Kernel Methods
in
Machine Learning and Statistics

Nikolas Nüsken
Marina Riabiz

King's College London
Department of Mathematics

February 20, 2023

Contents

These notes are based on material developed by Dr. Nikolas Nüsken and Dr. Marina Riabiz. We would appreciate if you point out any typos to:
nikolas.nusken@kcl.ac.uk
marina.riabiz@kcl.ac.uk.

Chapter 1

Kernels and RKHSs

1.1 Motivation

Let us consider the following situation ('regression problem'): We are given data $(x_i, y_i)_{i=1}^N \subset X \times \mathbb{R}$ and aim to find a function $f : X \to \mathbb{R}$ that 'explains' it. Importantly, we would also like f to give reasonable predictions on unseen data. A central topic is the dichotomy between *overfitting* and *underfitting*.

Complexity. If f is too simple, it will underfit, if f is too complex, it will overfit.

Deterministic vs. stochastic effects. If we underestimate the noise in the data, it will lead to overfitting. If we underestimate the noise, it will lead to underfitting.

Training vs prediction. Overfitting means performing well on training data, and badly on test data. Underfitting means performing badly on training data, but ok on test data.

Finding a 'sweat spot' between over- and underfitting is a key problem in machine learning and statistics. Typical approach: Specify a function class $\mathcal{F}$ in advance (e.g. $\mathcal{F} = \{\text{polynomials}\}$, $\mathcal{F} = \{\text{neural networks}\}$), and find $f \in \mathcal{F}$. We would also like a map $c : \mathcal{F} \to \mathbb{R}_{\geq 0}$ that measures the 'complexity' of $f \in \mathcal{F}$.

This course:

- $\mathcal{F} = \mathrm{RKHS}_k$ (reproducing kernel Hilbert space associated to the kernel k,
- $c = \|\cdot\|_{\mathcal{H}_k}$, norm on this space.

We will take the regression problem as a starting point, but touch on various other topics.

Further reading: In this course, we will not discuss background in the (beautiful!) theory of statistical learning (overfitting vs underfitting) very much. Have a look at the classic paper [5] or the modern treatment in [2, Chapter 2].

1.2 Positive definite kernels

In this section, we give the notion of positive definite kernels, as well as a few examples. Take this section as a prelude to Section 1.3; we will use positive definite kernels as the building blocks for the associated reproducing kernel Hilbert spaces.

Definition 1 (Positive definite kernels)**.** Let $X \neq \emptyset$ be a set. A function $k : X \times X \to \mathbb{R}$ is called a *positive definite kernel* if

1. k is *symmetric*, that is,

$$k(x,y) = k(y,x) \qquad \text{for all } x, y \in X, \tag{1.2.1}$$

2. k is *positive semi-definite*: For all $n \in \mathbb{N}$, $\alpha_1, \dots, \alpha_n \in \mathbb{R}$ and $x_1, \dots, x_n \in X$, we have

$$\sum_{i,j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) \geq 0. \tag{1.2.2}$$

Remark 1.2.1. *A few remarks:*

1. *Positive definiteness of kernels in the sense of Definition 1 is equivalent to the following statement: All matrices of the form*

$$\begin{pmatrix} k(x_1,x_1) & \dots & k(x_1,x_n) \\ \vdots & \ddots & \vdots \\ k(x_n,x_1) & \dots & k(x_n,x_n) \end{pmatrix} \tag{1.2.3}$$

   are positive semi-definite, for arbitrary choices of $x_1, \dots, x_n \in X$ and 'sizes' $n \in \mathbb{N}$. It is therefore reasonable to think of positive definite kernels as positive semi-definite matrices 'of arbitrary dimension'. The matrices in (1.2.3) are commonly referred to as Gram matrices.

2. *Intuition: Sometimes (not always) it is useful to think of $k(x,y)$ as a measure of* similarity *between x and y.*

3. *The property $k(x,y) \geq 0$ for all $x, y \in X$ is neither necessary nor sufficient for the second statement in Definition 1 to hold (this follows from the equivalent statement in 1. above – think about matrices!)*

4. *There is no structural assumption on X (allowing for its elements to be strings, images...), but often we will have $X = \mathbb{R}^d$.*

5. *$k : X \times X \to \mathbb{C}$ is interesting, but not in this course.*

6. *Verifying the first condition in Definition 1 is usually straightforward, proving the second condition may be difficult.*

Example 1.2.1. We will often use the following positive definite kernels:

1. **Squared-exponential (or Gaussian) kernel**:

$$k(x,y) = \exp\left(-\frac{|x-y|^2}{\sigma^2}\right), \qquad x, y \in \mathbb{R}^d. \tag{1.2.4}$$

   Here, $\sigma > 0$ specifies the width of the kernel.

2. **Laplace kernel:**

$$k(x,y) = \exp\left(\frac{|x-y|}{\sigma}\right), \qquad x, y \in \mathbb{R}^d. \tag{1.2.5}$$

3. More generally, we can consider the family of **p-kernels**,

$$k(x,y) = \exp\left(\frac{|x-y|^p}{\sigma^p}\right), \qquad x, y \in \mathbb{R}^d, \tag{1.2.6}$$

   where $p \in [1, 2]$ is necessary to guarantee positive definiteness.

4. **Polynomial kernels:**
$$k(x,y) = \left(x^\top y + c\right)^m, \qquad x, y \in \mathbb{R}^d, \tag{1.2.7}$$
with $c > 0$ and $m \in \mathbb{N}$.

5. **Sums and products:** Given two positive definite kernels $k_1, k_2 : X \times X \to \mathbb{R}$, their sum and product
$$(k_1 + k_2)(x,y) := k_1(x,y) + k_2(x,y) \tag{1.2.8a}$$
$$(k_1 \cdot k_2)(x,y) := k_1(x,y) \cdot k_2(x,y) \tag{1.2.8b}$$
are also positive definite. While this statement is straightforward to show for the sum kernel, to prove it for the product kernel we need the *Schur product theorem* (if you don't know it, look it up on wikipedia).

6. **Scalings:** Given a positive definite kernel $k : X \times X \to \mathbb{R}$ and a function $f : X \to \mathbb{R}$, the *rescaled kernel*
$$k_f(x,y) = f(x)k(x,y)f(y). \tag{1.2.9}$$
is positive definite as well.

Exercise 1. *Show that the Gaussian kernel in* (1.2.4) *is indeed positive definite. Hint: you may find it useful to write*
$$k(x,y) = \exp\left(-\frac{|x|^2}{\sigma^2}\right)\exp\left(\frac{2}{\sigma^2}x \cdot y\right)\exp\left(-\frac{|y|^2}{\sigma^2}\right), \tag{1.2.10}$$
use the power series expansion for the exponential function, as well as the properties in (1.2.8).

The following lemma gives some further information on possible 'shapes' of positive definite kernels.

Lemma 1. *For a positive definite kernel $k : X \times X \to \mathbb{R}$ the following hold:*

1. *The kernel k is nonnegative on the diagonal:*
$$k(x,x) \geq 0, \qquad \text{for all } x \in X. \tag{1.2.11}$$

2. *The kernel k is 'diagonally dominant':*
$$k(x,y)^2 \leq k(x,x)k(y,y), \qquad \text{for all } x, y \in X. \tag{1.2.12}$$

Proof. The first statement follows from the second item in Definition 1, with $n = 1$. For the second statement consider the same item with $n = 2$. We obtain that the matrices
$$\begin{pmatrix} k(x,x) & k(x,y) \\ k(y,x) & k(y,y) \end{pmatrix} \tag{1.2.13}$$
are positive semidefinite, therefore, have nonnegative determinant, $k(x,x)k(y,y) - k(x,y)^2 \geq 0$. This implies the statement. □

Remark 1.2.2. *We will later see that $k(x,y)$ encodes an inner product between (transformed versions of) x and y. Then,* (1.2.12) *is precisely the Chauchy-Schwarz inequality.*

1.3 Reproducing kernel Hilbert spaces (RKHSs)

In this section we construct reproducing kernel Hilbert spaces, using positive definite kernels as building blocks. We follow [10, Section 2.3].

Given a positive definite kernel $k : X \times X \to \mathbb{R}$, the reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$ is a Hilbert space[1] of functions on X, associated to k in a canonical way. Notice that for any $x \in X$, we obtain a function $k(x, \cdot) : X \to \mathbb{R}$. This family of functions (varying $x \in X$) provides the building blocks for $\mathcal{H}_k$:

$$''\mathcal{H}_k = \{\text{limits of linear combinations of functions of the form } k(x,\cdot), \text{for } x \in X\}'' . \tag{1.3.1}$$

In the remainder of this section, we will make the heuristic (1.3.1) precise. Notice that it is not yet clear in which sense the limit in (1.3.1) should be taken, and that in order to obtain a Hilbert space, we will also need to construct an inner product. These two remarks are connected: we will take the limits with respect to the norm induced by the inner product.

Remark 1.3.1. *Functions of the form $k(x, \cdot)$ are particularly intuitive if the kernel is* translation-invariant*, such as those in (1.2.4)-(1.2.6); in this case, those are translates by x of a basic kernel (centred at the origin).*

Construction. *Step 1:* Let us fix a positive definite kernel $k : X \times X \to \mathbb{R}$ and define

$$\mathcal{H}_k^0 := \operatorname{span}\{k(x,\cdot) : \quad x \in X\} \tag{1.3.2a}$$

$$:= \left\{ f = \sum_{i=1}^{n} c_i k(x,\cdot) : \quad n \in \mathbb{N},\, c_1,\dots,c_n \in \mathbb{R},\, x_1,\dots,x_n \in X \right\}. \tag{1.3.2b}$$

Clearly, $\mathcal{H}_k^0$ is a vector space over $\mathbb{R}$: it is closed under summation and scalar multiplication.

Step 2: We now construct an inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}_k}$ (and induced norm $\|\cdot\|_{\mathcal{H}_k}$) on $\mathcal{H}_k^0$. To do this, we demand that $\langle\cdot,\cdot\rangle_{\mathcal{H}_k}$ satisfies the *reproducing property*

$$f(x) = \langle f, k(x,\cdot)\rangle_{\mathcal{H}_k}, \qquad \text{for all } f \in \mathcal{H}_k^0,\, x \in X. \tag{1.3.3}$$

It is hard to overstate the significance of the key relation (1.3.3) for the whole subject! We will discuss its full meaning further below, but for now note that $k(x,\cdot)$ plays a similar role as the Dirac delta distribution (for the inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}_k}$).

