However, this pure exponential convergence breaks down in the infinite-dimensional setting ($p \to \infty$), or more generally whenever the time horizon t is small relative to p.

Indeed, by definition we have

$$\mathcal{E}_t \;=\; \sum_{j=1}^p \big|\theta_j^*\big|^2 \,\lambda_j\, e^{-2\lambda_j t}. \tag{14}$$

As $p \to \infty$, the shape of the decay curve is determined by the two sequences

$$(\theta_j^*)_{j\geq 1} \quad \text{(target coefficients)}, \qquad (\lambda_j)_{j\geq 1} \quad \text{(eigenvalues)}.$$

Intuitively, (14) acts like a discrete analogue of an inverse Laplace transform in t, so different asymptotic regimes of (θ_j^*) and (λ_j) can lead to qualitatively distinct convergence behaviors.

Proposition 3.2 (Convergence of GD)**.** *Under Assumptions 2.2 and 2.1, we have*

$$\mathcal{E}_t \lesssim t^{-s}.$$

Proof. By Lemma 3.1 and Assumption 2.1, we obtain

$$\begin{aligned}
\mathcal{E}_t &= \sum_{j=1}^{p} \lambda_j \lambda_j^{s-1} (a_j^*)^2 e^{-2\lambda_j t} \\
&= \sum_{j=1}^{p} (a_j^*)^2 \lambda_j^s e^{-2\lambda_j t} \\
&\lesssim \sum_{j=1}^{p} j^{-(s\beta+1)} e^{-\frac{2t}{j^\beta}} \lesssim \int_1^\infty h(z)\mathrm{d}z,
\end{aligned}$$

where we define

$$h(z) = z^{-(s\beta+1)} e^{-\frac{2t}{z^\beta}}$$

The integration can be bounded by

$$\begin{aligned}
\int_1^\infty h(z) &= \int_1^\infty z^{-(s\beta+1)} e^{-\frac{2t}{z^\beta}} \mathrm{d}z \\
&\overset{(i)}{=} \frac{1}{\beta} \int_0^1 u^{s-1} e^{-2tu} \mathrm{d}u \\
&\overset{(ii)}{=} \frac{1}{2^s \beta t^s} \int_0^{2t} v^{s-1} e^{-v} \mathrm{d}v \lesssim \frac{1}{t^s},
\end{aligned}$$

where (i) we use the variable substitution $u = z^{-\beta}$ and (ii) we use the variable substitution $v = 2tu$. This completes the proof. □

Remark 3.3. *Under the structural assumption $\theta_j^* = a_j^* \lambda_j^{\frac{s-1}{2}}$, the error decays at the rate $O(t^{-s})$, improving upon the classical $O(t^{-1})$ guarantee from standard convex analysis. Moreover, increased smoothness of the target function f^* (i.e., larger s) leads to faster convergence.*

This example shows that even in linear regression—a mere quadratic optimization problem—convergence in infinite dimensions can exhibit a strikingly rich variety of behaviors. This is very different from the pictures in traditional optimization.

4 Stochastic Gradient Descent

Note that the convergence rate given in Proposition 3.2 does not directly gives us the time complexity, as the compuational cost of each step is infinite. Moreover, we ignored the presence of label noise. In this section, we shift our focus to practical SGD under the noisy regime.

The key step in analyzing SGD, as opposed to the full-batch GD, lies in characterizing the structure of gradient noise. WLOG, we focus on the case $B = 1$, for which

$$\epsilon_t = \underbrace{\left(\varphi(x_t)\varphi(x_t)^\top - \mathbb{E}[\varphi(x)\varphi(x)^\top]\right)(\theta_t - \theta^*)}_{q_t} - \underbrace{\varphi(x_t)\xi_t}_{p_t}.$$

