NOTE ON REPRODUCING KERNEL HILBERT SPACES

KENTA OONO

1. notation

- $A := B$: define A by B.
- $\|v\|_2$: L^2 norm of the vector $v \in \mathbb{R}^d$.
- vectors are represented as column vectors.
- $\langle x, x' \rangle_{\mathcal{H}}$: The inner product of x and x' in Hilbert space $\mathcal{H}$.
- $\|x\|_{\mathcal{H}}$: The norm of x in Hilbert space $\mathcal{H}$.
- We only consider Hilbert space on $\mathbb{R}$.
- $\mathrm{Span}(x_1, \ldots, x_N) = \left\{ \sum_{n=1}^N \alpha_n x_n \mid \alpha_n \in \mathbb{R} \right\}$

2. Linear regression

2.1. **Problem setting.** Dataset $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, where $N = |\mathcal{D}|$ is the number of instances and $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$. For parameter $w \in \mathbb{R}^d$, we consider the linear model.

$$f_w(x) := \sum_{n=1}^N w_n x_n = w^T x \tag{2.1}$$

Remark 2.1. Normally linear model is formulated as $f(w, b) = w^T x + b$. But, we can omit the bias vector b by replacing $x$with $\tilde{x} = (x, 1)$. So, we omit the bias vector from the linear model in this note.

The task is to choose the most appropriate w that can model the dataset from $\mathbb{R}^d$. We evaluate the performance of the model by calculating the deviation of the output of the model $f(x)$ from the target value y. There are several choices for evaluating the deviation. One of such metric is the L^2 norm defined by

$$l(y, y') := \frac{1}{2}(y - y')^2 \tag{2.2}$$

for $y, y' \in \mathbb{R}$.

Remark 2.2. The coefficient $\frac{1}{2}$ is not essential. We add it to make the calculation simple.

So the performance of the model is defined as $L'(w) := \sum_{n=1}^N l(f(x_n), y_n)$. But we do not use this function as a measure of performance. We step one more further. We add the penalty so that we should not choose w with too large norm. The

Preferred Networks Inc.

simplest choice is to add the L^2 norm of the parameter $\|w\|_2^2$. So the resulting function is

$$\begin{aligned} L(w) &= L'(w) + \frac{\lambda}{2}\|w\|_2^2 \\ &= \sum_{n=1}^{N} l(f_w(x_n), y_n) + \frac{\lambda}{2}\|w\|_2^2 \end{aligned} \tag{2.3}$$

where $\frac{\lambda}{2}$ is a constant. In the context of machine learning, a function that evaluates the performance of the parameter like L is called the *loss function* and the function like the second term of L is call the *regularization term*. As both terms in L are non-negative, w must be chosen so that both $f_w(x_n, y_n)$ for each n and $\|w\|_2$ are small to reduce the value of L. λ regulates the effect of the second term. The larger w, we reluctant to choose w with large L^2 norm. When $\lambda = 0$, we do not impose the penalty on the norm at all.

Remark 2.3. Different from w, λ is not a tunable parameter that minimizes L but rather is treated as a constant. Such constants are called the *hyper parameters*. Tuning of hyper parameters (i.e. choice of the appropriate hyper paramters) is out of scope of this note.

Remark 2.4. L implicitly depends on the dataset $\mathcal{D}$, but as we do not change the dataset, we omit $\mathcal{D}$ from the argument for simplicity of the notation.

Remark 2.5. Sometimes, we add the normalizer $\frac{1}{N}$ to the definition of L'. In that case, L' is often called the *mean squared error*. The essence does not change even if we introduce this coefficient because we can absorb it to the change of λ.

Remark 2.6. Some of the readers may think that the choice of the form of L is artificial, but we can justify the choice of the functional form of L above from the probabilistic point of view. Specifically, the minimization of L with respect to w is equivalent to MAP estimate when we construct an appropriate probabilistic model of w, x and y.

2.2. **Solution.** We reformulate L as follows to calculate the derivative.

$$L(w) = \frac{1}{2}\|Xw - y\|_2^2 + \frac{\lambda}{2}\|w\|_2^2 \tag{2.4}$$

where

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} = \begin{bmatrix} x_1^1 & \cdots & x_1^d \\ \vdots & & \vdots \\ x_n^1 & \cdots & x_n^d \end{bmatrix}, y = \begin{bmatrix} y_1 \\ \vdots \\ y_d \end{bmatrix}$$

As L is quadratic in w, we can easily calculate the minimizer of L by differentiating L with respect to w, by direct calculation, $\frac{\partial L}{\partial w} = X^T(Xw - y) + \lambda w$. Setting $\frac{\partial L}{\partial w} = 0$,

$$\frac{\partial L}{\partial x} = 0 \Leftrightarrow (X^T X + \lambda)w = X^T y. \tag{2.5}$$

Note that as $X^T X$ is a symmetric positive semi-definite matrix and $\lambda > 0$, $X^T X + \lambda$ is invertible. So, the minimizer is

$$w^* := (X^T X + \lambda)^{-1} X^T y \tag{2.6}$$

2.3. **Introduction of kernel functions.** Now, let's reformulate what we have done in the previous subsection. Recall the minimizer w^* should satisfy $(X^TX + \lambda)w = X^TY$ We introduce the new variable K by

$$K = XX^T = \begin{bmatrix} x_1^T x_1 & \dots & x_n^T x_1 \\ \vdots & & \vdots \\ x_n^T x_1 & \dots & x_n^T x_n \end{bmatrix}$$

So, K is a matrix that collects pair-wise inner products of x_n's. We call K the *Gram matrix.* K is by definition, a symmetric positive semi-definite matrix.

Also, we define α by $\alpha = (K + \lambda)^{-1}y$. Note that $K + \lambda$ is invertible by the same reason as $X^TX + \lambda$.

Proposition 2.7. $w^* = X^T\alpha$

Proof.

$$\begin{aligned} (X^TX + \lambda)w^* &= X^Ty \\ &= X^T(K + \lambda)(K + \lambda)^{-1}y \\ &= X^T(XX^T + \lambda)(K + \lambda)^{-1}y \\ &= (X^TX + \lambda)X^T(K + \lambda)^{-1}y \\ &= (X^TX + \lambda)X^T\alpha \end{aligned} \tag{2.7}$$

Multiplying $(X^TX + \lambda)^{-1}$ from left yields the claim. □

Remark 2.8. As we will see later, this relation between w^* and α holds true in more general situation thanks to the theorem known as Representer theorem.

For a new instance $\tilde{x}$, the model predicts the target value $\tilde{y}$ by

$$\tilde{y} = f_w^*(\tilde{x}) = w^{*T}\tilde{x} = (X^T\alpha)^T\tilde{x} = \alpha^TX\tilde{x} = \alpha^T\tilde{k} \tag{2.8}$$

where k is a collection of inner products of x_n's and $\tilde{x}$

$$\tilde{k} = X\tilde{x} = \begin{bmatrix} x_1^T\tilde{x} \\ \vdots \\ x_n^T\tilde{x} \end{bmatrix}$$

.