The reproducing property (1.3.3) determines $\langle\cdot,\cdot\rangle_{\mathcal{H}_k}$ on $\mathcal{H}_0$. Indeed, for $f, g \in \mathcal{H}_k^0$ given in the form

$$f = \sum_{i=1}^{n} a_i k(x_i,\cdot), \qquad g = \sum_{j=1}^{m} b_j k(y_j,\cdot), \tag{1.3.4}$$

we can compute

$$\langle f, g\rangle_{\mathcal{H}_k} = \sum_{i=1}^{n}\sum_{j=1}^{m} a_i b_j \langle k(x_i,\cdot), k(y_j,\cdot)\rangle_{\mathcal{H}_k} = \sum_{i=1}^{n}\sum_{j=1}^{m} a_i b_j k(x_i, y_j), \tag{1.3.5}$$

which from now on serves as our definition of $\langle\cdot,\cdot\rangle_{\mathcal{H}_k}$. Note that in the first equality, we have used the (assumed) linearity of $\langle\cdot,\cdot\rangle_{\mathcal{H}_k}$, and in the second equality we have used the reproducing property (1.3.3) with $f = k(y,\cdot)$. As usual, the inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}_k}$ induces a corresponding norm

$$\|f\|_{\mathcal{H}_k}^2 = \langle f, f\rangle_{\mathcal{H}_k}, \qquad f \in \mathcal{H}_k^0. \tag{1.3.6}$$

But before proceeding, we need to check that the definition of the inner product is valid:

[1] Recall that a Hilbert space is a vector space equipped with an inner product, complete with respect to the induced norm, see Wikipedia.

Lemma 2. *The pairing $\langle \cdot, \cdot \rangle_{\mathcal{H}_k}$ is indeed an inner product on $\mathcal{H}_k^0$.*

Proof. First, we need to check *linearity* and *symmetry*, that is,

$$\text{(linearity)} \qquad \langle f+g, h\rangle_{\mathcal{H}_k} = \langle f, h\rangle_{\mathcal{H}_k} + \langle g, h\rangle_{\mathcal{H}_k}, \tag{1.3.7a}$$

$$\langle \lambda f, g\rangle_{\mathcal{H}_k} = \lambda \langle f, g\rangle_{\mathcal{H}_k}, \qquad f, g, h \in \mathcal{H}_k^0, \quad \lambda \in \mathbb{R}, \tag{1.3.7b}$$

and

$$\text{(symmetry)} \qquad \langle f, g\rangle_{\mathcal{H}_k} = \langle g, f\rangle_{\mathcal{H}_k}, \qquad g, f \in \mathcal{H}_k^0. \tag{1.3.8a}$$

These two properties are straightforward to check from the definition (1.3.5), and we leave it as a (simple) exercise. We hence turn to positive-definiteness and non-degeneracy of $\langle \cdot, \cdot \rangle_{\mathcal{H}_k}$. Recall that we need to show that

$$\text{(positivity)} \qquad \|f\|_{\mathcal{H}_k}^2 := \langle f, f\rangle_{\mathcal{H}_k} \geq 0, \qquad \text{for all } f \in \mathcal{H}_k^0, \tag{1.3.9}$$

as well as

$$\text{(nondegeneracy)} \qquad \langle f, f\rangle_{\mathcal{H}_k} = 0 \quad \text{if and only if} \quad f = 0. \tag{1.3.10}$$

To show (1.3.9), take $f = \sum_{i=1}^n a_i k(x_i, \cdot)$. By definition,

$$\langle f, f\rangle_{\mathcal{H}_k} = \sum_{i,j=1}^n a_i a_j k(x_i, x_j) \geq 0, \tag{1.3.11}$$

where the inequality follows from the fact that k is positive definite (see Definition 1). To show (1.3.10), assume that $f \in \mathcal{H}_k^0$ is such that $\langle f, f\rangle_{\mathcal{H}_k} = 0$. Then

$$|f(x)|^2 = |\langle f, k(x, \cdot)\rangle_{\mathcal{H}_k}|^2 \leq \|f\|_{\mathcal{H}_k^2} \|k(x, \cdot)\|_{\mathcal{H}_k}^2 = 0, \tag{1.3.12}$$

for all $x \in X$, so that indeed, $f = 0$.[2] □

Remark 1.3.2. *We have cheated a little bit. Defining the inner product in (1.3.5), we should convince ourselves that $\langle f, g\rangle_{\mathcal{H}_k}$ does not depend on the representations of f and g. This means the following. Suppose that*

$$f = \sum_{i=1}^n a_i k(x_i, \cdot) = \sum_{i=1}^{\widetilde{n}} \widetilde{a}_i k(\widetilde{x}_i, \cdot), \qquad g = \sum_{j=1}^m b_j k(y_j, \cdot) = \sum_{j=1}^{\widetilde{m}} \widetilde{b}_j k(\widetilde{y}_j, \cdot), \tag{1.3.13}$$

that is, f and g admit alternative representations in terms of coefficients $\widetilde{a}_i, \widetilde{b}_j$ and points $\widetilde{x}_i, \widetilde{y}_j$. Then, we should have that

$$\sum_{i=1}^n \sum_{j=1}^m a_i b_j k(x_i, y_j) = \sum_{i=1}^{\widetilde{n}} \sum_{j=1}^{\widetilde{m}} \widetilde{a}_i \widetilde{b}_j k(\widetilde{x}_i, \widetilde{y}_j), \tag{1.3.14}$$

in order for $\langle f, g\rangle_{\mathcal{H}_k}$ to be well defined (this is true, but we skip the proof).

Remark 1.3.3 (Terminology and definiteness)**.** *Notice that in Definition 1, the condition (1.2.2) only asks for positive* semi-*definiteness (and still the kernel is called* positive-definite*). This is because, maybe surprisingly, the condition (1.2.2) suffices for the strict posite-definiteness of $\langle \cdot, \cdot \rangle_{\mathcal{H}_k}$, as we have just proved.*

[2] In (1.3.12), we have used the Cauchy-Schwarz inequality. Note, however, that at this stage we do not yet know that $(\mathcal{H}_k^0, \langle \cdot, \cdot \rangle_{\mathcal{H}_k})$ is an inner product space, so is Cauchy-Schwarz valid? Luckily, Cauchy-Schwarz does not require the non-degeneracy condition (1.3.10), see, for instance, https://math.stackexchange.com/questions/2548494/does-cauchy-schwarz-inequality-depend-on-positive-definiteness.

Step 3: Lemma 2 shows that $(\mathcal{H}_k^0, \langle\cdot,\cdot\rangle_{\mathcal{H}_k})$ is a *pre-Hilbert space*: It satisfies all the axioms for Hilbert spaces, except for completeness. We can now employ a standard technique in analysis, and pass to the completion

$$\mathcal{H}_k := \overline{\mathcal{H}_k^0}. \tag{1.3.15}$$

The completion construction (heuristically[3]) includes limit points: for example, the real numbers $\mathbb{R}$ can be constructed by completing the rationals $\mathbb{Q}$. In our setting, the completion amounts to taking $n, m \to \infty$ in (1.3.2b) and (1.3.5). For more details on completions, we refer to [11, Section 1.6, Theorem 3.2-3].

Remark 1.3.4. *The construction of $\mathcal{H}_k$ from k makes it intuitive that the functions $f \in \mathcal{H}_k$ inherit many properties from k (consider again the heuristic (1.3.1)). Indeed, the following statements are true:*

$$k \text{ is bounded} \iff \forall f \in \mathcal{H}_k,\ f \text{ is bounded}. \tag{1.3.16a}$$

$$k \text{ is bounded, continuous} \iff \forall f \in \mathcal{H}_k,\ f \text{ is bounded and continuous}. \tag{1.3.16b}$$

Similar results hold for differentiability, measurability, etc, see, for example, [19, Section 4.3].

Here is another important result about Gaussian RKHSs that can be understood 'intuitively' using the construction from this section:

Lemma 3 (Comparison of Gaussian RKHSs)**.** *Recall the Gaussian kernel from (1.2.4), here denoted by k_σ. For two different widths $\sigma_2 > \sigma_1 > 0$, the following (continuous)[4] inclusion holds,*

$$\mathcal{H}_{k_{\sigma_2}} \subset \mathcal{H}_{k_{\sigma_1}}. \tag{1.3.17}$$

Proof. See [19, Proposition 4.46]. Instead of a proof, we discuss an intuition: The kernel k_{σ_1} is more flexible (because it has a smaller width), and so more functions can be built with it. □

A word of CAUTION! We have introduced the inner product/norm as in (1.3.5). It is a very common mistake to write the RKHS-norm between $f, g \in \mathcal{H}_k$ as

$$\int_X \int_X f(x)k(x,y)g(y)\rho(\mathrm{d}x)\rho(\mathrm{d}y), \tag{1.3.18}$$

for some measure ρ on X (equipped with an appropriate σ-algebra). This is wrong! In fact, it is usually very difficult to compute the RKHS-norm or inner product for f and g explicitly given in functional form. This is because we would need to find the coefficients a_i in $f = \sum a_i k(x_i, \cdot)$ first.

Remark 1.3.5. *The expression (1.3.18) does have a role to play in the subject and using integrals of this form is often useful. This is because (1.3.18) is the dual norm associated to $\|\cdot\|_{\mathcal{H}_k}$, under identification through $L^2(\rho)$. More on this (perhaps/probably) in later chapters.*

1.4 Function evaluations, and the interpretation of $\|\cdot\|_{\mathcal{H}_k}$

The reproducing property (1.3.3) connects function evaluations with the inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}_k}$, and hence with the norm $\|\cdot\|_{\mathcal{H}_k}$. We have the following heuristic:

If two functions are close with respect to $\|\cdot\|_{\mathcal{H}_k}$, then they are close pointwise.

The precise statement is as follows:

[3] Rigorously, the completion is a space of equivalence classes of Cauchy sequences, capturing the notion of limits.

[4] In fact, we have $\|f\|_{\mathcal{H}_{k_{\sigma_1}}} \leq \left(\frac{\sigma_2}{\sigma_1}\right)^d \|f\|_{\mathcal{H}_{k_{\sigma_2}}}$, for all $f \in \mathcal{H}_{k_{\sigma_2}}$.

Lemma 4 (Continuity of point evaluations)**.** *For all $x \in X$, there exists a constant $C_x > 0$ such that*

$$|f(x) - g(x)| \leq C_x \|f - g\|_{\mathcal{H}_k}, \tag{1.4.1}$$

for all $f, g \in \mathcal{H}_k$.

Remark 1.4.1. *As a corollary, we see that if a sequence of functions f_n converges in $\mathcal{H}_k$, then it also converges pointwise. If k is bounded, then we can choose the constant C_x in (1.4.1) independently of x. In this case, convergence of f_n in $\mathcal{H}_k$ even implies uniform convergence.*

Proof. Similar to (1.3.12), we write

$$|f(x) - g(x)|^2 = |\langle f - g, k(x, \cdot)\rangle_{\mathcal{H}_k}|^2 \leq \underbrace{\|k(x,\cdot)\|^2_{\mathcal{H}_k}}_{C_x^2} \|f - g\|^2_{\mathcal{H}_k}, \tag{1.4.2}$$

using the reproducing property (1.3.3) as well as Cauchy-Schwarz. □

We say that *function evaluations are continuous with respect to* $\|\cdot\|^2_{\mathcal{H}_k}$. Indeed, for $x \in X$, we can define the evaluation functional

$$\delta_x : \mathcal{H}_k \to \mathbb{R}, \tag{1.4.3a}$$
$$f \mapsto f(x). \tag{1.4.3b}$$

It follows directly from the definition that δ_x is linear (for all $x \in X$), and, by Lemma 4, δ_x is continuous.