Let $\delta_t = \theta_t - \theta^*$ for $t \in \mathbb{N}$. We have

$$\delta_{t+1} = \delta_t - \eta_t(\Sigma\delta_t + \epsilon_t).$$

Noting $\Sigma = \mathrm{diag}(\lambda_1, \cdots, \lambda_p)$, the j-th component of δ_t is updated as

$$\delta_{t+1,j} = (1 - \eta_t\lambda_j)\delta_{t,j} - \eta_t e_j^\top \epsilon_t,$$

where $e_j \in \mathbb{R}^p$ denotes the cannonical basis of $\mathbb{R}^p$. Let $\Delta_{t,j} = \mathbb{E}\,\delta_{t,j}^2$. Since $\mathbb{E}\,\epsilon_t = 0$, we have

$$\Delta_{t+1,j} = (1 - \eta_t\lambda_j)^2\Delta_{t,j} + \eta_t^2\mathbb{E}(e_j^\top\epsilon_t)^2. \tag{15}$$

This implies that the problem boils down to estimate the noise energy (i.e., the second-order moment) along each coordinates.

The anisotropic structure of SGD noise. We turn to examine the quantity $\mathbb{E}(\epsilon_t^\top u)^2$ for any $u \in \mathbb{S}^{p-1}$. It holds that

$$\mathbb{E}(\epsilon_t^\top u)^2 = \mathbb{E}(q_t^\top u)^2 + \mathbb{E}(p_t^\top u)^2 - 2\,\mathbb{E}[(q_t^\top u)(p_t^\top u)] = \underbrace{\mathbb{E}(q_t^\top u)^2}_{Q_t} + \underbrace{\mathbb{E}(p_t^\top u)^2}_{P_t},$$

where the second step uses the independence between label noise and input. Then

$$P_t = \mathbb{E}[\xi_t^2]\,\mathbb{E}(\varphi(x_t)^\top u)^2 = \sigma^2 u^\top\Sigma u \tag{16}$$

$$Q_t \le \mathbb{E}(u^\top\varphi(x)\varphi(x)^\top\delta_t)^2 = \mathbb{E}[(u^\top\varphi(x))^2(\varphi(x)^\top\delta_t)^2]$$

To bound Q_t, we assume the feature map $\varphi(\cdot)$ satisfies the following condition:

Assumption 4.1 ((4,2)-Hypercontractivity condition)**.** *There exists some $C > 0$ such that it holds for any $u, v \in \mathbb{S}^{p-1}$ that*

$$\mathbb{E}[(u^\top\varphi(x))^2(\varphi(x)^\top v)^2] \le C\,\mathbb{E}[(u^\top\varphi(x))^2]\,\mathbb{E}[(\varphi(x)^\top v)^2]$$

This condition means that the fourth-order moment can be controlled by the second-order moment. This condition essentially means that our features $\varphi(X)$ are not too heavy-tailed and it is obvious that this condition holds when $\varphi(X)$ is sub-Gaussian for $X \sim \rho$.

With Assumption 4.1, we have

$$Q_t \le C\,\mathbb{E}[(u^\top\varphi(x))^2]\,\mathbb{E}[(\varphi(x)^\top\delta_t)^2] = 2C\mathcal{E}(\theta_t)u^\top\Sigma u \tag{17}$$

due to $\mathcal{E}(\theta_t) = \frac{1}{2}\delta_t^\top\,\mathbb{E}[\varphi(x)\varphi(x)^\top]\delta_t$, by definition.

Combining (16) and (17), we obtain:

Lemma 4.2 (Noise structure)**.** *For any $u \in \mathbb{S}^{p-1}$, it holds that*

$$\mathbb{E}(\epsilon_t^\top u)^2 \lesssim (u^\top \Sigma u)(\mathcal{E}(\theta_t) + \sigma^2) \asymp \mathcal{R}(\theta_t) u^\top \Sigma u.$$

Note that $u^\top \Sigma u$ represents the curvature along the direction u. This lemma therefore implies:

- The noise magnitude scales with the loss value.
- The noise energy in direction u scales with the directional curvature $u^\top \Sigma u$.

This stands in contrast to the usual stochastic-optimization assumption

$$\mathbb{E}[\|\epsilon_t\|^2] \le C$$

for some constant C. Instead, Lemma 4.2 shows that the gradient noise is anisotropic and aligned with the local geometry encoded by the Hessian Σ.