3. NONLINEARLITY

The crux here is that we need the values α and $\tilde{x}$, which does not need x_n's and $\tilde{x}$ themselves but their inner products to make a prediction of new instance $\tilde{x}$. As we do not have to handle x_n directly, x_n's even need not to be a vector. It is enough for us to define the function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that works as an inner product of x's (we will show the requirements imposed to k to make it be used as the substitute of the inner product). The function k is called the *kernel function.* Here, $\mathcal{X}$ is a set that works as a domain of x's. $\mathcal{X}$ need not endowed with inner product even $\mathbb{R}^d$) domain of x,

We substitute the inner product with k.

$$f_w^*(\tilde{x}) = \alpha^T\tilde{k} \tag{3.1}$$

where

$$K = \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ \vdots & & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{bmatrix}, \tilde{k} = \begin{bmatrix} k(x_1, \tilde{x}) \\ \vdots \\ k(x_n, \tilde{x}) \end{bmatrix} \tag{3.2}$$

and $\alpha = (K + \lambda)^{-1} y$. If there exists a Hilbert space $\mathcal{H}$ and function $\phi : \mathcal{X} \to \mathcal{H}$ such that

$$k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}, \tag{3.3}$$

then, the discussion in the previous section works by replacing x with $\phi(x)$.

Example 3.1. If $\mathcal{X}$ is endowed with the inner product, we can use it to define k by $k(x, x') = x^T x'$. We should choose $\phi(x) = x$ in this case. Of course, we do not have to define k in this way even if $\mathcal{X}$ has an inner product. By defining more complex k, we can make the model f more complex.

Example 3.2 (Polynomial kernel)**.** Let's think of the first example of this generalization. We define $k(x, x') = (x^T x' + 1)^2$ for $x, x' \in X = \mathbb{R}^D$.

$$k(x, x') = (x^T x' + 1)^2 = \phi(x)^T \phi(x') \tag{3.4}$$

where $\phi(x) = (x^1 x^1, \ldots x^i x^j, \ldots, x^D x^D, \sqrt{2} x^1, \ldots \sqrt{2} x^D, 1)^T$. So, using k as an alternative of the ordinal input is equivalent to first mapping each sample x to more high-dimensional space $\mathbb{R}^{\tilde{D}}$ where $\tilde{D} = D^2 + D + 1$ by ϕ and consider the linear model in $\mathcal{R}^{\tilde{D}}$.

We can generalize this discussion to the case $k(x, x') = (x^T x' + 1)^e$ for some non-negative integer e. This kernel is known as *polynomial kernel*.

Example 3.3 (RBF kernel)**.** We define $k(x, x') = a \exp(-b|x - x'|^2)$ for $x, x' \in \mathbb{R}^d$ where a, b are hyper parameters. We can no longer write $k(x, x')$ as the inner product of finite-dimensional vectors. But from the general theory explained later, we can prove that there exists ϕ and $\mathcal{H}$ that satisfies the relation. By considering the Taylor expansion of k, $\phi(x)$ should be intuitively the infinite collection of $1, a_i x_i, b_{ij} x_i x_j, c_{ijk} x_i x_j x_k, \ldots$ where a_i, b_{ij}, c_{ijk} are some constants. This kernel is known as the *RBF kernel* (Radial Basis Function kernel).

Example 3.4 (String kernel)**.** This example describes that x need not to be a $\mathbb{R}$-valued vector(TBD)

Remark 3.5. In machine learning, we often preprocess data in which we convert each raw sample into a vector that represents the characteristic of the sample so that we can handle the data mathematically. Such a preprocess is called *feature extraction*. From that view point, ϕ is sometimes called a *feature map*.

Example 3.6. Experiment condition of Figure 3.6.

- $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{100}$
- $x_n \sim \mathrm{Unif}(-1, 1)$, i.i.d.
- $y_n \sim \mathcal{N}(y_n \mid f(x_n), \sigma^2)$ where $f(x) = 2x^3 - 3x^2 + 2x + 2$
- $k(x, x') = \exp(-\|x - x'\|^2)$ (RBF kernel)
- Introduce bias term $\tilde{x} = (x, 1)$
- Regularization parameter $\lambda = 1$

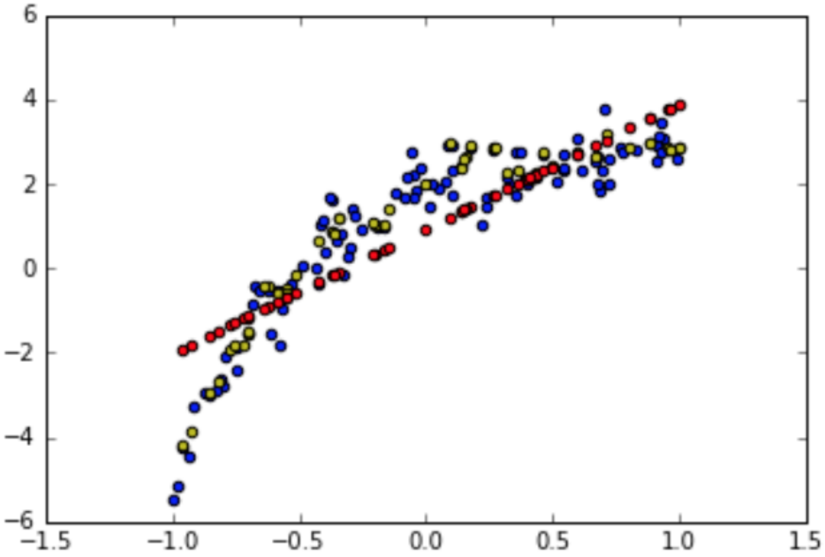

Figure 1. Kernel regression. Blue: training samples. Red: Linear regression of test samples. Yellow: Regression with RBF kernel of test samples.

4. reproducing kernel

In this section, we will introduce three concepts, which turned out to be equivalent.

- Reproducing kernels and associated Reproducing Kernel Hilbert Spaces (RKHSs).
- Kernels
- Symmetric positive semi-definite functions.

4.1. Definitions.

Let $\mathcal{X}$ be a set.

Definition 4.1. A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is *positive semi-definite* if for any $N \in \mathbb{N}_{>0}$ and any $x_1, \dots, x_N \in \mathcal{X}$, the *Gram matrix*

$$K = \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ \vdots & & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{bmatrix} \tag{4.1}$$

is positive semi-definite.

Definition 4.2. A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a *kernel* if there exists a Hilbert space $\mathcal{H}$ and a function $\phi : \mathcal{X} \to \mathcal{H}$ such that

$$k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}} \tag{4.2}$$

Remark 4.3. The feature map ϕ in this definition is not unique in general.

Definition 4.4. A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a *reproducting kernel* if there exists a Hilbert space $\mathcal{H}$ consists of real-valued functions of $\mathcal{X}$ (i.e. $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$) that satisfies the following conditions

(1) $k(x, \cdot) \in \mathcal{H}$ for all x.
(2) $f(x) = \langle k(x, \cdot), f \rangle_{\mathcal{H}}$ (reproducing property).

Proposition 4.5. *For a Hilbert space $\mathcal{H}$ in $\mathbb{R}^{\mathcal{X}}$, the followings are equivalent.*

(1) *For all $x \in \mathcal{X}$, the evaluation functional $\mathrm{ev}_x : \mathcal{H} \to \mathbb{R}, \mathrm{ev}_x(f) = f(x)$ is continuous.*
(2) *$\mathcal{H}$ has a reproducing kernel.*

Proof. (1)⇒(2): As ev_x is continuous, there uniquely exists $\phi_x \in \mathcal{H}$ such that $\mathrm{ev}_x(f) = \langle \phi_x, f \rangle$ by Riesz's representation theorem. Also, we define $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ by $k(x, x') = \langle \phi_x, \phi_{x'} \rangle_{\mathcal{H}}$. By definition, k is a symmetric function on $\mathcal{X}$. We prove that k is a reproducing kernel of $\mathcal{H}$. As ϕ_x itself is an element of $\mathcal{H}$, we can feed $\mathrm{ev}_{x'}$ with $\phi(x)$ for any $x' \in \mathcal{X}$ and

$$\phi_x(x') = \mathrm{ev}'_x(\phi_x) = \langle \phi_x, \phi_{x'} \rangle_{\mathcal{H}} = k(x, x'). \tag{4.3}$$

Therefore, $\phi_x = k(x, \cdot)$ holds. Especially, $k(x, \cdot) \in \mathcal{H}$. The reproducing property is proved as follows:

$$f(x) = \mathrm{ev}_x(f) = \langle f, \phi_x \rangle_{\mathcal{H}} = \langle f, k(x, \cdot) \rangle_{\mathcal{H}}. \tag{4.4}$$

Therefore, k is a reproducing kernel of $\mathcal{H}$.