Counterexample 1. Consider the L^2-norm, defined by

$$\|f\|^2_{L^2([0,1])} = \int_0^1 f^2 \,\mathrm{d}x, \tag{1.4.4}$$

as well as the sequence of functions $(q_n)_{n\in\mathbb{N}}$, defined by

$$q_n(x) = x^n, \qquad x \in [0, 1]. \tag{1.4.5}$$

We see that $q_n \to 0$ in $L^2([0, 1])$, that is,

$$\lim_{n\to\infty} \|q_n - 0\|_{L^2([0,1])} = 0. \tag{1.4.6}$$

However, we also have that $q_n(1) = 1$, for all $n \in \mathbb{N}$. Therefore, an estimate of the form

$$|f(x) - 0| \leq C\|f - 0\|_{L^2([0,1])} \tag{1.4.7}$$

cannot hold (for a fixed constant C): function evaluation is not continuous with respect to $L^2([0, 1])$.[5]

We have seen now that point evaluations (evaluation functionals) are continuous in RKHSs (to reiterate, this is mainly a property of the norm $\|\cdot\|_{\mathcal{H}_k}$). Perhaps surprisingly, this property *characterises* the class of RKHSs:

Theorem 1.4.1. *Let $X \neq \emptyset$ be a set, and $(H, \langle\cdot,\cdot\rangle_H)$ be a Hilbert space of functions $f : X \to \mathbb{R}$. Then $(H, \langle\cdot,\cdot\rangle_H)$ is an RKHS[6] if and only if all point evaluations are continuous.*

[5] Strictly speaking, function evaluation is not *allowed* in $L^2([0, 1])$, see Section **??**.

[6] We say that $(H, \langle\cdot,\cdot\rangle_H)$ is an RKHS if there exists a positive definite kernel $k : X \times X \to \mathbb{R}$ such that the construction from Section 1.3 gives rise to $(H, \langle\cdot,\cdot\rangle_H)$ (more precisely, if $(\mathcal{H}_k, \langle\cdot,\cdot\rangle_{\mathcal{H}_k})$ constructed this way is isometrically isomorphic to $(H, \langle\cdot,\cdot\rangle_H)$).

Proof. (Sketch). We have already established "$\Longrightarrow$". For the other direction, assume that $(\mathcal{H}_k, \langle\cdot,\cdot\rangle_{\mathcal{H}_k})$ is a Hilbert space of functions on X so that all function evaluations are continuous. We now construct a kernel $k : X \times X \to \mathbb{R}$ as follows: For $x \in X$, the evaluation functional δ_x defined in (1.4.3) is continuous by assumption. From the Riesz' representation theorem[7], there exist unique $g_x \in H$ such that

$$\delta_x(f) = f(x) = \langle g_x, f\rangle_H, \tag{1.4.8}$$

for all $f \in H$. Using these *representers* g_x, we define

$$k(x,y) := \langle g_x, g_y\rangle_H, \qquad x, y \in X. \tag{1.4.9}$$

Is k defined in this way a positive definite kernel (see Definition 1)? First, it is clear that k is symmetric. To check positive (semi-)definiteness, choose $x_1, \ldots, x_n \in X$ and $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ and compute

$$\sum_{i,j=1}^n \alpha_i\alpha_j k(x_i,x_j) = \sum_{i,j=1}^n \left\langle \alpha_i g_{x_i}, \alpha_j g_{x_j}\right\rangle_H = \left\langle \sum_{i=1}^n \alpha_i g_{x_i}, \sum_{j=1}^n \alpha_j g_{x_j}\right\rangle_H \geq 0, \tag{1.4.10}$$

by positivity of $\langle\cdot,\cdot\rangle_H$. Therefore, k is indeed a positive definite kernel.

We now claim that if we proceed with the construction from Section 1.3 to construct $\mathcal{H}_k, \langle\cdot,\cdot,\rangle_{\mathcal{H}_k}$, we will get back to $(H, \langle\cdot,\cdot\rangle_H)$. Notice that the 'only choice' in this construction is the reproducing property (1.3.3), everything else is determined from that. It is thus sufficient to check that (1.3.3) holds. For this, we observe that the representers g_x can be written in terms of the kernel,

$$g_x = k(x,\cdot), \qquad x \in X. \tag{1.4.11}$$

Indeed, we have

$$k(x,y) = \langle g_x, g_y\rangle_H = \delta_y(g_x) = g_x(y), \qquad x, y \in X, \tag{1.4.12}$$

proving (1.4.11). We can now use (1.4.11) to check (1.3.3),

$$\langle f, k(x,\cdot)\rangle_H = \langle f, g_x\rangle_H = \delta_x(f) = f(x). \tag{1.4.13}$$

Therefore, $\langle\cdot,\cdot\rangle_H$ satisfies the reproducing property for k, and the result follows. □

Remark 1.4.2. *Recall Section 1.1. When designing machine learning/statistical methodology, we often need to chose a function class $\mathcal{F}$. It is reasonable to ask for continuous point evaluations. After all, we will be interested in predictions (= point evaluations), and if there is no continuity, the method will not be robust. It is also convenient to work with Hilbert spaces: the inner product will allow us to calculate a lot explicitly. Theorem 1.4.1 shows that these two desiderata con only be satisfied by reproducing kernel Hilbert spaces, that is, we are automatically forced to use the construction in Section 1.3!*

Let us take stock for a moment and summarise what we have seen so far in definitions. If you look at textbooks, you may find any of the following:

Definition 2 (RKHS 1)**.** A reproducing kernel Hilbert space is one that can be obtained from a positive definite kernel $k : X \times X \to \mathbb{R}$ by the construction in Section 1.3.

Definition 3 (RKHS 2)**.** Given a positive definite kernel $k : X \times X \to \mathbb{R}$ on a set X, the corresponding reproducing kernel Hilbert space $\mathcal{H}_k$ is characterised by the following two properties:

1. For all $x \in X$, we have $k(x,\cdot) \in \mathcal{H}_k$.
2. The reproducing property holds:

$$f(x) = \langle f, k(x,\cdot)\rangle_{\mathcal{H}_k}, \qquad \text{for all } x \in X,\ f \in \mathcal{H}_k. \tag{1.4.14}$$

[7] Note that the Riesz representation theorem crucially requires δ_x to be continuous!

Definition 4 (RKHS 3)**.** A Hilbert space of functions is an RKHS if all point evaluations are continuous.

Of these, clearly the last definition is the most succinct, but also the most mysterious. When we discuss the kernel trick, we will encounter another equivalent definition (and viewpoint).

Remark 1.4.3. *It is an intriguing question whether there can be a Hilbert space of functions where point evaluations are* not *continuous. Apparently, the Counterexample 1 suggests that $L^2((0,1))$ would be such an example. Recall, however, that this is in fact a Hilbert space of equivalence classes of functions (we need to factor out almost everywhere equivalence). In that sense, L^2 is neither an example nor a counterexample for a function space with discontiuous point evaluations. Indeed, it can be shown that the construction of a Hilbert space of functions with discontinuous point evaluations requires the* Axiom of Choice*![8] In this sense, there is no 'counterexample'.*

1.4.1 The norm $\|\cdot\|_{\mathcal{H}_k}$ as a measure of complexity.

In Section 1.1, we promised that $\|f\|_{\mathcal{H}_k}$ would measure the 'complexity' of $f \in \mathcal{H}_k$, in an appropriate sense. Here, we make this statement (a bit) more precise. We are going to argue that the fact that $\|\cdot\|_{\mathcal{H}_k}$ makes point evaluations continuous (see Theorem 1.4.1) expresses the idea that $\|\cdot\|_{\mathcal{H}_k}$ measures complexity.

Back to Counterexample 1: What goes wrong? The L^2-norm (which does not make point evaluations continuous, hence is not an RKHS norm) defined in (1.4.4) only measures the magnitude of function values. This is not sufficient to prevent the wild behaviour close to the boundary of the interval of the sequence q_n. The 'complexity' of q_n, for n large is not appropriately taken into account. We get the feeling that a proper measure of complexity should involve both function values and derivatives.

Indeed, the following heuristic is often[9] true:

The norm $\|f\|_{\mathcal{H}_k}$ measures both the magnitude of f, as well as (some) of its derivatives.

To discuss this a little bit, recall the definition of Sobolev spaces,

$$H^s(\mathbb{R}^d) := \left\{ f \in L^2(\mathbb{R}^d) : \|f\|^2_{H^s(\mathbb{R}^d)} := \sum_{\alpha \in \mathbb{N}_0^d, |\alpha| \leq s} \|D^\alpha f\|^2_{L^2(\mathbb{R}^d)} < \infty \right\}, \tag{1.4.15}$$

that is, $\|f\|^2_{H^s(\mathbb{R}^d)}$ is sensitive to the magnitude of both f and its derivatives. The *Sobolev embedding theorem* tells us that $H^s(\mathbb{R}^d)$ is a space of continuous[10] functions (in particular, with continuous point evaluations!) if and only if $s > \frac{d}{2}$. From this, we deduce the following:

1. By definition 4, the (classical) spaces $H^s(\mathbb{R}^d)$ are reproducing kernel Hilbert spaces if s is large enough (in comparison with the dimension). Indeed, their reproducing kernels are given by the Matérn kernels (see [10, Example 2.2] or Wikipedia).

2. Folklore wisdom: Reproducing kernel Hilbert spaces necessarily become 'small' in high dimensions. Indeed, for $H^s(\mathbb{R}^d)$ to be an RKHS for large d, its members have to be very smooth. This is a concern for kernel methods in high-dimensional settings. More on this later (neural networks, etc.).

[8] More precisely, the existence of a Hilbert space of functions with discontinuous point evaluations is independent of the ZF axiomatic system without the axiom of choice. See https://math.stackexchange.com/questions/2689457/example-of-an-infinite-dimensional-hilbert-space-that-is-not-an-rkhs.

[9] This claim mainly makes sense for $X = \mathbb{R}^d$

[10] Note that D^α in (1.4.15) refers to the *weak* derivative. In particular, f does not need to be continuous a priori.