Theorem 4.3. *Define the intrinsic time as $T_t = \sum_{\tau=0}^{t} \eta_t$. Under the source condition (Assumption 2.1), we have*

$$\mathbb{E}[\mathcal{E}(\theta_t)] \lesssim \sum_{j=1}^{p} |a_j^*|^2 \lambda_j^s e^{-2\lambda_j T_t} + \sigma^2 \sum_{\tau=0}^{t} \eta_\tau^2 \sum_{j=1}^{p} \lambda_j^2 e^{-2\lambda_j (T_t - T_\tau)}$$

Proof. Towards the end of training, the excess risk becomes relatively small, i.e., $\mathcal{E}(\theta_t) \ll \sigma^2$, so we approximately have

$$\mathbb{E}(\epsilon_t^\top u)^2 \lesssim (u^\top \Sigma u)\sigma^2.$$

Plugging this into (15),

$$\Delta_{t+1,j} \le (1 - \eta_t \lambda_j)^2 \Delta_{t,j} + \eta_t^2 \lambda_j \sigma^2$$

It then holds that

$$\Delta_{t+1,j} \le \prod_{\tau=0}^{t} (1 - \eta_\tau \lambda_j)^2 \Delta_{0,j} + \lambda_j \sigma^2 \sum_{\tau=0}^{t} \eta_\tau^2 \prod_{\tau'=\tau+1}^{t} (1 - \eta_{\tau'} \lambda_j)^2 \tag{18}$$

Then for any τ, t, we have

$$\prod_{\tau'=\tau+1}^{t} (1 - \eta_{\tau'} \lambda_j)^2 = e^{\sum_{\tau'=\tau+1}^{t} \log(1-\eta_{\tau'}\lambda_j)^2} \approx e^{-2\sum_{\tau'=\tau+1}^{t} \eta_{\tau'}\lambda_j} = e^{-2\lambda_j (T_t - T_\tau)}$$

Therefore,

$$\Delta_{t+1,j} \le e^{-2\lambda_j T_t} \Delta_{0,j} + \lambda_j \sigma^2 \sum_{\tau=0}^{t} \eta_\tau^2 e^{-2\lambda_j (T_t - T_\tau)}$$

For the zero initialization, $\Delta_{0,j} = (\theta_j^*)^2$. Then, the excess risk is given by

$$\begin{aligned}
\mathbb{E}[\mathcal{E}(\theta_{t+1})] &= \frac{1}{2} \sum_{j=1}^{p} \lambda_j \Delta_{t,j} \\
&= \frac{1}{2} \sum_{j=1}^{p} \lambda_j e^{-2\lambda_j T_t} \Delta_{0,j} + \frac{1}{2} \sigma^2 \sum_{j=1}^{p} \lambda_j^2 \sum_{\tau=0}^{t} \eta_\tau^2 e^{-2\lambda_j (T_t - T_\tau)} \\
&= \frac{1}{2} \sum_{j=1}^{p} (a_j^*)^2 \lambda_j^s e^{-2\lambda_j T_t} + \frac{1}{2} \sigma^2 \sum_{\tau=0}^{t} \eta_\tau^2 \sum_{j=1}^{p} \lambda_j^2 e^{-2\lambda_j (T_t - T_\tau)}
\end{aligned}$$

□

Corollary 4.4. *Under Assumptions 2.1 and 2.2, we have*

$$\mathbb{E}[\mathcal{E}(\theta_{t+1})] \lesssim \frac{1}{1+T_t^s} + \sigma^2 \underbrace{\sum_{\tau=0}^{t} \eta_\tau^2 \frac{1}{1+(T_t - T_\tau)^{2-\frac{1}{\beta}}}}_{N_t}.$$

Proof. Applying the capacity condition, we have

$$\mathbb{E}[\mathcal{E}(\theta_{t+1})] = \frac{1}{2}\sum_{j=1}^{p} j^{-s\beta+1} e^{-2j^{-\beta}T_t} + \frac{1}{2}\sigma^2 \sum_{\tau=0}^{t} \eta_\tau^2 \sum_{j=1}^{p} j^{-2\beta} e^{-2j^{-\beta}(T_t - T_\tau)}$$