(2)⇒(1): For all $x \in \mathcal{X}$,

$$|f(x)|^2 = \langle f, k(x, \cdot) \rangle_{\mathcal{H}}^2 \le \|f\|_{\mathcal{H}}^2 \|k(x, \cdot)\|_{\mathcal{H}}^2 \to 0 \tag{4.5}$$

as $f \to 0$ in $\mathcal{H}$. Therefore, ev_x is continuous. □

Definition 4.6. The Hilbert space $\mathcal{H}$ is called *Reproducing Kernel Hilbert Space* (RKHS in short) if it satisfies one of (therefore both of) coditions in the previous proposition.

Remark 4.7. We can prove that for RKHS $\mathcal{H}$, there exists an *unique* reproducing kernel k.

4.2. **Equivalence of three concepts.**

Proposition 4.8. *For a symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, the followings are equivalent*

(1) *k is a reproducing kernel.*
(2) *k is a kernel.*
(3) *k is a positive semi-definite function.*

Proof. (1)⇒(2): We have already proved it in the proof of 4.5.

(2)⇒(3): Suppose k is written as $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$ for some function $\phi : \mathcal{X} \to \mathcal{H}$. Then, for any $N \in \mathbb{N}_{>0}$, $a = (a_1, \ldots a_N) \in \mathbb{R}^N$ and $x_1, \ldots, x_N$, we have

$$\begin{aligned} a^T K a &= \begin{bmatrix} a_1 & \cdots & a_N \end{bmatrix} \begin{bmatrix} \langle \phi_1, \phi_1 \rangle_{\mathcal{H}} & \cdots & \langle \phi_1, \phi_N \rangle_{\mathcal{H}} \\ \vdots & & \vdots \\ \langle \phi_N, \phi_1 \rangle_{\mathcal{H}} & \cdots & \langle \phi_N, \phi_N \rangle_{\mathcal{H}} \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_N \end{bmatrix} \\ &= \sum_{i,j=1}^{N} a_i a_j \langle \phi_i, \phi_j \rangle_{\mathcal{H}} = \left\langle \sum_i a_i \phi_i, \sum_j a_j \phi_j \right\rangle_{\mathcal{H}} \geq 0, \end{aligned} \tag{4.6}$$

were, we write $\phi_i = \phi(x_i)$ for short. The last inequality follows from the positivity of the inner product. Therefore, k is positive semi-definite.

(3)⇒(1): This is known as *Moore-Aronszajn* theorem. We omit the proof here, see [8] section 4 for the complete proof. □

Corollary 4.9. *There is one to one correspondence between kernel functions and RKHSs in* $\mathbb{R}^{\mathcal{X}}$*.s*

Remark 4.10. Although we do not prove Moore-Aronszajn theorem in this note, it is instructive to describe the concrete construction of the Hilbert space associated to the positive semi-definite kernel k.

First, we make "pre-RHKS" (we do not define this term in this note, but I think this wording is easy to convey the underlying motivation) $\mathcal{H}_0$ and extend it to the genuine RHKS by adding the limit of sequences in $\mathcal{H}$. But we must be care the extension is a bit different from the ordinal completion to make a Hilbert space from pre-Hilbert space.

$\mathcal{H}_{\prime}$ is defined as the linear span of the function of the form $k(x, \cdot)$.

$$\mathcal{H}_0 = \left\{ \sum_{i=1}^{N} a_i k(x_i, \cdot) \mid N \in \mathbb{N}, a_i \in \mathbb{R}, x_i \in \mathcal{X} \right\}. \tag{4.7}$$

We define the inner product of $f = \sum_{i=1}^{N} a_i k(x_i, \cdot)$ and $g = \sum_{j=1}^{M} b_j k(y_j, \cdot) \in \mathcal{H}_{\prime}$ by

$$\langle f, g \rangle := \sum_{i,j} a_i b_j k(x_i, y_j). \tag{4.8}$$

Then, $\mathcal{H}$ consists of the function $f : \mathcal{X} \to \mathbb{R}$ which has a Cauthy sequence (with respect to the topology induced by the inner product defined above) $\{f_n\}_n$ that converge to f pointwise. We define the inner product of $\mathcal{H}$ as the limit of inner product of two Cauthy sequences in $\mathcal{H}_0$. We can prove that

- $\mathcal{H}$ is a Hilbert space, that is, the inner product is well-defined and $\mathcal{H}$ is complete with respect to the norm induced by the inner product.
- The inner product of $\mathcal{H}_0$ is indentical to that of $\mathcal{H}$ restricted to $\mathcal{H}_0$.
- $\mathcal{H}_0$ is dense in $\mathcal{H}$.
- k is a reproducing kernel of $\mathcal{H}$.

5. Kernel method

With the results of previous section in hand, let's generalize the linear regression problem of section 2. Dataset $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, where N is the number of instances and $x_i \in \mathcal{X}$ and $y_i \in \mathbb{R}$. Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a positive semi-definite kernel. From the previous discussion, there exists a Hilbert space $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ whose reproducing kernel is k such that $k(x, \cdot) \in \mathcal{H}$ for all $x \in \mathcal{X}$ and $f(x) = \langle k(x, \cdot), f \rangle_{\mathcal{H}}$

for all $\mathcal{H}$. We define the feature extraction function $\phi : \mathcal{X} \to \mathcal{H}$ by $\phi(x) := k(x, \cdot)$. Note that $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$ for all $x, x' \in \mathcal{H}$.

Remark 5.1. As ϕ is not unique for fixed k, we choose and fix one particular ϕ. We will see the result does not depend on the choice of ϕ that satisfies $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$.

For parameter $w \in \mathcal{H}$, we consider the linear model.

$$f_w(x) := \langle w, \phi(x) \rangle_{\mathcal{H}} \, (= \langle w, k(x, \cdot) \rangle_{\mathcal{H}}) \tag{5.1}$$

Let $l : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ be a function that evaluates the deviation of the prediction from the target value. Also we introduce the regularization of the parameter as we did in the previous problem setting. We assume that the regularizer depends on the norm of the parameter and non-decreasing with respect to the norm. Regularization term can be written as $R(\|w\|_{\mathcal{H}}^2)$ for some non-decreasing function $R : \mathbb{R} \to \mathbb{R}$. We define the loss function L by the weighted sum of the error term and regularization term:

$$L(w) = \sum_{n=1}^{N} l(f_w(x_n), y_n) + \frac{\lambda}{2} R(\|w\|_{\mathcal{H}}^2) \tag{5.2}$$

where λ is a hyper parameter. The task is to find w that minimizes L.

Remark 5.2. The existence nor uniqueness of the minimizer is not known in general. So, it is enough for us to find one of the minimizers, if any.

At first sight, this problem is difficult to solve because we must find the optimal parameter from potentially infinite space $\mathcal{H}$. But thanks to the following theorem known as *Representer theorem*, we can restrict the search space to the finite-dimensional one.