Remark 1.4.4 (The size of the Gaussian RKHS)**.** *It is interesting to know that the Gaussian RKHS is 'very small'. Indeed, it does not contain constant functions (apart from the zero function), and all members are real-analytic.*

Remark 1.4.5 (Very small RKHSs: the finite-dimensional case)**.** *It is a common misconception that RKHSs have to be infinite-dimensional, since they are spaces of functions. This is not true. Consider for example the kernel*

$$k(x, y) = f(x)f(y), \tag{1.4.16}$$

for some function $f : X \to \mathbb{R}$. All of the functions $k(x, \cdot)$ are the same, up to a multiplicative constant! Therefore, the RKHS associated to (1.4.16) is one-dimensional,

$$\mathcal{H}_k = \{\alpha f : \quad \alpha \in \mathbb{R}\}. \tag{1.4.17}$$

We can build RKHSs of arbitrary (finite dimension) by

$$k(x, y) = \sum_{i=1}^{N} f_i(x)f_i(y), \tag{1.4.18}$$

for an appropriate collection of functions $(f_i)_{i=1}^N$. In fact, in some sense, any positive definite kernel can be represented in the form (1.4.18), with $N \to \infty$. This is the content of Mercer's theorem. Maybe we will come back to that later.

Remark 1.4.6 (Generalised Sobolev spaces)**.** *In some sense, all RKHSs on $\mathbb{R}^d$ with translation-invariant kernels are Sobolev space, if we allow ourselves to generalise the notion of derivative in (1.4.15), as well as consider infinitely many terms in the sum. This is the content of [7].*

Chapter 2

Kernel Ridge Regression

In this chapter (like in Section 1.1), we assume that data $(x_i, y_i)_{i=1}^N$ is given, with $x_i \in X$ (for some set X, but typically $X = \mathbb{R}^d$) and $y_i \in \mathbb{R}$. We will use the notation

$$\mathcal{D} = \{x_1, \dots, x_N\} \tag{2.0.1}$$

for the set of data points (more precisely, for their x-coordinates).

Our goal is to find a function $f : X \to \mathbb{R}$ thats 'learns' the relationship between x and y,

$$f(x_i) \approx y_i, \qquad\qquad i = 1, \dots, N, \tag{2.0.2}$$

on the *training data*, but also, we would like f to generalise in an appropriate way to unseen data (see Section 1.1). Recall the challenge of balancing over- and underfitting from Section 1.1. The following problem formulation is inspired by this.

Problem 1 (Kernel ridge regression)**.** Fix $\lambda > 0$ and a positive definite kernel $k : X \times X \to \mathbb{R}$ with associated RKHS $\mathcal{H}_k$, and solve

$$\min_{f \in \mathcal{H}_k} \left(\underbrace{\frac{1}{N} \sum_{i=1}^{N} (f(x_i) - y_i)^2}_{\text{data term}} + \underbrace{\lambda \|f\|_{\mathcal{H}_k}^2}_{\text{regulariser}} \right) \tag{2.0.3}$$

Remark 2.0.1. *The data term encodes (2.0.2), while the parameter λ is related to* regularisation. *More precisely, it aims at avoiding overfitting by controlling the complexity of f, as measured by $\|\cdot\|_{\mathcal{H}_k}^2$. The regime $\lambda \to 0$ corresponds to overfitting, and $\lambda \to \infty$ to underfitting. Judicious choice of λ is important, but somewhat beyond the scope of these lectures.*

Notice that for $\lambda > 0$, the objective in (2.0.3) is strictly convex in f, and therefore we expect there to be a unique minimiser f^*. However, the optimisation problem in (2.0.3) is posed over the potentially infinite-dimensional space of functions $\mathcal{H}_k$, and so it might seem that it is computationally difficult to obtain a solution. The following key result establishes that the solution is located in an a priori[1] known finite-dimensional subspace.

Theorem 2.0.1 (Representer theorem)**.** *Let $l : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ (in the following referred to as loss function), $\lambda > 0$, and*

$$f^* \in \operatorname*{arg\,min}_{f \in \mathcal{H}_k} \left(\frac{1}{N} \sum_{i=1}^{N} l(f(x_i), y_i) + \lambda \|f\|_{\mathcal{H}_k}^2 \right). \tag{2.0.4}$$

[1] As we will see below, this subspace only depends on $(x_i)_{i=1}^N$, and not on the 'observations' $(y_i)_{i=1}^N$

Then f^* is of the form

$$f^* = \sum_{i=1}^{N} \alpha_i k(x_i, \cdot), \tag{2.0.5}$$

for appropriate $\alpha_1, \dots, \alpha_N \in \mathbb{R}$.

Remark 2.0.2. *The formulation in (2.0.4) is as slight generalisation; setting $l(z, y) = (z - y)^2$ recovers (2.0.3).*

Remark 2.0.3 (Geometrical interpretation)**.** *Let us define the finite-dimensional subspace*[2]

$$U_{\mathcal{D}} = \operatorname{span}\{k(x, \cdot), \quad x \in \mathcal{D}\} \tag{2.0.6a}$$

$$= \left\{ \sum_{i=1}^{N} \alpha_i k(x_i, \cdot), \quad \alpha_1, \dots, \alpha_N \in \mathbb{R}, \quad x_i \in \mathcal{D} \right\} \subseteq \mathcal{H}_k, \tag{2.0.6b}$$

depending only on $(x_i)_{i=1}^N$, but not on $(y_i)_{i=1}^N$. The representer theorem equivalently states that the solution f^* to (2.0.3) necessarily belongs to $U_{\mathcal{D}}$. In other words, the problem (2.0.3) does not change if we replace $\mathcal{H}_k$ by $U_{\mathcal{D}}$, dramatically reducing the dimensionality of the optimisation problem.

Proof of Theorem 2.0.1. Recall the data subspace $U_{\mathcal{D}}$ defined in (2.0.6), and consider its orthogonal complement

$$U_{\mathcal{D}}^{\perp} := \{f \in \mathcal{H}_k : \quad \langle f, g \rangle_{\mathcal{H}_k} = 0, \quad \text{for all } g \in U_{\mathcal{D}}\}. \tag{2.0.7}$$

Note that $U_{\mathcal{D}}$ is finite-dimensional, hence closed[3], and so we have the decomposition

$$\mathcal{H}_k = U_{\mathcal{D}} \oplus U_{\mathcal{D}}^{\perp}. \tag{2.0.8}$$

This means that any $f \in \mathcal{H}_k$ can written in the form

$$f = g + h, \tag{2.0.9}$$

with $g \in U_{\mathcal{D}}$ and $U_{\mathcal{D}}^{\perp}$ uniquely determined. The idea now is to show that for a minimiser f^* of (2.0.3), we necessarily have $h = 0$ in the decomposition (2.0.9). For this, we use the fact that $U_{\mathcal{D}}^{\perp}$ can be written as follows,

$$U_{\mathcal{D}}^{\perp} = \{f \in \mathcal{H}_k : \quad f(x) = 0, \quad \text{for all } x \in \mathcal{D}\}, \tag{2.0.10}$$

see Lemma 5 below (here we proceed assuming that this is true). We use the decomposition (2.0.9) and write the objective in (2.0.4) in terms of g and h,

$$\frac{1}{N} \sum_{i=1}^{N} l(g(x_i) + \underbrace{h(x_i)}_{=0}, y_i) + \lambda \|g + h\|_{\mathcal{H}_k}^2 = \frac{1}{N} \sum_{i=1}^{N} l(g(x_i), y_i) + \lambda \left(\|g\|_{\mathcal{H}_k}^2 + \|h\|_{\mathcal{H}_k}^2 \right), \tag{2.0.11}$$

using the 'Pythogarean theorem'

$$\|g + h\|_{\mathcal{H}_k^2}^2 = \|g\|_{\mathcal{H}_k}^2 + \underbrace{2\langle g, h \rangle_{\mathcal{H}_k}}_{=0} + \|h\|_{\mathcal{H}_k}^2 = 0, \tag{2.0.12}$$

which holds due to the orthogonality of the decomposition (2.0.8). Now observe that in the reformulation (2.0.11), the components g and h can be varied independently, and that (2.0.9) is minimised for $h = 0$. This proves the claim. □

We proceed by giving the proof for the representation (2.0.10):

[2] Convince yourself that $U_{\mathcal{D}}$ is closed under linear combinations!

[3] Closedness is crucial for the decomposition (2.0.8). To see why, think about dense subspaces.

Lemma 5. *The orthogonal complement of $U_{\mathcal{D}}$ as defined in (2.0.6) can be written in the form (2.0.10).*

Proof. If $f \in U_{\mathcal{D}}^{\perp}$, then in particular

$$\langle f, k(x_i, \cdot)\rangle_{\mathcal{H}_k} = 0, \qquad \text{for all } i = 1, \dots, N. \tag{2.0.13}$$

Using the reproducing property (1.3.3), it is indeed the case that $f(x_i) = 0$, for all $i = 1, \dots, N$. Conversely, if $f(x_i) = 0$, for all $i = 1, \dots, N$, then (2.0.13) holds. We then have $\langle f, g\rangle_{\mathcal{H}_k} = 0$ for all $g \in U_{\mathcal{D}}$ by extending this relation linearly. □

Remark 2.0.4. *It is clear from the proof that the representer theorem crucially relies on the Hilbert space structure of $\mathcal{H}_k$ (orthogonality, etc...).*

Solving the kernel ridge regression problem (2.0.3) explicitly. We can now use the representer Theorem 2.0.1 to solve (2.0.3) explicitly, up to the inversion of the Gram matrix (1.2.3). According to the representer theorem, we are permitted to write $f = \sum_{j=1}^{N} \alpha_j k(x_j, \cdot)$, the objective now being to determine the coefficients $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_N) \in \mathbb{R}^N$.

We calculate

$$\Psi(\boldsymbol{\alpha}) := \frac{1}{N}\sum_{i=1}^{N}\left(\underbrace{\sum_{j=1}^{N}\alpha_j k(x_i, x_j)}_{=f(x_i)} - y_i\right)^2 + \lambda \|\sum_{j=1}^{N}\alpha_j k(x_j, \cdot)\|_{\mathcal{H}_k}^2 \tag{2.0.14a}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=1}^{N}\alpha_j k(x_i, x_j) - y_i\right)^2 + \lambda \sum_{i,j=1}^{N}\alpha_i\alpha_j k(x_i, x_j), \tag{2.0.14b}$$

as well as the derivatives

$$\frac{\partial \Psi(\boldsymbol{\alpha})}{\partial \alpha_l} = \dots. \tag{2.0.14c}$$

Setting $\frac{\partial \Psi(\boldsymbol{\alpha})}{\partial \alpha_l} = 0$ gives the following result.

Theorem 2.0.2. *The solution f^* to (2.0.3) takes the form*

$$f^* = \sum_{j=1}^{N}\alpha_j k(x_j, \cdot), \tag{2.0.15}$$

where $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_N)$ solves the linear system

$$(K + \lambda I_{N\times N})\boldsymbol{\alpha} = \boldsymbol{y}. \tag{2.0.16}$$

Here, $I_{N\times N} \in \mathbb{R}^{N\times N}$ refers to the identity matrix, $K \in \mathbb{R}^{N\times N}$ is given by $K_{ij} = k(x_i, x_j)$ and $\boldsymbol{y} = (y_1, \dots, y_n)\top$.