Following the same derivation as in Lemma 3.2, we have

$$\sum_{j=1}^{p} j^{-s\beta+1} e^{-2j^{-\beta}T_t} \lesssim \frac{1}{1+T_t^s}$$
$$\sum_{j=1}^{p} j^{-2\beta} e^{-2j^{-\beta}(T_t - T_\tau)} \lesssim \frac{1}{1+(T_t - T_\tau)^{2-\frac{1}{\beta}}}.$$

where we get the bound $\frac{1}{1+T^s}$ instead of $\frac{1}{T^s}$ by applying the argument when T is large. For small T, we naturally have a constant upper bound. The expected excess risk is then bounded as

$$\mathbb{E}[\mathcal{E}(\theta_{t+1})] \lesssim \frac{1}{1+T_t^s} + \sigma^2 \underbrace{\sum_{\tau=0}^{t} \eta_\tau^2 \frac{1}{1+(T_t - T_\tau)^{2-\frac{1}{\beta}}}}_{N_t}.$$

□

Lemma 4.5. *When using a constant learning rate schedule $\eta_\tau \equiv \eta$, we have*

$$N_t \lesssim \eta.$$

Proof. Let $\nu = 2 - \frac{1}{\beta} > 1$, then

$$\begin{aligned} N_t &= \eta^2 \sum_{\tau=0}^{t} \frac{1}{1+(\eta\tau)^\nu} \\ &= \eta^2 \left(\sum_{\tau < \frac{1}{\eta}} \frac{1}{1+(\eta\tau)^\nu} + \sum_{\tau > \frac{1}{\eta}} \frac{1}{1+(\eta\tau)^\nu} \right) \\ &\lesssim \eta^2 \left(\frac{1}{\eta} + \int_{\frac{1}{\eta}}^{+\infty} \frac{1}{(\eta z)^\nu} \, \mathrm{d}z \right) \\ &\lesssim \eta^2 \left(\frac{1}{\eta} + \frac{1}{\eta} \right) = \eta \end{aligned} \tag{19}$$

□

Convergence Rate. Therefore, under the constant LR $\eta_\tau \equiv \eta$, we have $T_t = t\eta$ and

$$\mathbb{E}[\mathcal{E}(\theta_t)] \lesssim \inf_{\eta>0} \left(\frac{1}{(\eta t)^s} + \eta \right).$$

By optimizing over η, we have

$$\eta_{\text{op}} \propto \left(\frac{1}{t}\right)^{\frac{s}{s+1}} \tag{20}$$

and

$$\mathbb{E}[\mathcal{E}(\theta_t)] \lesssim \left(\frac{1}{t}\right)^{\frac{s}{s+1}} \tag{21}$$

Sample Complexity. Note that in SGD, each step only uses one sample and thus, for n samples, SGD achives the error $O(n^{-s/(s+1)})$. Let us compare it with the one for KRR: $O(n^{-\beta \min(s,2)/(\beta \min(s,2)+1)})$.

- When β is close to 1, SGD is nearly optimal and moreover, can adapt to the smoothness of any $s \in (0, \infty)$. In contrast, KRR's smoothness adaptation saturate at $s = 2$, beyond which extra smoothness does not yield improvement.
- When β is significantly larger than 1, it seems that KRR's adaptation is stronger, as SGD's error is independent of the value of β.

However, it should be remarked that in driving (21), we only consider the schedule of constant LR. We conjecture that when adopting better LR schedule (such as exponential decay), we are able to obtain the minimax optimal rate $O(n^{-\beta s/(\beta s+1)})$.

Time Complexity. In practice, we are unable directly use the SGD algorithm, as p can be infinite. Thus, there are two approaches: truncation p to a finite one and representer theorem. Here, we discuss the second one:

Lemma 4.6 (Representer theorem for SGD)**.** *Let θ_t be the solution of SGD. Then, under zero initialization, we have $f_t := f_{\theta_t} \in \operatorname{span}\{k(x_i, \cdot) : i \in [t]\}$ given by*

$$f_{t+1} = \sum_{i=0}^{t} r_i k(x_i, \cdot), \quad r_t = -\eta_t (f_t(x_t) - y_t), \text{ for } t = 0, 1, \ldots.$$

with the initialization $f_0(x) \equiv 0$.