Theorem 5.3 (Representer theorem)**.** *Let* $\mathcal{H}_0 = \mathrm{Span}(\phi(x_1), \ldots, \phi(x_N))$. *Let* $w \in \mathcal{H}$ *and* w_0 *be the projection of* w *to* $\mathcal{H}_0$*, then,* $L(w_0) \leq L(w)$.

Proof. We project decompose w into $w = w_0 + w_1$ where $w_0 \in \mathcal{H}_0$ and $w_1 \in \mathcal{H}_0^{\perp} := \{f \in \mathcal{H} \mid \langle f, g \rangle_{\mathcal{H}} = 0 \quad \forall g \in \mathcal{H}_0\}$. Then,

$$\begin{aligned} f_w(x) &= \langle w, \phi(x) \rangle_{\mathcal{H}} \\ &= \langle w_0 + w_1, \phi(x) \rangle_{\mathcal{H}} \\ &= \langle w_0, \phi(x) \rangle_{\mathcal{H}} + \langle w_1, \phi(x) \rangle_{\mathcal{H}} \\ &= \langle w_0, \phi(x) \rangle_{\mathcal{H}} \quad (\because w_1 \in \mathcal{H}^{\perp}) \\ &= f_{w_0}(x) \end{aligned} \tag{5.3}$$

The fourth line follow from the fact $w_1 \in \mathcal{H}_0^{\perp}$. Further, as $\langle w_0, w_1 \rangle_{\mathcal{H}} = 0$, $\|w\|_{\mathcal{H}}^2 = \|w_0\|_{\mathcal{H}}^2 + \|w_1\|_{\mathcal{H}}^2$. In particular, $\|w\|_{\mathcal{H}}^2 \geq \|w_0\|_{\mathcal{H}}^2$. As R is non-decreasing, it follows that $R(\|w\|_{\mathcal{H}}^2) \geq R(\|w_0\|_{\mathcal{H}}^2)$. Combining the two, we get $L(w) \geq L(w_0)$. □

Therefore, if there exists minimizers of L, one of them should be of the form $w = \sum_{n=1}^{N} \alpha_n \phi(x_n)$. As k is the reproducing kernel of $\mathcal{H}$, $\phi(x_n) = k(x_n, \cdot) \in \mathcal{H}$. Therefore, $w = \sum_{n=1}^{N} \alpha_n \phi(x_n) \in \mathcal{H}$ for all $\alpha \in \mathbb{R}^N$. As our goal is to find at least one minimizer, we should search for $\alpha = (\alpha_1, \ldots, \alpha_N)$, not w, that minimizes L.

From that perspective, we define f_α as an alternative of f_w as $f_\alpha = \sum_n \alpha_n k(x_n, \cdot) \in \mathcal{H}$ so that $f_\alpha = \langle w, \phi(x) \rangle_{\mathcal{H}}$ for $w = \sum_{n=1}^N \alpha_n \phi(x_n)$. For such w, we can show that

$$\begin{aligned} \|w\|_{\mathcal{H}}^2 &= \left\langle \sum_{n=1}^N \alpha_n \phi(x_n), \sum_{m=1}^N \alpha_m \phi(x_m) \right\rangle_{\mathcal{H}} \\ &= \sum_{n,m} \alpha_n \alpha_m \left\langle \phi(x_n), \phi(x_m) \right\rangle_{\mathcal{H}} \\ &= \|f_\alpha\|_{\mathcal{H}}^2. \end{aligned} \tag{5.4}$$

Therefore, we redefine the loss function L as

$$L(\alpha) = \sum_{n=1}^N l(f_\alpha(x_n), y_n) + \frac{\lambda}{2} R(\|f\|_{\mathcal{H}}^2). \tag{5.5}$$

Remark 5.4. $\|w\|_{\mathcal{H}}^2$ can be also written as $\alpha^T K \alpha$ where we again define the Gram matrix K by

$$K = \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ \vdots & & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{bmatrix}. \tag{5.6}$$

Remark 5.5. The fact that one of the minimizers (if exists) can be written as the linear combination of $\phi(x_n)$'s corresponds to the previous proposition 2.7, in which we set $\mathcal{X} = \mathcal{H} = \mathbb{R}^d, k(x, x') = x^T x'$, and $\phi(x) = x$.

Remark 5.6. The correspondence of $w \in \mathcal{H}_0$ and $\alpha \in \mathbb{R}^N$ is not 1-to-1 as ϕ can "collapse" the input space and send different α's to same w. That is why we introduce new function f_α and redefine L as a function of α.

The task we have to do is

- (Train) to find optimal (or at least near-optimal) α that minimizes L.
- (Inference) to calculate $f_\alpha(x)$ with fixed α and x.

Remark 5.7. As we can see, the feature space ϕ does not appear in the functional form of L and f_α. So, optimal α and the value of f_α do not depend of the choice of the feature function ϕ.

In some cases, we can explicitly calculate the minimizer of L explicitly, as we see in section 2. But in general, we cannot calculate the optimal α analytically (i.e. we cannot derive the explicit formula of α). In that case, we need to make use of some optimization method to find optimal or at least near-optimal α. If l is convex, there exist various kind of optimization algorithms to find the minimizer of L. Even if l is non-convex, gradient-based algorithm sometimes achieve local optimal. In the next note, we will derive how to approximately solve this optimization problem.

6. Randomized algorithm for fast and memory-efficient calculation of kernels

6.1. **motivation.** Kernel trick is a technique that elegantly introduces non-linearity to linear models without handling feature map directly. But it requires to work on the Gram matrix of the training dataset. For example, in the simplest case we dealt with in the previous note, we have to calculate $\alpha = (K + \lambda)^{-1} y$, whose complexity is $O(N^3)$ computational complexity where N is the number of training data. It is prohibitive when N is large. Another example is the evaluation of $f_w(x) = \sum_{n=1}^{N} \alpha_n k(x_n, x)$. It typically takes $O(ND)$ operation if $\dim \mathcal{X} = D$ (e.g. $k(x, x') = x^T x$ or $\exp(-a\|x - x'\|^2)$). Further, we must retain whole training dataset to make a inference of newly added test sample.

Inference is equivalent to calculate $f_w(x) = \langle w, \phi(x) \rangle_{\mathcal{H}}$ for new sample x, as we do in the linear regression, which is prohibitive in general because $\phi(x)$ is typically high dimensional, or possibly infinite dimensional. But if we could approximate the feature map ϕ with the (not necessarily injective) embedding to the low dimensional space $z : \mathcal{X} \to \mathbb{R}^S$, where $\dim \mathcal{X} \gg S$, we can use it to construct a linear classifier $f_{\tilde{w}}(x) := \tilde{w}^T z(x)$, it drastically decrease the operation cost to typically $O(D + S)$, which is independent of N.

So the goal is to find the approximation of feature map so that, the following approximation holds:

$$k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}} \approx z(x)^T z(y). \tag{6.1}$$

In this section, we restrict ourselves to translation-invariant kernel on $\mathcal{X} = \mathbb{R}^D$, i.e. a kernel function k that satisfies $k(x, x') = k(x - a, x' - a)$ for all $x, x', a \in \mathbb{R}^D$. By setting $\psi(x) := k(x, 0)$, k is written as $k(x, x') = \psi(x - x')$. Conversely, even function $\psi : \mathbb{R}^D \to \mathbb{R}$ defines a symmetric function k.

Remark 6.1. When we consider $\mathbb{R}^D$ an abelian group by ordinal addition. $\mathbb{R}^D$ acts on $L^2(\mathbb{R}^D \times \mathbb{R}^D)$ by translation,$a \cdot k(x, x') := k(x - a, x' - a)$. Translation-invariant kernel is a fixed point of this action.