Remark 2.0.5. *Since k is positive definite and $\lambda > 0$, the matrix $K + \lambda I_{N\times N}$ is strictly positive definite, and hence (2.0.16) admits a unique solution.*

The limit as $\lambda \to 0$. If the Gram matrix K is invertible, then we can (at first, formally) take $\lambda = 0$ in (2.0.16). The function f^* from (2.0.15) then solves the following problem (sometimes called 'ridgeless regression', notice that overfitting is still somewhat prevented by looking for a minimum-norm solution):

$$f^* \in \operatorname*{arg\,min}_{f\in\mathcal{H}_k}\left\{\|f\|_{\mathcal{H}_k} : \quad f(x_i) = y_i, \quad i = 1, \dots, N\right\}. \tag{2.0.17}$$

Furthermore, we have $f_\lambda^* \to f_0^*$ in $\mathcal{H}_k$, where we have used the notation f_λ^* for the solution to (2.0.3) as a function of λ.

2.1 Extensions

The representer theorem can be generalised in a number of ways:

1. The term $\frac{1}{N}\sum_{i=1}^{N} l(f(x_i), y_i)$ may be replaced by a general cost $c((x_1, y_1, f(x_1), \dots (x_N, y_N, f(x_N))$, even attaining the value ∞. This allows us to encode hard constraints.
2. The regularisation term $\|v\|_{\mathcal{H}_k^2}$ may be replaced by $g(\|v\|_{\mathcal{H}_k^2})$, with $g : \mathbb{R}_+ \to \mathbb{R}_+$ strictly monotonically increasing.

Here, we discuss another quite far-reaching generalisation. Note that we can write

$$\frac{1}{N}\sum_{i=1}^{N} l(f(x_i), y_i) = \frac{1}{N}\sum_{i=1}^{N} l(\delta_{x_i}(f), y_i), \tag{2.1.1}$$

using the evaluation functionals (1.4.8). Abstractly, we can view

$$\{u_1', \dots, u_N'\} := \{\delta_{x_1}, \dots, \delta_{x_N}\} \subset \mathcal{H}_k'$$

as a subset of the (continuous) dual of $\mathcal{H}_k$. By the Riesz representation theorem, there exist $u_1, \dots, u_N \in \mathcal{H}_k$ such that

$$\langle u_i, v\rangle_{\mathcal{H}_k} = u_i'(v), \qquad \text{for all } v \in \mathcal{H}_k, \tag{2.1.2}$$

and in fact we know from the reproducing property (1.3.3) that $u_i = k(x_i, \cdot)$. The representer theorem states that $f^* \in \operatorname{span}(u_1, \dots, u_N)$. In fact, this is true in general (see [20]):

Theorem 2.1.1. *Let $(H, \langle\cdot,\cdot\rangle_H)$ be a Hilbert space over $\mathbb{R}$ and denote its continuous dual by $(H', \langle\cdot,\cdot\rangle_{H'})$. Let $U' = \{u_1', \dots, u_N'\} \subset H'$ be a collection of continuous linear functionals on H. Denote the set of associated Riesz representers by $U = \{u_1, \dots, u_N\} \subset H$, that is, we have that*

$$u_j'(v) = \langle u_j, v\rangle_H, \tag{2.1.3}$$

for all $v \in H$ and $j = 1, \dots, N$. Furthermore, let $\{y_1, \dots, y_N\} \subset \mathbb{R}$ be a collection of real numbers, $\lambda > 0$ a regularisation parameter, and consider the regression problem

$$f^* \in \operatorname*{arg\,min}_{v\in H} \left(\frac{1}{N}\sum_{j=1}^{N} \left(u_j'(f) - y_j\right)^2 + \lambda\|f\|_H^2 \right). \tag{2.1.4}$$

Then, (2.1.4) admits a unique solution f^*. Moreover, f^* belongs to the linear span of U, that is,

$$f^* = \sum_{i=1}^{N} \alpha_i u_i, \tag{2.1.5}$$

for appropriate coefficients $\alpha_i \in \mathbb{R}$. The coefficient vector $(\alpha_i)_{i=1}^N = \alpha \in \mathbb{R}^N$ can be obtained as the unique solution to the linear system

$$(\boldsymbol{\xi} + \lambda I_{N\times N})\alpha = \boldsymbol{y}, \tag{2.1.6}$$

where $\boldsymbol{y} = (y_1, \dots, y_N)^\top \in \mathbb{R}^N$, and the matrix $\boldsymbol{\xi} \in \mathbb{R}^{N\times N}$ is given by $\boldsymbol{\xi}_{ij} = \langle u_i, u_j\rangle_H$.

Proof. The proof of Theorem 2.0.1 can be copied, with appropriate (more or less obvious) modifications. □

Derivative reproducing property. How can the generalisation provided by Theorem **??** be useful? Imagine our aim is to solve a linear PDE, for example

$$\Delta f = g \qquad \text{in } \Omega, \tag{2.1.7a}$$
$$f = 0 \qquad \text{on } \partial\Omega, \tag{2.1.7b}$$

on a compact domain Ω. We could do the following: Choose $\{x_1, \dots, x_N\} \subset \Omega$, and define

$$u_i'(f) = \Delta f(x_i), \tag{2.1.8}$$

and do something similar for the boundary constraint (2.1.7b). If we can find a positive definite kernel k with associated RKHS $\mathcal{H}_k$ such that the functionals u_i' are continuous, then we can use Theorem 2.0.1 to produce an approximate solution. Indeed, we have the following result:

Proposition 2.1.1. *Let $s \in \mathbb{N}$ and $k : X \times X \to \mathbb{R}$, with $X \subset \mathbb{R}^d$ compact. Assume that $k \in C^{2s}(X \times X)$. For a multiindex $\alpha \in \mathbb{N}^d$ with $|\alpha| \leq s$ the following holds:*

1. *We have $D_x^\alpha k(x, \cdot) \in \mathcal{H}_k$, for all $x \in X$.*
2. *The* derivative reproducing property *holds:*

$$D^\alpha f(x) = \langle D_x^\alpha k(x, \cdot), f \rangle_{\mathcal{H}_k}, \qquad f \in \mathcal{H}_k,\ x \in X. \tag{2.1.9}$$

3. *For all $x \in X$, the mapping $\mathcal{H}_k \ni f \mapsto D_x^\alpha f(x) \in \mathbb{R}$ is continuous.*

Proof. (Sketch). The continuity follows from (2.1.9) via

$$|D^\alpha f(x)| = |\langle D_x^\alpha k(x, \cdot), f \rangle_{\mathcal{H}_k}| \leq \|D_x^\alpha k(x, \cdot)\| \|f\|_{\mathcal{H}_k} \tag{2.1.10}$$

as in the proof of Lemma 4. We only sketch the proof of (2.1.9), for $d = 1$ and $\alpha = 1$. For this, notice that

$$\frac{f(x+h) - f(x)}{h} = \left\langle f, \frac{k(x+h, \cdot) - k(x, \cdot)}{h} \right\rangle_{\mathcal{H}_k}, \tag{2.1.11}$$

for $f \in \mathcal{H}_k$. Now (carefully), take the limit $h \to 0$. □

Chapter 3

Kernel Embeddings

3.1 Motivation

In this chapter we study the *kernel trick*, a machine-learning method that allows to use 'linear' algorithms for 'non-linear' situations. This concept is referred to as *algorithm kernelization*. To explain it we use the following example of a linear algorithm.

Principal Component Analysis (PCA). We assume data $(x_i)_{i=1}^N$ is given, with $x_i \in \mathbb{R}^D$. In this section we will use the notation

$$X = \begin{pmatrix} x_1^T \\ \vdots \\ x_N^T \end{pmatrix} \in \mathbb{R}^{N \times D} \tag{3.1.1}$$

for the set of data points, so that $X_{ij} = x_i^{(j)}$ is the jth component of the ith data point. Our goal is to reduce the dimensionality of the data to $\mathbb{R}^d$, with $d \ll D$, such that *information loss is minimized* for any fixed d. The PCA algorithm can be used either as a data pre-processing step, followed by other algorithms, or in its own right, as a pattern recognition technique in unsupervised learning[1], to identify which part of the information contained in the data is relevant.
For a fixed target dimension $d < D$ onto which we would like to project the data, we want to determine a projection (subspace of dimension d) such that the information loss is minimal. This is equivalent to requiring that the variance of the projected data is maximal. The algorithm is as follows:

Step 1: Without loss of generality, we can assume that the data is centered, so that $\bar{x} = \frac{1}{N}\sum_{i=1}^N x_i = 0$, otherwise we apply the algorithm to the centered data $(\tilde{x}_i)_{i=1}^N = (x_i)_{i=1}^N - \bar{x}$.

Step 2: In order to define the projection, we need to compute the covariance matrix associated with the data:

$$C = \frac{1}{N}\sum_{i=1}^N (x_i - \bar{x})(x_i - \bar{x})^T = \frac{1}{N} X^T X \in \mathbb{R}^{D \times D}. \tag{3.1.2}$$

It is simple to prove that:

Lemma 6. *The covariance matrix C defined in (3.1.2) is symmetric and positive semidefinite.*

Proof.

$$\text{(Symmetry)} \qquad C^T = \left(\frac{1}{N} X^T X\right)^T = \frac{1}{N} X^T \left(X^T\right)^T = C, \tag{3.1.3}$$

[1] Meaning that the data could have some categories associated to it, but we do not know them.

and

$$\text{(Positive semidefiniteness)} \qquad a^T C a = \frac{1}{N} a^T X^T X a = \frac{1}{N} \sum_{i=1}^{N} \left((a^T X^T)_i \right)^2 \qquad \forall a \in \mathbb{R}^D. \quad (3.1.4)$$

□

Step 3: Any symmetric matrix can be diagonalized. We can find an orthonormal basis $\{u_1, \dots, u_D\} \subset \mathbb{R}^D$, $u_i^T u_j = 0$, if $i \neq j$ and $u_i^T u_j = 1$, if $i = j$, such that $\|u_i\| = 1$, $i = 1, \dots, D$. Then

$$C u_i = \lambda_i u_i, \quad \lambda_i > 0, \quad i = 1, \dots, D, \tag{3.1.5}$$

and

$$C = U \Lambda U^T, \tag{3.1.6}$$

where

$$\Lambda = \begin{pmatrix} \lambda_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \lambda_D \end{pmatrix}, \quad U = \begin{pmatrix} u_1 \dots u_D \end{pmatrix}. \tag{3.1.7}$$

We assume that the eigenvalues (and corresponding eigenvectors) are sorted in decreasing order:

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_D \geq 0. \tag{3.1.8}$$

Step 4: Equation (3.1.6) can be interpreted as a change of basis for the covariance matrix C, but we can equivalently write C in terms of scalings and projections

$$C = \sum_{i=1}^{D} \lambda_i u_i u_i^T. \tag{3.1.9}$$

In fact, the mapping $x \mapsto u_i(u_i^T x)$ is a projection of x onto the direction of u_i, for any $x \in \mathbb{R}^D$. This holds more generally for orthogonal projections, so that, for example, $u_1 u_1^T + u_2 u_2^T$ is the orthogonal projection onto the subspace spanned by $\{u_1, u_2\}$. In (3.1.9) the scaling factor λ_i is the variance of the data in direction u_i, and because we assumed the eigenvalues are sorted (3.1.8), then u_1 is the direction with largest variance, u_2 is the direction with second largest variance and so on. It is therefore sensible to chose the following rank-d projection

$$P = \sum_{i=1}^{d} u_i u_i^T, \quad d < D, \tag{3.1.10}$$

as output to the algorithm for projecting the data onto the subspace spanned by $\{u_1, \dots, u_d\}$.