Proof. Note that the one-step update of SGD is given by

$$\theta_{t+1} = \theta_t - \eta_t \varphi(x_t)(\varphi(x_t)^\top \theta_t - y_t).$$

Let $f_t(x) := \varphi(x)^\top \theta_t$. Then,

$$\varphi(x)^\top \theta_{t+1} = \varphi(x)^\top \theta_t - \eta_t \varphi(x)^\top \varphi(x_t)(\varphi(x_t)^\top \theta_t - y_t)), \tag{22}$$

which is equivalent to

$$f_{t+1} = f_t - \eta_t k(\cdot, x_t)(f_t(x_t) - y_t).$$

□

By this lemma, at the t-th step, the compuation cost is $O(t)$. Thus, we have that the total compuation cost to reach the ϵ accuracy, we need $T_\epsilon = \epsilon^{-(1+1/s)}$ steps and the total computational cost is

$$C_\epsilon^{\mathrm{sgd}} = O(T_\epsilon^2) = O(\epsilon^{-(2+2/s)}).$$

This impoves the one of direct KRR given in (3) when β is close to 1.

5 A Fine-Grained Viewpoint: Spectral Bias

In the foregoing analysis, we have focused exclusively on the convergence of the overall generalization error $\mathcal{E}(\theta_t)$, which effectively averages convergence across the entire spectrum. However, by examining each spectral component individually, one uncovers the phenomenon of *spectral bias*, a critical concept for understanding model training in practice that extends beyond linear regression and kernel methods.

Specifically, by Eq. (13), the j-th spectral component converges at an exponential rate

$$O\big(e^{-\lambda_j t}\big).$$

Consequently, modes associated with smaller eigenvalues λ_j decay far more slowly than those linked to larger λ_j. Identifying which features or functions correspond to large versus small eigenvalues in real-world settings provides deeper insight into the algorithm's behavior.

For the purpose of illustration, consider a translation-invariant kernel on the torus $\mathbb{T}$:

$$k(x,x') \;=\; \kappa(x-x') \;=\; \sum_{j\in\mathbb{Z}} \hat{\kappa}(j)\, e^{i2\pi j(x-x')} \;=\; \sum_{j\in\mathbb{Z}} \hat{\kappa}(j)\, e_j(x)\, \overline{e_j(x')},$$

where $e_j(x) = e^{i2\pi jx}$ is the Fourier basis and $\hat{\kappa}(j)$ is the Fourier coefficients (which serves as the jth eigenvalue of k) of κ. The smoother κ is, the faster $\hat{\kappa}(j)$ decays as $|j| \to \infty$. Consequently, high-frequency modes (large $|j|$) correspond to small eigenvalues, while low-frequency modes correspond to large eigenvalues. Since the j-th component converges at rate $O(e^{-\hat{\kappa}(j)\,t})$, low frequencies are learned exponentially faster than high frequencies—a phenomenon we term the *curse of frequency*.

Although this example uses a periodic kernel, the same intuition applies to general **smooth kernels**: components associated with higher "frequency" or complexity decay more slowly. Moreover, a very similar spectral bias persists in nonlinear models such as **neural networks**:

- In Figure 1, we show the training curves for learning $f^*(x) = \sin(kx)$ for $k = 1, 5$ with two-layer ReLU networks. The optimizers are online SGD.
  - Obviously, two-layer neural networks also exhibit the curse of frequency.
  - Learning high-frequence functions is much slower than low-frequency functions.
- In Figure 4, we show the learning of a function with multiple frequencies:

  $$f^*(x) = c(1 + 0.5\sin(x) + 0.5\sin(5x))$$

  with the Gaussian kernel $k(x,x') = e^{-\frac{-(x-x')^2}{h^2}}$ with $h = 1$. We can see that the optimization process exhibit a very clean frequency bias. The low-frequency component is learned firstly.