6.2. **Random Fourier Feature.** Rahimi and Recht proposed in [5] that fast and memory efficient approximation of the feature function. Schematically, they made use of the following calculation.

$$k(x, x') = C \int \zeta_\omega(x) \zeta^*_\omega(x') p(\omega) \mathrm{d}\omega \tag{6.2}$$

The following theorem known as Bochner's theorem is the key for our algorithm.

Theorem 6.2 (Bochner's theorem). *For continuous even function $\psi : \mathbb{R}^D \to \mathbb{R}$ such that $\psi(0) = 1$, we define symmetric function $k(x, x') := \psi(x - x')$. k is a kernel function if and only if ψ is a real-valued characteristic function for some probability measure.*

Proof. "If" part is easy to derive. Let μ be a probability measure and write $\psi(x)$ as $\int \zeta_\omega(x) \mathrm{d}\mu(\omega)$ where $\zeta_\omega(x) := \exp(\sqrt{-1}\omega^T x)$. Let $N \in \mathbb{N}_{>0}$, $a = (a_1, \ldots, a_N) \in \mathbb{R}^N$, and $x_1, \dots, x_N \in \mathbb{R}^N$. We set $K = (k(x_i, x_j))_{i,j=1}^N = (\psi(x_i - x_j))_{i,j}$. Then,

$$\begin{aligned} a^T K a &= \sum_{i,j=1}^{N} a_i a_j \psi(x_i - x_j) \\ &= \sum_{i,j} a_i a_j \int \zeta_\omega(x_i - x_j) \mathrm{d}\mu(\omega) \\ &= \sum_{i,j} a_i a_j \int \zeta_\omega(x_i) \zeta^*_\omega(x_j) \mathrm{d}\mu(\omega) \\ &= \int \left| \sum_i a_i \zeta_\omega(x_i) \right|^2 \mathrm{d}\mu(\omega) \geq 0, \end{aligned} \tag{6.3}$$

$$\psi(0) = \int 1 \mathrm{d}\mu(\omega) = 1. \tag{6.4}$$

$$\begin{aligned} \psi(-x) &= \int \zeta_\omega(-x) \mathrm{d}\mu(\omega) \\ &= \int \zeta^*_\omega(x) \mathrm{d}\mu(\omega) = \psi^*(x). \end{aligned} \tag{6.5}$$

As ψ is real-valued, $\psi(-x) = \psi(x)$, i.e. ψ is symmetric.

See e.g. [11] Theorem 1.1 for the proof of ”only if” part. □

By conbining the case $k(x, x') = \psi(x - x') \neq 1$, we can write k as

$$\begin{aligned} k(x, x') &= \psi(0) \int \zeta_\omega(x - x') \mathrm{d}\mu(\omega) \\ &= \psi(0) \int \zeta_\omega(x) \zeta^*_\omega(x') \mathrm{d}\mu(\omega) \end{aligned} \tag{6.6}$$

Further, if there exists a distribution p such that $p(w)\mathrm{d}\omega = \mathrm{d}\mu(\omega)$ (e.g. μ is absolutely continuous with respect to Lebesgue measure), we can estimate the value of k by sampling from the distribution f

$$k(x, x') \approx \frac{1}{S} \sum_{s=1}^{S} \zeta_{\omega_s}(x) \zeta^*_{\omega_s}(x'), \tag{6.7}$$

where S is the number of samples and $\omega_s \sim p$. Therefore, we can employ $z(x) = \frac{1}{\sqrt{S}} \left[\zeta_{\omega_1}(x), \dots, \zeta_{\omega_S}(x)\right]^T$ as the approximation of the feature function. [1] listed other examples of kernel functions that we can explicitly write and sample from the corresponding distributions.

Remark 6.3. As $\zeta_\omega(x)$ is complex-valued. We usually use $(\cos(\omega^T x), \sin(\omega^T x)) \in \mathbb{R}^2$ instead. We will use both of them interchangeably in this note.

Remark 6.4. For later use, We rewrite $z(x)$ as $z(x) = \frac{1}{\sqrt{S}} \exp(\sqrt{-1}\Omega x)$ where $\Omega = [\omega_1, \dots, \omega_S]$ and exp is applied in a element-wise way.

The Fourier transform and its inverse is defined as

$$\begin{aligned} \mathcal{F}f(\omega) &:= \frac{1}{(\sqrt{2\pi})^D} \int f(x)\zeta_\omega^*(x)\mathrm{d}x \\ \check{\mathcal{F}}g(x) &:= \frac{1}{(\sqrt{2\pi})^D} \int g(\omega)\zeta_\omega(x)\mathrm{d}\omega. \end{aligned} \tag{6.8}$$

where $f, g \in \mathcal{L}^1(\mathbb{R}^D)$. We will write $\hat{f} := \mathcal{F}f$ and $\check{g} := \check{\mathcal{F}}g$ for short. If $\psi \in \mathcal{L}^1(\mathbb{R}^D)$ and its Fourier transform $\hat{\psi}$ is also in $\mathcal{L}^1(\mathbb{R}^D)$, we can use $\hat{\psi}$ to define the measure in the Bochner's theorem. As

$$\begin{aligned} k(x, x') &= \psi(x - x') \\ &= \check{\mathcal{F}}\mathcal{F}f(x - x') \\ &= \int \mathcal{F}f(w)\zeta_w(x - x')\mathrm{d}w \\ &= \int \mathcal{F}f(w)\zeta_w(x)\zeta_w^*(x')\mathrm{d}w, \end{aligned} \tag{6.9}$$

we should take $d\mu(w) \propto \mathcal{F}f(w)\mathrm{d}w$.

Example 6.5. In some specific case, we can explicitly write the functional form of μ and sample from it. Consider RBF kernel $k(x, x') = \exp(-a\|x - x'\|^2)$ for $a > 0$. Where we can explicitly write the Fourier transform of ψ as

$$\hat{\psi}(\omega) = \frac{1}{(\sqrt{2a})^D} \exp\left(-\frac{\omega^2}{4a}\right). \tag{6.10}$$

By properly choosing the normalizing constant, we obtain

$$k(x, x') = \psi(0) \int \zeta_\omega(x)\zeta_\omega^*(x')p(\omega)\mathrm{d}\omega \tag{6.11}$$

where $p(\omega) = \frac{1}{(\sqrt{4\pi a})^D} \exp\left(-\frac{\|\omega\|^2}{4a}\right)$.

Algorithm 1 Random Fourier Feature

```
Require: Kernel function $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$
Require: Sampling number $S$
Ensure: Approximated feature map: $z : \mathbb{R}^D \to \mathbb{R}^S$ s.t. $z(x)^T z(x') \approx k(x - x')$
  Calculate $p(\omega)$ from $k$
  Sample i.i.d. $\omega_s$ from $p$ for $s = 1, \ldots S$
  $z(x) := \frac{1}{\sqrt{S}}[\zeta_{\omega_1}(x), \ldots, \zeta_{\omega_S}(x)]$
```

[5] analyzed the asymptotic behavior of this randomized feature map and proved that the approximated kernel converges to the true kernel on compact set of $\mathbb{R}^D \times \mathbb{R}^D$. [10] proved more sharp convergence rate of this approximated feature map. See [5] Claim 1 and [10] Theorem 1 for detail.

The following proposition shows that using Random Fourier Feature and constructing linear predictor is almost equivalent to restricting search space in RKHS.