It is possible to prove that:

Lemma 7. *The projection P in (3.1.10) has the following optimality properties*

$$P \in \arg\max_{P \in \mathcal{P}} \mathit{Var}(P\{x_i\}_{i=1}^N), \tag{3.1.11a}$$

$$P \in \arg\min_{P \in \mathcal{P}} \frac{1}{N} \sum_{i=1}^{N} |x_i - P x_i|^2, \tag{3.1.11b}$$

where $\mathcal{P}$ is the set of projections onto d-dimensional subspaces, and in (3.1.11a) $P\{x_i\}_{i=1}^N$ is the projected dataset, and the maximum should be interpreted in terms of positive definite matrices (Loewner order).[2]

[2] That is $x^T(P - Q)x > 0$ for any $Q \in \mathbb{R}^{d \times d}$ positive definite matrix and $x \in \mathbb{R}^d$.

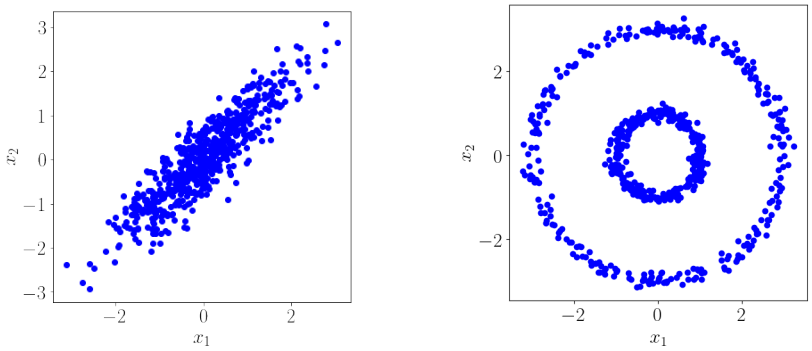


Figure 3.1.1: Left: linear data; The projection onto the linear sub-space identified by the PCA algorithm captures the pattern (direction of highest variability) in the data. Right: nonlinear data; any projection on a linear subspace fails to efficiently identify patterns in the data.

Remark 3.1.1. *(New coordinates). The dimensionality reduced dataset is*

$$Y = \begin{pmatrix} y_1^T \\ \vdots \\ y_N^T \end{pmatrix} \in \mathbb{R}^{N \times d}, \tag{3.1.12}$$

where

$$y_i^s = x_i^T u_s, \qquad i = 1, \dots, N, \quad s = 1, \dots, d. \tag{3.1.13}$$

Remark 3.1.2. *(Computational bottleneck). The most expensive operation in the PCA algorithm is the eigendecomposition of the matrix C in Step 3. This can be challenging if D, the original data dimension, is large.*

Remark 3.1.3. *(When PCA does not work).Given that the PCA is based on the eigendecomposition of the sample covariance matrix C, the algorithm works well when this captures the data distribution, i.e. when the data directions are linearly correlated. When, on the other hand, the data has a nonlinear structure, PCA is not able to identify it. See figure 3.1.1.*

In order to find a suitable pattern in the data in situations like that in the left-hand side of Figure 3.1.1, a possible solution is to embed the data in a higher (infinite) dimensional *feature space*. The idea is that a richer representation of the data enables to efficiently use linear algorithms.

3.2 Feature maps

Before introducing the specific feature space that we will operate with, we define the general concept of features, beyond kernel methods. Real-data (be this text, pictures, molecules etc.) can be thought as belonging to a generic set X. Computational methods based on data manipulation require a numerical representation of the data, called *feature map*. For example we can have a vector valued feauture map

$$\phi : X \to \mathbb{R}^D, \tag{3.2.1}$$

were elements in $\mathbb{R}^D$ corresponding to $\phi(x)$, $x \in X$ are called *features*. For example we can think of $X = \{\text{people in a certain group}\}$ and $\phi(x) = (\text{height}) \in \mathbb{R}$, or $\phi(x) = (\text{height, age, weight})^T \in \mathbb{R}^3$. We would like to be able to do numerical computations on the set of features. For example we would like to be able to compute the scalar product, for $x, y \in X$:

$$\phi(x) \cdot \phi(y) = \langle \phi(x), \phi(y) \rangle_{\mathbb{R}^D} = \sum_{i=1}^{D} \phi^{(i)}(x) \phi^{(i)}(y), \tag{3.2.2}$$

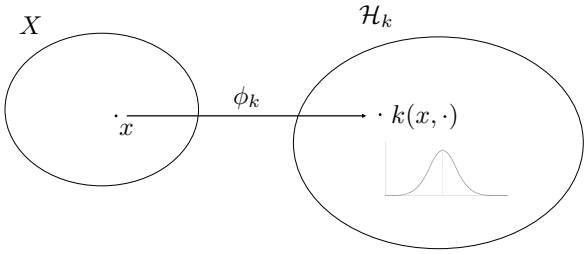


Figure 3.2.1: The canonical feature map contains information about potentially infinitely many numbers (via the second argument of the kernel). It thus provide a very rich feature representation of the sample $x \in X$ (in fact this is an infinite dimensional embedding of x).

where $\phi^{(i)}$ denotes the i-th component of the feature.

From features to kernels. Features defined in the way above are related to kernels, in that they naturally induce a kernel on the underlying set X. In fact:

Proposition 3.2.1. *Let $\phi : X \to H$ be any map, where $(H, \langle \cdot, \cdot \rangle_H)$ is a Hilbert space. Then*

$$k(x, y) := \langle \phi(x), \phi(y) \rangle_H \tag{3.2.3}$$

defines a positive definite kernel on X.

Proof. Both symmetry and positivity are inherited from the scalar product $\langle \cdot, \cdot \rangle_H$. □

For example, given vector-valued features $\phi : X \to \mathbb{R}^D$ we could define a kernel using the scalar product $\langle \phi(x), \phi(y) \rangle_{\mathbb{R}^D}$ defined in (3.2.2) or indeed any other scalar products on $\mathbb{R}^D$ induced by a positive definite matrix A:

$$\langle \phi(x), \phi(y) \rangle_A := \sum_{i,j=1}^{D} \phi^i(x) A_{i,j} \phi^j(y). \tag{3.2.4}$$

From kernels to features. Kernels also naturally induce features valued in RKHSs. Specifically, let $k : X \times X \to \mathbb{R}$ be a positive definite kernel and recall the reproducing property that holds for functions in the associated RKHS:

$$\langle f, k(x, \cdot) \rangle_{\mathcal{H}_k} = f(x), \qquad \text{for all } f \in \mathcal{H}_k,\ x \in X. \tag{3.2.5}$$

If we take $f = k(y, \cdot)$ for some $y \in X$, then the above becomes (by symmetry)

$$\Big\langle \underbrace{k(x, \cdot)}_{\phi_k(x)}, \underbrace{k(y, \cdot)}_{\phi_k(y)} \Big\rangle_{\mathcal{H}_k} = k(x, y), \qquad \forall x, y \in X. \tag{3.2.6}$$

Equation (3.2.6) leads to the definition of the so called *canonical feature map*:

$$\phi_k : X \to \mathcal{H}_k, \tag{3.2.7}$$

$$x \mapsto k(x, \cdot). \tag{3.2.8}$$

Thus every positive definite kernel induces a feature map, which is a function in a function space (the RKHS corresponding to the kernel). This operation is called *kernel embedding*. See Figure 3.2.1.

Remark 3.2.1. *(Non unique mapping features → kernels). From equation (3.2.6), given a kernel k, we obtain the canonical (function-valued) feature map. However, from proposition 3.2.1 there might be other (possibly function-valued) features that lead to the same kernel k, so that we can write*

$$\langle \phi(x), \phi(y) \rangle_{\mathcal{H}_k} = \langle k(x, \cdot), k(y, \cdot) \rangle_{\mathcal{H}_k} = k(x, y), \qquad \forall x, y \in X. \tag{3.2.9}$$

It is of course possible (in fact more common) that two different feature maps induce different kernels.

3.2.1 Review of connections and equivalences

In chapter 1 we provided a first definition of positive definite kernel (see definition 1). An alternative definition, in light of the idea of feature maps introduced in this chapter, is the following:

Definition 5 (Positive definite kernels - alternative)**.** Let $X \neq \emptyset$ be a set. A function $k : X \times X \to \mathbb{R}$ is called a *positive definite kernel* if there exist a Hilbert space $(H, \langle \cdot, \cdot \rangle_H)$ and a map $\phi : X \to H$ (feature map) such that

$$k(x, y) := \langle \phi(x), \phi(y) \rangle_H, \qquad \forall x, y \in X. \tag{3.2.10}$$

In fact:

Theorem 3.2.1. *Definitions 1 and 5 are equivalent.*

We do not provide a detailed proof, but this would follow the steps performed in section 3.2 to define kernels from feature maps and vice-versa.

Remark 3.2.2. *A few remarks on alternative definitions of positive definite kernels:*

1. *Definition 1 is related to use of kernels as building blocks for RKHSs, as presented in chapter 1; on the other hand, definition 5 is related to feature maps and the kernel trick. The equivalence is reflected in the equivalence we draw between the solution to KRR and the kernelized version of regularized LS.*

2. *Possible candidates Hilbert spaces in definition 5 are $(\mathbb{R}^D, \langle \cdot, \cdot \rangle_A)$ with scalar product induced by a positive definite matrix A, or the Sobolev space 1.4.15, or yet $L^2(\mathbb{R})$.*[3]

3. *As mentioned in remark 1.2.1, we can often think of $k(x, y)$ as encoding a measure of similarity between x, $y \in X$ (think for example of the squared exponential kernel). In such cases, equation (3.2.10) in definition 5 means that the kernel $k(x, y)$ encodes also a measure of similarity in the Hilbert space H: the kernel takes large values when $\phi(x)$ and $\phi(y)$ lie in similar directions in H, and is vanishing when $\phi(x)$ and $\phi(y)$ are orthogonal.*

4. *For a given positive definite kernel $k : X \times X \to \mathbb{R}$ the unique RKHS associated to it can serve as a feature space. In fact, applying the reproducing property to $\phi(x) = k(x, \cdot)$, we have $\langle k(x, \cdot), k(y, \cdot) \rangle_{\mathcal{H}_k} = k(x, y)$. However, as stated in remark 3.2.1, to any kernel we can associate many feature spaces (one of which is the unique RKHS).*

3.3 Algorithm kernelization

The feature space $\mathcal{H}_k$ is infinite-dimensional. Algorithms that use features projections, such as PCA[4], have the computational advantage of being able to work with finite dimensional feature spaces. It is in fact enough to consider $\mathcal{U} \in \mathcal{H}_k$ to be a linear finite dimensional subspace and $P_{\mathcal{U}}$ be the corresponding projection. Then $(P_{\mathcal{U}} \circ \phi)(x) = P_{\mathcal{U}}(\phi(x))$ gives a finite dimensional feature map (taking values in $\mathcal{U}$).