For more illustration of the curse of frequency of neural networks, we refer to [Rahaman et al., 2019, Xu et al., 2019]

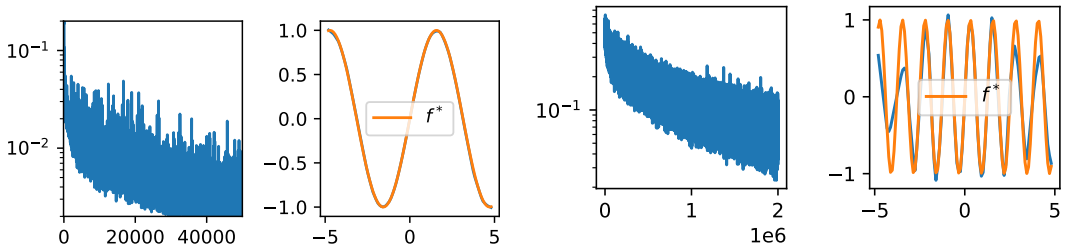

Figure 1: (left) $f^*(x) = \sin(x)$; (right) $f^*(x) = \sin(5x)$.

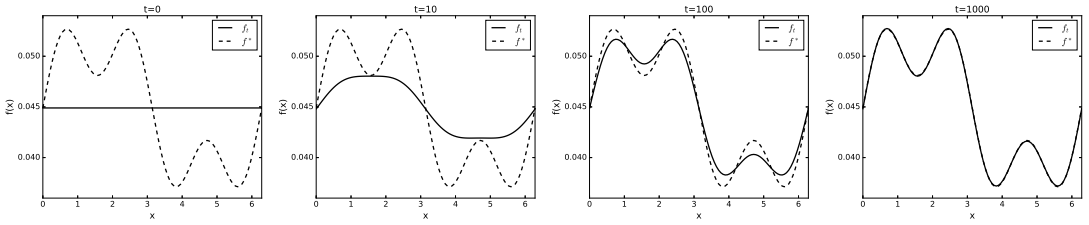

Figure 2: The example that demonstrates the curse of frequency. The model is linear regression with Gaussian kernel $e^{-(x-x')^2/(h^2)}$ with $h = 1$. The four plots correspond to the function f_t at $t = 0,\ 10,\ 100,\ 1000$, compared to the target function.

Alleviate the curse of frequency. When the target function f^* has substantial energy in its high-frequency components, learning with a smooth kernel of geneally smooth model can be very slow due to the curse of frequency. Therefore, one may wnat to accelerate the learning convergence by inflating the eigenvalues corresponding to the high-frequency eigenfunctions. One way to achieve this is to rescale the kernel:

$$k_h(x - x') = k\left(\frac{x - x'}{h}\right) = \sum_{j\in\mathbb{Z}} \hat{\kappa}(j) e^{i\frac{2\pi j}{h}(x-x')} = \sum_{j\in\mathbb{Z}} \hat{\kappa}(hj) e^{i2\pi j(x-x')},$$

which yields

$$\lambda_j = \hat{\kappa}(jh) \sim h^{-\gamma-\frac{1}{2}} j^{-\gamma-\frac{1}{2}}.$$

Although λ_j still decays at the same rate $j^{-\gamma-\frac{1}{2}}$, it is effectively scaled by a factor of $h^{-\gamma-\frac{1}{2}}$. Consequently, the high-frequency components (those with larger $|j|$) are associated with larger λ_j compared with before, thereby enabling faster convergence. In the limiting case $h \to 0$, we have

$$\lim_{h\to 0} k_h(x - x') \propto \delta(x - x'),$$

which implies $\lambda_j = O(1)$ for all frequencies. In Figure 3, we plot the triangular wave function

$$\kappa_h(x) = \left(1 - \frac{|x|}{h}\right) \mathbb{I}\{|x| < h\} \tag{23}$$

along with its corresponding Fourier transform $\hat{\kappa}_h(j)$. As h decreases, $\hat{\kappa}_h(\cdot)$ concentrates more of its energy in the higher-frequency regime.