Proposition 6.6 ([12] Proposition 1)**.** *Let $\mathcal{H}_{\mathrm{RFF}} = \mathrm{Span}\{S_s, C_s \mid s = 1, \dots, S\}$ where $S_s(x) = \sin(\omega_s^T x)$ and $C_s(x) = \cos(\omega_s^T x)$. The optimization problem*

$$\min_{f \in \mathcal{H}_{\mathrm{RFF}}} \sum_{n=1}^{N} l(f(x_n), y_n) + \lambda \|f\|_{\mathcal{H}}^2 \tag{6.12}$$

is equivalent to the following optimization problem:

$$\min_{w \in \mathbb{R}^{2S}} \sum_{n=1}^{N} l(f_w(x_n), y_n) + \lambda R(w) \tag{6.13}$$

where $f_w(x) = w^T z(x)$ and $R(w) = \sum_{s=1}^{S} w_{2s-1}^2 \sin(\omega_s^T x) + w_{2s}^2 \cos(\omega_s^T x)$.

6.3. **Mercer's Theorem.** There is another justification of this approximation by sampling from the distribution on frequency with Mercer's theorem[4]. The following statement is from [**fukumizu2008**].

Let $(\mathcal{X}, \mathcal{B}, \mu)$ be a measure space where $\mathcal{X}$ is a compact Hausdorff space and μ is a finite measure. Suppose that $L^2(\mathcal{X}, \mu)$ is separable (i.e. it has countable orthogonal basis). Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a continuous reproducing kernel. We define T_k as

$$(T_k f)(x) = \int k(x, x') f(x') \mathrm{d}x' \tag{6.14}$$

Then T_k is a linear bounded operator on $\mathrm{L}^2(\mathcal{X}, \mu)$.

Theorem 6.7 (Mercer's theorem)**.** *There exists an sequence of positive eigenvalues $\{\lambda_i\}_{i=1}^{\infty}$ and eigenfunctions $\{\varphi_i\}_{i=1}^{\infty}$ of T_k such that $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$ and $\lim_{i \to 0} \lambda_i = 0$. Further, $k(x, x')$ can be expanded as*

$$k(x, x') = \sum_{i=1}^{\infty} \lambda_i \phi(x) \phi^*(x') \tag{6.15}$$

where the convergence is absolute and uniform.

We can show that $\{\lambda_i\}$ is in $\ell^2(\mathbb{R})$. Suppose T_k is a Trace-class operator, $\{\lambda_i\}$ is in $\ell^1(\mathbb{R})$, then $p(i) = \frac{\lambda_i}{\|\lambda\|_1}$ defines the probability measure on $\mathbb{N}$. Therefore, by sampling from this distribution, we can approximate the reproducing kernel k as the inner product $k(x, x') \approx z(x)^T z(x')$ where

$$z(x) = \frac{\|\lambda\|_1}{\sqrt{S}} \left[\varphi_{i_1}(x), \dots, \varphi_{i_S}(x)\right]^T \tag{6.16}$$

where $i_s \sim p(i)$ for $s = 1, \dots, S$.

6.4. **Fastfood.** In RFF, we first create random matrix V by sampling each element from Gaussian distribution independently and make approximate feature vector by $z(x) = \frac{1}{\sqrt{S}} \exp(\sqrt{-1} V x)$. Fastfood [3] is a technique of the fast calculation of Random Fourier Feature by devising the calculation of matrix V. We first assume $D = S = 2^L$ for some $L \in \mathbb{N}_{>0}$. Fastfood calculates V by

$$V = \frac{1}{\sigma \sqrt{D}} S H G \Pi H B \tag{6.17}$$

where

- Π is a permutation matrix.
- H is a Walsh-Hadamard matrix.

- G is a diagonal matrix such that $G_{ii} \sim \mathcal{N}(0,1)$.
- S is a diagonal matrix $S_{ii} = s_i \|G\|_F^{-1/2}$ where $p(s_i) \propto s_i^{S-1}\exp(-s_i^2/2)$
- B is a diagonal matrix such that $B_{ii} \sim \{\pm 1\}$ uniformly random.

Walsh-Hadamard matrix is defined inductively as follows:

$$H_1 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, H_{2d} = \begin{bmatrix} H_d & H_d \\ H_d & -H_d \end{bmatrix}. \tag{6.18}$$

We can easily show that $BB^T = I$, $HH^T = DI$, $\Pi\Pi^T = I$. The calculation cost of $z(x)$ is $O(D\log d)$ rather than $O(Dd)$ by making use of FFT-like calulation of Walsh-Hadamard transform (See [3] Lemma 6 for detail).

Theorem 6.8 ([3] Lemma 7)**.** $\mathbb{E}[z^*(x)z(x')] = \exp(-\frac{\|x-x'\|}{2\sigma^2})$

Proof. See [3] Lemma 7. □

7. Optimization of the loss function of kernel regression

In this section, we consider the training of kernel machines. In order to explain fast and efficient learning of kernel machines, known as doubly stochastic functional gradient, we will explain first gradient descent (GD) and stochastic gradient descent (SGD).

7.1. Gradient Descent.

Consider some objective function $L : \mathbb{R}^D \to \mathbb{R}$. Gradient descent is an optimization algorithm in which we iteratively update parameter toward the most steepest direction.

$$\theta \leftarrow \theta - \eta \nabla_\theta L(\theta) \tag{7.1}$$

where $\eta > 0$ is a small constant the detemines the step size of each iteration.

Algorithm 2 Gradient Descent

```
Require: Object function $L : \mathbb{R}^D \to \mathbb{R}$
Require: Initial parameter $\theta_0 \in \mathbb{R}^D$
Require: Step size $\eta_t \in \mathbb{R}$
Require: Time length $T$
Ensure: optimized parameter $\theta$
  $\theta \leftarrow \theta_0$
  for $t = 1, \dots T$ do
    $\theta \leftarrow \theta - \eta_t \nabla_\theta L(\theta)$
  end for
  Return $\theta$ (or $\overline{\theta} = \frac{1}{T}\sum_{t=0}^{T-1}\theta_t$)
```

Remark 7.1. In some context, η is called the *learning rate*.

Remark 7.2. η need not to be constant during whole training process. It can be changed over iterations. i.e. schematically, $\eta = \eta_t$. Typically, we set η_t so that $\lim_{t\to\infty}\eta_t = 0$, $\sum_{t=1}^{\infty}\eta_t = \infty$ and $\sum_{t=1}^{\infty}\eta_t^2 < \infty$ because there are several cases where we can theoretically prove that the algorithm converges to the local (or sometimes global) minimum. See also Robbins-Monro algorithm [6], although it should be categorized as stochastic algorithm, which we will explain in the next section.

The following theorem is one of the theoretical justifications of the gradient descent algorithm.