However, in the case of general algorithms not involving projections on finite dimensional subspaces, it is often impossible, or at least inconvenient, to work directly with the canonical (or other function-valued) features ϕ_k and the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_k}$, which is difficult to compute explicitly. Luckily, equation 3.2.9 tells us that the scalar products of features, which might have an important role in course of an algorithm, coincides with a kernel evaluation, which is easy to compute. This leads to the following fundamental concept, used in many many machine learning and statistical applications:

[3] Hence it is not necessary for the Hilbert space in the image of the feature map to be an RKHS, in order to enable the definition of a positive definite kernel.

[4] In the motivating example in section 3.1, PCA was applied to a D-dimensional vector. In general, this will be the set of features corresponding to a sample in the data, which can here we consider to be the kernel − infinite dimensional embedding.

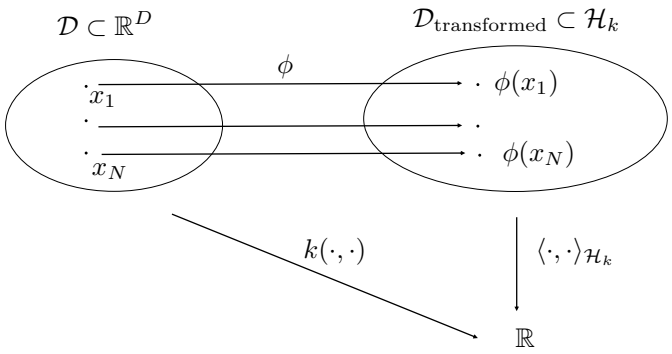


Figure 3.3.1: The kernel trick consist of working implicitly with data embeddings, by replacing scalar products in RKHSs directly with kernel evaluations (if an algorithm only performs this operation on the data).

Kernel trick. Algorithms that take D-dimensional data $\mathcal{D} = \{x_1, \dots, x_N\} \subset \mathbb{R}^D$ and only rely on standard inner products

$$\langle x_i, x_j \rangle_{\mathbb{R}^D} = \sum_{l=1}^{D} x_i^{(l)} x_j^{(l)} \tag{3.3.1}$$

can be kernelized. The trick is simply:

Choose a kernel $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ and replace every instance of $\langle x_i, x_j \rangle_{\mathbb{R}^D}$ by $k(x_i, x_j)$.

The kernelized algorithm can thus be thought of as working on the transformed data $\mathcal{D}_{\text{transformed}} = \{\phi(x_1), \dots, \phi(x_N)\} \subset \mathcal{H}_k$ corresponding to the kernel embeddings of the original data $\mathcal{D}$, because of (3.2.9). Notice however that the kernelized algorithm does not explicitly use $\mathcal{D}_{\text{transformed}}$. Figure 3.3.1 summarizes the kernel trick.

Guideline on kernel choice. When kernelizing an algorithm, kernel evaluations are replacing scalar products in $\mathbb{R}^D$. This provides intuition about how to choose the kernel: k needs to take *large values* when two data points x_i and x_j, $i, j = 1, \dots, N$ are *similar*, and low values when they are dissimilar.

The most important consequence of using the kernel trick is to transform a linear algorithm into a non-linear one, so that it can now work efficiently with non-linear data. Notice that not any algorithm can be expressed in terms of scalar products of the data only, and recognizing weather or not this is the case is key to applying the kernel trick. We give some examples in the following.

3.3.1 Kernel PCA

We introduce the *kernel matrix*

$$K = XX^T \in \mathbb{R}^{N \times N}, \tag{3.3.2}$$

and we relate it to the unnormalized covarince matrix

$$\tilde{C} = X^T X = NC \in \mathbb{R}^{D \times D}. \tag{3.3.3}$$

The Matrix $\tilde{C}$ is connected to the shape of the D-dimensional distribution of the data in that

$$\tilde{C}_{ij} = \sum_{n=1}^{N} x_n^{(i)} x_n^{(j)}, \qquad i, j = 1, \dots, D, \tag{3.3.4}$$

quantifies the correlation between the directions i and j in $\mathbb{R}^D$. On the other hand, the matrix K describes the alignment between data points, in that

$$K_{ij} = x_i^T x_j, \qquad i, j = 1, \dots, N, \tag{3.3.5}$$

is the scalar product between the i-th and the j-th samples (out of N).

Relationship between the eigendecomposition of K and $\tilde{C}$. The spectra of K and $\tilde{C}$ coincide, up to possibly a sequence of zeros, so that the following holds:

Lemma 8. *If $\lambda \neq 0$ is an eigenvalue of K, then it is also an eigenvalue of $\tilde{C}$ and vice versa.*

On the other hand, the respective eigenvectors can be transformed into each other:

Lemma 9. *(i) If $a \in \mathbb{R}^N$ is an eigenvector of K with associated eigenvalue $\lambda \neq 0$, then $v = X^T a \in \mathbb{R}^D$ is the corresponding eigenvector of $\tilde{C}$. Furthermore $\|v\|^2 = \lambda \|a\|^2$.*
(ii) If $v \in \mathbb{R}^D$ is an eigenvector of $\tilde{C}$ with associated eigenvalue $\lambda \neq 0$, then $a = Xv \in \mathbb{R}^N$ is the corresponding eigenvector of K. Furthermore $\|a\|^2 = \lambda \|v\|^2$.

Proof. We prove (i), and (ii) can be proved in a similar way. Assume that, for $\lambda \neq 0$ and $a \neq 0$

$$Ka = \lambda a. \tag{3.3.6}$$

This is equivalent to

$$XX^T a = \lambda a \tag{3.3.7}$$

and left-multiplying each side with X^T we get

$$X^T X X^T a = \lambda X^T a \tag{3.3.8}$$

which can be re-written

$$Cv = \lambda v, \quad v = X^T a. \tag{3.3.9}$$

We know that $v = X^T a \neq 0$ because equation (3.3.7) holds for $\lambda \neq 0$. Furthemore,

$$\|v\|^2 = v^T v = a^T X X^T a = a^T K a \tag{3.3.10}$$
$$= a^T \lambda a \tag{3.3.11}$$
$$= \lambda \|a\|^2 \tag{3.3.12}$$

□

Computational advantage. As a consequence, to address remark 3.1.2, if $D \gg N$ we can compute the eigendecomposition of K and accordingly transform the eigenvectors and eigenvalues to obtain those of $\tilde{C}$. In particular, if we determine the sorted eigenvalues-eigenvectors of K

$$Ka_i = \lambda a_i, \qquad i = 1, \dots, N, \tag{3.3.13}$$
$$\lambda_1 \geq \lambda_2 \geq \lambda_N \geq 0 \tag{3.3.14}$$
$$a_i^T a_j = \begin{cases} 0, \text{ if } i \neq j \\ 1, \text{ if } i = j \end{cases} \tag{3.3.15}$$

then we can compute the eigenvectors of $\tilde{C}$ as

$$u_i = \frac{1}{\sqrt{\lambda_i}} X^T a_i, \quad i = 1, \dots, D, \tag{3.3.16}$$

where the scaling by $\sqrt{\lambda_i}$ is applied to ensure that $\|u_i\|^2 = 1$, as required in the PCA algorithm for the eigendecomposition of C.[5]

PCA kernelization. The relationship between the matrices K and $\tilde{C}$ is also essential to recognize that the PCA algorithm can be kernelized and to tackle remark 3.1.3. In fact, the elements of K can be written in terms of scalar products (3.3.5), whilst this is not possible for the elements of $\tilde{C}$ (3.3.4). Notice that the data X^T also enters the formula for computing the eigenvectors of $\tilde{C}$ (3.3.16), and this dependence alone cannot be expressed in terms of scalar products. However, the eigenvectors u_i are consequently used in the algorithm for projections, see equation (3.1.13), which in turn depend only on scalar products of the data. In fact, for any $i = 1, \dots, N$, and $s = 1, \dots, d \leq D$, the new projected coordinates are

$$y_i^s = x_i^T u_s = \frac{1}{\sqrt{\lambda_s}} x_i^T \left(X^T a_s \right) \tag{3.3.17}$$

$$= \frac{1}{\sqrt{\lambda_s}} \left(X x_i \right)^T a_s \tag{3.3.18}$$

$$= \frac{1}{\sqrt{\lambda_s}} \begin{pmatrix} x_1^T x_i \\ \vdots \\ x_N^T x_i \end{pmatrix}^T a_s \tag{3.3.19}$$

$$= \frac{1}{\sqrt{\lambda_s}} \sum_{l=1}^{N} (x_l^T x_i) a_s^{(l)}, \tag{3.3.20}$$

that clearly depend only on scalar products of the data points. Dimensionality reduction is achieved by choosing $d \ll D$.

Thus the PCA algorithm can be kernelized by replacing every instance of scalar products with a chosen positive definite kernel. In practice, this is equivalent to replacing the kernel matrix $K = XX^T$ by the *Gram matrix*

$$\begin{pmatrix} k(x_1, x_1) & \dots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \dots & k(x_n, x_n) \end{pmatrix}. \tag{3.3.21}$$

In kernel PCA, the new projected coordinates are for $i = 1, \dots, N$, and $s = 1, \dots, n \leq N$

$$y_i^s = \frac{1}{\sqrt{\lambda_s}} \sum_{l=1}^{N} k(x_i, x_l) a_s^{(l)}, \tag{3.3.22}$$

and dimensionality reduction is now achieved if $n \ll N$. Recall that, by replacing the scalar products with kernel evaluations it is as if we worked with transformed data made by infinite dimensional features. And whilst the matrix K has at most D non-zero eigenvalues, the Gram matrix (3.3.21) can have a larger non vanising spectrum.