Remark 5.1. *The basic idea is to modify the kernel so that its eigenvalue spectrum places greater weight on high-frequency components. Analogously, in neural networks one can choose*

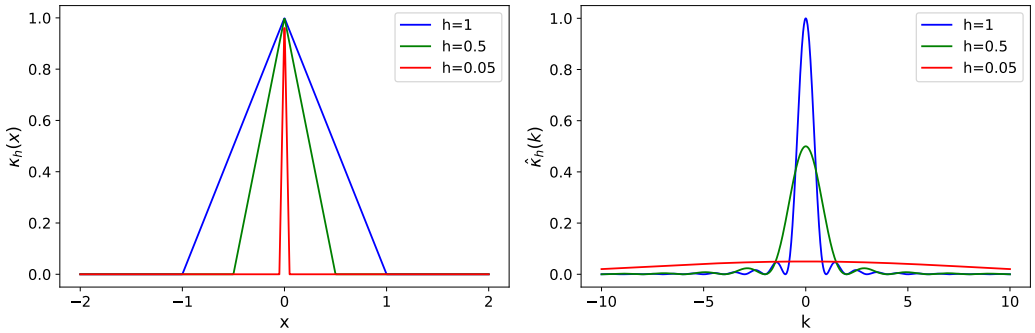

Figure 3: For smaller h, the Fourier transform $\hat{\kappa}_h(k)$ allocates a larger portion of its energy to higher frequencies.

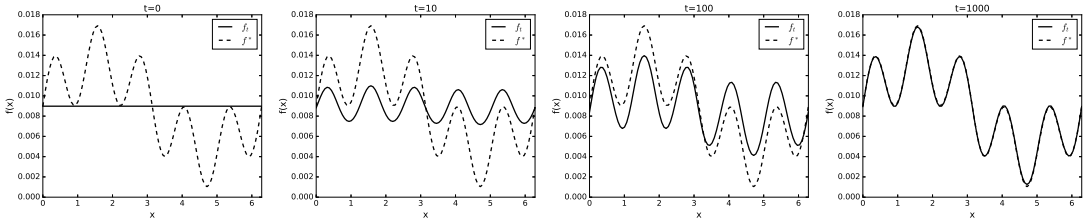

Figure 4: The example that demonstrates how we can alliciate the curse of frequency by modifying the spectrum of kernel functions. The model is linear regression with Gaussian kernel $e^{-(x-x')^2/(2h^2)}$ with $h = 0.2$. The four plots correspond to the function f_t at $t = 0,\ 10,\ 100,\ 1000$, compared to the target function.

activation functions that emphasize high frequencies, e.g., $\sigma(z) = \sin(kz)$ with large k, or a compactly supported bump function with small support, thereby skewing the network's spectral bias toward high-frequency directions.

In Figure 2, we illustrate learning with a Gaussian kernel of reduced bandwidth. Compared to Figure 4, the high-frequency components are learned significantly faster.

5.1 Link between KRR and numerical method of PDE

At first glance, the spectral bias phenomenon appears to contradict the observation that high-frequency components often converge first in numerical solutions of PDEs. To illustrate this, consider the 1D Poisson equation

$$u'' = f \quad \text{with} \quad u(0) = u(1) = 0,$$

and let us solve it by the finite element method. We partition the interval $[0, 1]$ into N sub-intervals of equal length $h = 1/N$, yielding grid points $x_i = ih$ where $h = 1/N$ for $i = 0, 1, \cdots, N$. We then define a family of piecewise linear basis functions

$$\varphi_i(x) = \frac{|x - h|}{h} \mathbb{I}\{|x - x_i| \le h\}, \quad i = 1, \cdots, N - 1$$

and seek an approximate solution

$$u_h(x) = \sum_{j=1}^{N-1} u_j \varphi_j(x),$$

where u_j are the undetermined coefficients. Obviously, $u_h(0) = u_h(1) = 0$. For any test function $v(x)$ satisfying $v(0) = v(1) = 0$, we have

$$\begin{aligned} -\int_0^1 f(x)v(x)\mathrm{d}x &= -\int_0^1 u''(x)v(x)\mathrm{d}x \\ &= -[u'(x)v(x)]\,|_0^1 + \int_0^1 u'(x)v'(x)\mathrm{d}x \\ &= \int_0^1 u'(x)v'(x)\mathrm{d}x. \end{aligned}$$