Theorem 7.3 ([9] Corollary 14.2)**.** *Suppose L is differentiable, convex and $\|\nabla L\| \leq \rho$. Let θ^* be the minimizer of L. Suppose $\|\theta^*\| \leq B$ and $\|\theta_0\| \leq B$. Then, running the gradient descent algorithm with $\eta = \sqrt{\frac{2B^2}{\rho^2 T}}$ yields the output $\overline{\theta}$ with*

$$L(\overline{\theta}) - L(\theta^*) \leq \sqrt{\frac{2}{T}} B\rho. \tag{7.2}$$

Proof. As L is convex,

$$L(\overline{\theta}) - L(\theta^*) = L\left(\frac{1}{T}\sum_{t=0}^{T-1}\theta_t\right) - L(\theta^*) \leq \frac{1}{T}\sum_{t=0}^{T-1}\left(L(\theta_t) - L(\theta^*)\right). \tag{7.3}$$

Again, as L is convex,

$$\begin{aligned} &L(\theta^*) \geq L(\theta_t) + (\theta^* - \theta_t)^T \nabla_\theta L(\theta_t) \\ \Leftrightarrow &L(\theta_t) - L(\theta^*) \leq (\theta_t - \theta^*)^T \nabla_\theta L(\theta_t). \end{aligned} \tag{7.4}$$

The right hand side can be transformed as follows:

$$\begin{aligned} (\theta_t - \theta^*)^T \nabla_\theta L(\theta_t) &= \frac{1}{2\eta} \cdot 2(\theta_t - \theta^*)^T \eta \nabla_\theta L(\theta_t) \\ &= \frac{1}{2\eta}\left(-\|\theta_t - \eta\nabla_\theta L(\theta_t) - \theta^*\|^2 + \|\theta_t - \theta^*\|^2 + \eta^2\|\nabla_\theta L(\theta_t)\|^2\right) \\ &= \frac{1}{2\eta}\left(-\|\theta_{t+1} - \theta^*\|^2 + \|\theta_t - \theta^*\|^2 + \eta^2\|\nabla_\theta L(\theta_t)\|^2\right). \end{aligned} \tag{7.5}$$

By concatenating these formulae, we get

$$\begin{aligned} \sum_{t=0}^{T-1}\left(L(\theta_t) - L(\theta^*)\right) &\leq \frac{1}{2\eta}\sum_{t=0}^{T-1}\left(-\|\theta_{t+1} - \theta^*\|^2 + \|\theta_t - \theta^*\|^2 + \eta^2\|\nabla_\theta L(\theta_t)\|^2\right) \\ &= -\frac{1}{2\eta}\|\theta_{T+1} - \theta^*\|^2 + \frac{1}{2\eta}\|\theta_1 - \theta^*\|^2 + \frac{\eta}{2}\sum_{t=0}^{T-1}\|\nabla_\theta L(\theta_t)\|^2 \\ &\leq \frac{1}{2\eta}\|\theta_1 - \theta^*\|^2 + \frac{\eta}{2}\sum_{t=0}^{T-1}\|\nabla_\theta L(\theta_t)\|^2. \end{aligned} \tag{7.6}$$

As $\|\theta_1 - \theta^*\|^2 \leq 2(\|\theta_1\|^2 + \|\theta^*\|^2) \leq 2B^2$ and $\|\nabla_\theta L(\theta_t)\|^2 \leq \rho^2$,

$$\text{(RHS of 7.6)} \leq \frac{1}{\eta}B^2 + \frac{\eta}{2}T\rho^2 \leq \sqrt{2T}B\rho. \tag{7.7}$$

The second equality holds when $\eta = \sqrt{\frac{2B^2}{\rho^2 T}}$. With (7.3), we get $L(\overline{\theta}) - L(\theta^*) \leq \sqrt{\frac{2}{T}}B\rho$. □

Remark 7.4. The gradient descent works even if L is non-differentiable. In that case, the gradient should be replaced with the subgradient and the condition on the norm of the derivative should be replaced with Lipchitz continuity condition.

7.2. **Stochastic Gradient Descent.** Sometimes it is computationally prohibitive to calculate the derivative $\nabla_\theta L$. For example, suppose L is of the form

$$L(\theta) = \sum_{n=1}^{N} L_\theta(x_n) \tag{7.8}$$

where x_n is n-th sample, then, the derivative with respect to θ is

$$\nabla_\theta L(\theta) = \sum_{n} \nabla_\theta L_\theta(x_n). \tag{7.9}$$

The loss function we consider so far is in this case because we should take L_θ as $l(f_\theta(x_n), y_n)$, where we slightly abuse the notation. The computational cost is typically $O(N)$, which is prohibitive when the size of the data set is enormous (e.g. ImageNet[2] dataset consists of more than 14 million images).

Stochastic Gradient Descent is the algorithm that alleviate this problem by estimating the derivative by computationally cheap method. Specifically, we use the random variable ξ that is easy to calculate and use it as the substitute of the derivative. Typically, the random variable is *unbiased* estimate of the derivative in the sense that the expectation of ξ equals to the derivative of the loss function:

$$\mathbb{E}[\xi] = \nabla_\theta L(\theta). \tag{7.10}$$

Algorithm 3 Stochastic Gradient Descent

```
Require: Objective function $L : \mathbb{R}^D \to \mathbb{R}$
Require: Initial parameter $\theta_0 \in \mathbb{R}^D$
Require: Step size $\eta_t \in \mathbb{R}$
Require: Time length $T$
Ensure: optimized parameter $\theta$
  $\theta \leftarrow \theta_0$
  for $t = 1, \ldots, T$ do
    Sample $\xi$, the estimate of $\nabla_\theta L(\theta)$.
    $\theta \leftarrow \theta - \eta_t \xi$
  end for
```

Example 7.5. Let's consider the previous example

$$L(\theta) = \frac{1}{N} \sum_{n=1}^{N} l(f_\theta(x_n), y_n), \tag{7.11}$$

where we have introduced the normalizer $1/N$ to make the calculation simple.

The derivative is

$$\nabla_\theta L(\theta) = \frac{1}{N} \sum_{n} l'(f_\theta(x_n), y_n) \nabla_\theta f_\theta(x_n) \tag{7.12}$$

Instead of calculate the derivative for all samples (x_n, y_n), we sample the subset of the dataset $\mathcal{B} = \{(x_{n_1}, y_{n_1}), \ldots, (x_{n_B}, y_{n_B})\}$ uniformly random from the dataset and estimate the derivative by

$$\xi = \frac{1}{B} \sum_{b=1}^{B} l'(f_\theta(x_{n_b}), y_{n_b}) \nabla_\theta f_\theta(x_{n_b}). \tag{7.13}$$

It is easy to check that ξ is unbiased estiamte of the derivative of L with respect to the parameter θ: $\mathbb{E}_{\mathcal{B}}[\xi] = \nabla_\theta L(\theta)$. At the extreme, if we sample only single sample from the dataset $\mathcal{D}$, the estimate is

$$\xi = l'(f_\theta(x_n), y_n)\nabla_\theta f_\theta(x_n), \tag{7.14}$$

where $n \sim \mathrm{Unif}([N])$

In order to calculate ξ, it is necessary to calculate the derivative of f_θ with respect to θ for each sample x_n. One of such situations is that f_θ is realized as a neural network whose weights aree parameterized by θ. We can calculate the derivative by the well-known *back propagation* algorithm [7].

Remark 7.6. Sometimes, the random variable ξ is an unbiased estimate of not only the *training* loss but also the *generalization* loss. Suppose we assume that the trainine dataset $\mathcal{D} = \{x_1, \dots, x_N\}$ is sampled i.i.d. from some (unknown) distribution $\mathcal{P}$: $x_n \sim \mathcal{P}$. The training loss L is defined as

$$L(\theta) = \sum_{n=1}^{N} L_\theta(x_n), \tag{7.15}$$

where L_θ is the loss function for one sample, parameterized by θ, while the generalization error $\tilde{L}$ is

$$\tilde{L}(\theta) = \mathbb{E}_{x\sim\mathcal{P}}\left[L_\theta(x)\right] \tag{7.16}$$

The unbiased estimate ξ of the derivative of the (training) loss in the previous example is also the unbiased estimate of the generalization loss as well.

7.3. **Optimization problem in RKHS.**

For the kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, consider again the optimization problem

$$L(f) = \sum_{n=1}^{N} l(f(x_n), y_n) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 \tag{7.17}$$

where $\mathcal{H}$ is the RKHS of k, $l : \mathbb{R} \to \mathcal{R}$, $\lambda > 0$, and $\mathcal{D} = \{(x_1, y_1), \dots, (x_N, y_N))\}$. In order to apply (S)GD to this optimization problem, we need to calculate the derivative of L with respect to f. As this is a functional derivative, we use the Fréchet derivative instead of ordinal derivative.

Definition 7.7. Let $\mathcal{H}$ be a Hilbert space. $\nabla L(f) \in \mathcal{H}$ is the Fréchet derivative of functional $L : \mathcal{H} \to \mathbb{R}$ at f if

$$L(f + \epsilon g) = L(f) + \epsilon \left\langle \nabla L(f), g\right\rangle_{\mathcal{H}} + O(\epsilon^2) \tag{7.18}$$

for all $g \in \mathcal{H}$ and $\epsilon > 0$.

Remark 7.8. We can prove that the Fréchet derivative is unique if it exists.

Remark 7.9. For Banach space $\mathcal{B}$, Frećhet derivative is usually defined as the bounded linear operator on $\mathcal{B}$. When $\mathcal{B}$ is a Hilbert space, we can convert the bounded linear operator to the element of $\mathcal{B}$ by the Riesz's representational theorem.

As $\mathrm{ev}_x(f) = f(x) = \langle f, k(x,\cdot)\rangle_{\mathcal{H}}$, $\nabla f(x)\,(= \nabla \mathrm{ev}_x(f)) = k(x,\cdot)$. Also, as $\|f + \epsilon g\|_{\mathcal{H}}^2 = \|f\|_{\mathcal{H}}^2 + 2\epsilon\left\langle f, g\right\rangle_{\mathcal{H}} + \epsilon^2\|g\|_{\mathcal{H}}^2$, we can show that $\nabla\|f\|_{\mathcal{H}}^2 \left(= \nabla\|\cdot\|_{\mathcal{H}}^2(f)\right) = 2f$.

Therefore,

$$\nabla L(f) = \sum_{n=1}^{N} l'(f(x_n), y_n)k(x_n, \cdot) + \lambda f, \tag{7.19}$$

where l' is the (sub)gradient of l.

Remark 7.10. In fact, we do not have to consider the differentiation on possibly infinite dimensional space $\mathcal{H}$ if we only have to find at least one of the minimizers thanks to the Representer theorem. But we follow [1] and introduce the funtional derivative.

7.4. **Doubly Stochastic Gradient.** Doubly Stochastic Gradient [1] is a technique to apply the gradient descent to the optimization problem on RKHS by sampling from training dataset (as we did in SGD) and from frequency (as we did in Random Fourier Feature). As we did in RFF, we assume that the kernel k is translation-invariant and decompose the kernel function k as

$$k(x, x') = \phi(0)\mathbb{E}_{\omega\sim p}\left[\zeta_\omega(x)\zeta_\omega^*(x')\right] \tag{7.20}$$

from now on, we normalize $k(x, x) = \phi(0) = 1$. We estimate the derivative $\nabla L(f)$ with

$$\xi = l'(f(x), y)\zeta_\omega(x)\zeta_\omega^*. \tag{7.21}$$

It is straight forward to see that $\mathbb{E}_{(x,y)\sim\mathcal{D},\omega\sim p}\left[\xi\right] = \nabla L(f)$.

Remark 7.11. Be aware that in general $\xi \notin \mathcal{H}$. But $\mathbb{E}[\xi] \in \mathcal{H}$ as it equals to $\nabla L(f)$.

Therefore, the naive algorithm is like Algorithm 7.4. But we need more consideration to implement Algorithm 7.4 because it is not obvious how to paramete rize a function f. As we look at the algorithm, we notice that f must be of the form $\sum_{n:finite} \alpha_n \zeta_{\omega_n}^*$ during the optimization process. We also notice that we add the term of the form $\alpha\zeta_\omega$ in each iteration. So we write f of this form and update α's in each iteration.

Algorithm 4 Doubly Stochastic Gradient

```
Require: Objective function $L$
Require: Translation invariant kernel $k$
Require: Step size $\eta_t$
Require: Time length $T$
Ensure: Optimized predictor $f$
  for $t = 1, \dots T$ do
    sample $(x, y)$ uniformly from $\mathcal{D}$
    sample $\omega$ from $p$
    $\xi = \phi(0)l'(f(x), y)\zeta_\omega(x)\zeta_\omega^* + f$
    $f \leftarrow f - \eta_t\xi$
  end for
```

We use $\omega_{t'}$ for $t' < t$ in iteration t to calculate f_x. One of the straight forward way to do so is to memorize all ω_t's. Another way is to to use *pseudo random generator*, which returns same random value for the same seed. The drawback of this algorithm is that its complexity is $O(T^2)$ as we have to calculate f_x in every iteration, which takes $O(t)$ in interation t.

Algorithm 5 Doubly Stochastic Gradient

```
Require: Objective function $L$
Require: Translation invariant kernel $k$
Require: Step size $\eta_t$
Require: Time length $T$
Ensure: Optimized predictor $f$
  for $t = 1, \dots T$ do
    sample $(x, y)$ uniformly from $\mathcal{D}$
    sample $\omega_t$ from $p$
    $f_x = \sum_{t'=1}^{t-1} \alpha_t \zeta^*_{\omega_t}$
    $\alpha_t \leftarrow -\phi(0) l'(f_x, y) \zeta_\omega(x)$
    $\alpha_{t'} \leftarrow (1 - \eta_t)\alpha_{t'}$   for $1 \leq t' \leq t$
  end for
  Return $f = \sum_{t=1}^{T} \alpha_t \zeta^*_{\omega_t}$
```

References

[1] Bo Dai et al. “Scalable kernel methods via doubly stochastic gradients”. In: *Advances in Neural Information Processing Systems*. 2014, pp. 3041–3049.

[2] J. Deng et al. “ImageNet: A Large-Scale Hierarchical Image Database”. In: *CVPR09*. 2009.

[3] Quoc Le, Tamas Sarlos, and Alex Smola. “Fastfood - Approximating Kernel Expansions in Loglinear Time”. In: *30th International Conference on Machine Learning (ICML)*. 2013. URL: `http://jmlr.org/proceedings/papers/v28/le13.html`.

[4] James Mercer. “Functions of positive and negative type, and their connection with the theory of integral equations”. In: *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character* 209 (1909), pp. 415–446.

[5] Ali Rahimi and Benjamin Recht. “Random features for large-scale kernel machines”. In: *Advances in neural information processing systems*. 2007, pp. 1177–1184.

[6] Herbert Robbins and Sutton Monro. “A stochastic approximation method”. In: *The annals of mathematical statistics* (1951), pp. 400–407.

[7] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. “Learning representations by back-propagating errors”. In: *Cognitive modeling* 5.3 (), p. 1.

[8] Dino Sejdinovic and Arthur Gretton. “What is an RKHS?” In: 2014. URL: `http://www.stats.ox.ac.uk/~sejdinov/teaching/atml14/Theory_2014.pdf`.

[9] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

[10] Bharath Sriperumbudur and Zoltán Szabó. “Optimal rates for random Fourier features”. In: *Advances in Neural Information Processing Systems*. 2015, pp. 1144–1152.

[11] Rongfeng Sun. In: URL: `http://www.math.nus.edu.sg/~matsr/ProbI/Lecture7.pdf`.

[12] Tianbao Yang et al. “Nyström method vs random fourier features: A theoretical and empirical comparison”. In: *Advances in neural information processing systems*. 2012, pp. 476–484.

E-mail address: `k.oono.delta@gmail.com`