Substituting the finite element approximation $u = u_h$ and choosing test function $v = \varphi_i(x)$ into the above expression yields

$$\sum_{j=1}^{N-1} u_j \int_0^1 \varphi_j'(x)\varphi_i'(x)dx = -\int_0^1 f(x)\varphi_i(x)dx.$$

This can be compactly written in matrix form as

$$Au = f. \tag{24}$$

Here, $A = [a_{ij}] \in \mathbb{R}^{(N-1)\times(N-1)}$ is the stiffness matrix with $a_{ij} = \int_0^1 \varphi_j'(x)\varphi_i'(x)dx$, $f = [f_i]$ is the load vector with $f_i = -\int_0^1 f(x)\varphi_i(x)dx$, and $u = [u_1, \cdots, u_{N-1}]$ is the solution vector. Since each $\varphi_i(x)$ is piecewise linear, entries of A follow the standard finite element pattern:

$$a_{ii} = \frac{2}{h}, \quad a_{i,i+1} = a_{i+1,i} = -\frac{1}{h}, \quad a_{ij} = 0 \quad \text{if } |i-j| \geq 2. \tag{25}$$

We can construct a kernel $k_h(x, x')$ for which $a_{ij} = k_h(x_i, x_j)$. A straightforward choice is $k_h(x, x') = \kappa_h(x - x')$ where

$$\kappa_h(x) = \left(2 - \frac{3}{h}|x|\right)\mathbb{I}\{0 \leq |x| < h\} + \left(-2 + \frac{|x|}{h}\right)\mathbb{I}\{h \leq |x| \leq 2h\}.$$

Figure 5 displays $\kappa_h(x)$ along with its Fourier transform for $h = 0.5, 0.2, 0.1$. As shown, $\hat{\kappa}(\omega)$ initially grows with $|\omega|$ and then decreases. Moreover, as h becomes smaller, the location of the maximal amplitude in the frequency domain shifts to larger $|\omega|$. Consequently, high-frequency components converge more rapidly, and diminishing h accelerates the convergence of these higher-frequency components.

There is a contrast between the finite element method and KRR. Generally speaking, in KRR where the kernel is global, the eigenfunctions corresponding to larger eigenvalues tend to be lower-frequency compared to those associated with smaller eigenvalues. However, the analysis above seems to show a contrary result, that higher-frequency components converge faster. The reason is that, as $h \to 0$, the kernel k_h becomes more localized.

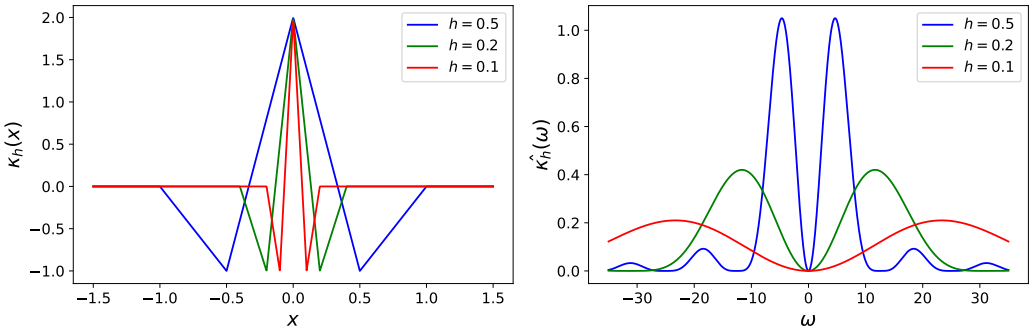

Figure 5: Plots of the piecewise function $\kappa_h(x)$ (left) and and its Fourier transform $\hat{\kappa}_h(\omega)$ (right) for $h = 0.5, 0.2, 0.1$. The peak of $\hat{\kappa}_h(\omega)$ moves to higher frequencies as h decreases.

References

[Rahaman et al., 2019] Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A. (2019). On the spectral bias of neural networks. In *International Conference on Machine Learning*, pages 5301–5310. PMLR.

[Xu et al., 2019] Xu, Z.-Q. J., Zhang, Y., Luo, T., Xiao, Y., and Ma, Z. (2019). Frequency principle: Fourier analysis sheds light on deep neural networks. *arXiv preprint arXiv:1901.06523*.