Example 3.3.1. We continue the example on the right-hand side in Figure 3.1.1, for which the regular PCA algorithm was not able to efficiently identify directions of data variation, and hence to efficently reduce the dimension of the data. Following our previous recommendation on kernel choice, we select a Gaussian kernel (1.2.4) with scale $\sigma = 2$ and perform kernel PCA.[6] The left-hand side of figure 3.3.2 shows the first three new data coordinates y_i^s, $i = 1, \dots, N$ and $s = 1, \dots, 3$, computed as in equation (3.3.22). These coordinates are visibly able to identify the principal directions of variation in the

[5] Notice that Lemma 9 relates the eigendecomposition of K and $\tilde{C}$, but it is easy to prove that C has the same eigenvectors as $\tilde{C}$. Furthermore, if $\lambda^{\tilde{C}}$ is an eigenvalue of $\tilde{C}$, then $\lambda^C = \frac{1}{N}\lambda^{\tilde{C}}$ is the corresponding eigenvalue of C. However, only the eigenvectors play a role in determining the projection matrix used in PCA.

[6] Other choices of σ (and in fact kernel) are possible, as long as the corresponding kernel *sees* the data on the two circles as orthogonal, when embedded in the feature space. That is the kernel centered at a point on one of the two circles is nearly zero when evaluated on the other circle.

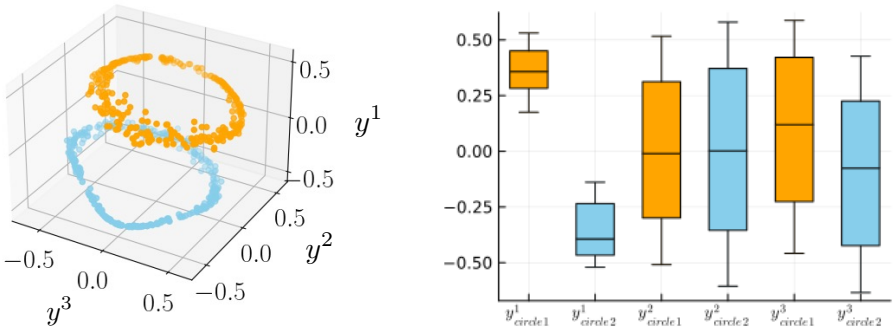


Figure 3.3.2: Kernel PCA for the example on the right-hand side in Figure 3.1.1. Left: three-dimensional visualization of the projected coordinates, colour-coded based on belonging to inner/outer circle in the original data; right: boxplot of each projected coordinate, grouped by inner/outer circle association.

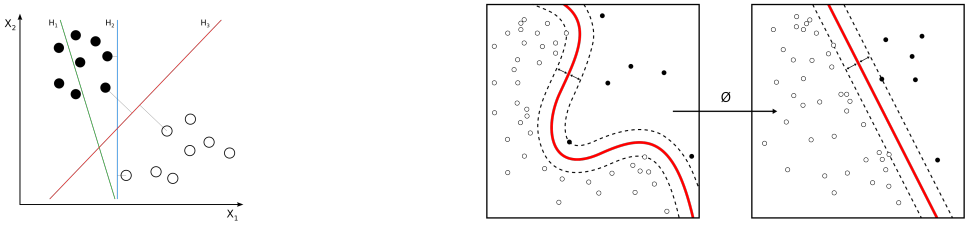


Figure 3.3.3: Representation of the SVM algorithm. Left: linear SVM finds the maximum-margin hyperplane; Right: kernel SVM in the original coordinates can be thought of as linear SVM in the RKHS feature space. Image credit: Wikipedia.

data. In fact, in the new coordinates we are able to *separate* the two circles around which the data lie. The boxplot on the righ-hand side shows how the main source of variability is captured by the first coordinate.

3.3.2 Other kernelizable algorithms

Here we give examples of other machine learning and statistics algorithms that can be kernelized, without entering the details.

Support vector machine (SVM). This is historically the first algorithms to be kernelized, see for example [19]. SVM is a (without loss of generality) binary classification algorithm for supervised learning. Data is given in the form $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, with $x_i \in \mathbb{R}^D$ and $y_i \in \{-1, 1\}$. The linear SVM algorithm finds an optimal separating hyperplane, so as to maximise the width of the gap between the two categories. This is showed on the left-hand side of Figure 3.3.3. New data points can then be classified according to the side on which the lie, in relation to the hyperplane.
The kernel trick helps to separate non-linearly separable data, by lifting the data through a feature map to a high-dimensional space, where the data can now be linearly separated. This is illustrated on the right-hand side of figure 3.3.3, where the non-linear separating curve represents the projection in the original space of the hyperplane in the infinite dimensional feature space (RKHS).

Regularized least squares (LS) regression. We assume to have data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, with covariates $x_i \in \mathbb{R}^D$ and regressed variables $y_i \in \mathbb{R}$. The aim is to find an optimal linear fit to the data, of the form

$$f(x_i) = w^T x_i \approx y_i, \qquad i = 1, \dots, N, \tag{3.3.23}$$

where $w \in \mathbb{R}^D$ is vector of weights of the linear function f. In regularized LS regression w is determined by finding the unique minimum of the quadratic loss function

$$L(w) = \sum_{i=1}^{N} \left(y_i - w^T x_i\right)^2 + \lambda \|w\|^2 \qquad \lambda \geq 0. \tag{3.3.24}$$

where the first term measures the distance of the estimated linear function from the data, whilst the second term constrains the weights assigned coordinates of the covariates. It is possible to prove that

$$w^* \in \arg\min_{w \in \mathbb{R}^D} L(w) = \left(X^T X + \lambda I_{N \times N}\right)^{-1} X^T y, \tag{3.3.25}$$

where the matrixFurthermore, X is defined as in (3.1.1), $y = \{y_1, \ldots, y_N\}$, and it is possible to notice how the regularizing term in the loss function also stabilizes the algorithm for $\lambda \neq 0$.
However, as in our motivating example in section 1.1 and chapter 2, the regressed variables might not be a linear function of the covariates. It is possible to prove that regularized LS regression can be kernelized following similar steps to those used in kernel PCA. Furthermore, The solution to the resulting algorithm coincides with the KRR optimal solution provided in theorem 2.0.2.

Remark 3.3.1. *(Connection between SVM and KRR). Given the connection between regression and classification problems, it is possible to establish a link also between SVM and KRR, but we do not provide details here.*

3.4 Extensions

We conclude this chapter presenting the ideas of RKHS-embeddings of probability distributions. These can be thought of as the generalization of kernel feature maps, and they allow to define meaningful and computable metrics (and discrepancies) between distributions. We follow the presentation in [13] and references therein.

3.4.1 Kernel mean embeddings

Let $X \neq \emptyset$ and $k : X \times X \to \mathbb{R}$ be a positive definite kernel with associated RKHS $\mathcal{H}_k$. Throughout this chapter, we studied how the map

$$\phi_k : X \to \mathcal{H}_k, \tag{3.4.1}$$
$$x \mapsto k(x, \cdot) \tag{3.4.2}$$

can be thought of (i) as a high (infinite)-dimensional *representer* of x, for any $x \in X$, in the sense of canonical feature map; (ii) as the *representer of evaluations of any function in* $\mathcal{H}_k$ at the point x, in the sense that the reproducing property holds:

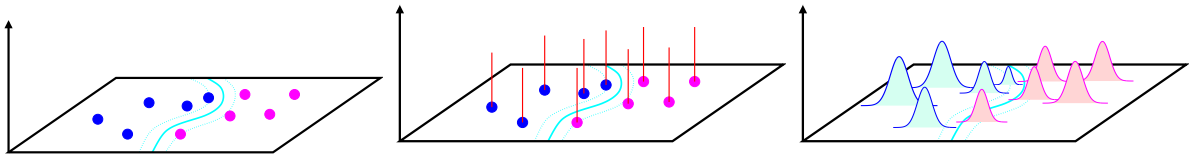
$$f(x) = \langle f, k(x, \cdot) \rangle_{\mathcal{H}_k}, \qquad \text{for all } x \in X,\ f \in \mathcal{H}_k. \tag{3.4.3}$$

These interpretations can be generalized to probability measures[7] $\mathcal{P}$ on a measurable space (X, Σ), where Σ is a σ-algebra of subsets of X. We start with the simple case of *Dirac measure* and we then extend to general probability measures.

Kernel mean embedding of Dirac measures. Let δ_x be a Dirac measure, defined for any $x \in X$ and $A \in \Sigma$ as

$$\delta_x(A) = \begin{cases} 1, \text{ if } x \in A \\ 0, \text{ if } x \notin A, \end{cases} \tag{3.4.4}$$

[7] In most practical applications these will be of interest, but we could also work with signed measures.



(a) $\mathbf{x} \mapsto k(\mathbf{x}, \cdot)$ **(b)** $\delta_{\mathbf{x}} \mapsto \int k(\mathbf{y}, \cdot)\, \mathrm{d}\delta_{\mathbf{x}}(\mathbf{y})$ **(c)** $\mathbb{P} \mapsto \int k(\mathbf{x}, \cdot)\, \mathrm{d}\mathbb{P}(\mathbf{x})$

Figure 3.4.1: From data points to probability measures: a) Illustration of an application of a positive definite kernel as high-dimensional feature map of individual points. b) Measure theoretic view of a high-dimensional feature map. An embedding of data points into a high-dimensional feature space can be seen as embedding of a Dirac measure assigning mass 1 to each data point. c) Extension of b) to embeddings of general probability distributions. Image credit: [13].

Any measurable function f on X can be integrated with respect to δ_x and it holds that

$$\int f(t)\, \mathrm{d}\delta_x(t) = f(x). \tag{3.4.5}$$

When f also belongs to a given RKHS $\mathcal{H}_k$, we can rewrite equation (3.4.5) as:

$$\int f(t)\, \mathrm{d}\delta_x(t) = \int \langle f, k(t, \cdot)\rangle_{\mathcal{H}_k}\, \mathrm{d}\delta_x(t) = \Big\langle f, \underbrace{\int k(t, \cdot)\, \mathrm{d}\delta_x(t)}_{\mu_{\delta_x}} \Big\rangle_{\mathcal{H}_k} = \langle f, \mu_{\delta_x}\rangle_{\mathcal{H}_k}, \tag{3.4.6}$$

where in the first equality we used the reproducing property, and the second equality trivially holds because we integrate functions with respect to the Dirac measure.

Definition 6. The map[8]

$$\mu_{\delta_x} : \mathcal{P} \to \mathcal{H}_k, \tag{3.4.7}$$

$$\delta_x \mapsto \int k(t, \cdot)\, \mathrm{d}\delta_x(t) = k(x, \cdot) \tag{3.4.8}$$

is the *kernel mean embedding* of the Dirac measure δ_x.

Remark 3.4.1. *(Measure theoretic interpretation). By analogy to the canonical feature map and its interpretation in the reproducing property, μ_{δ_x} acts (i) as a representer of the measure δ_x in the RKHS $\mathcal{H}_k$; (ii) as a representer of evaluations of the functional expected value of f with respect δ_x. For the simple Dirac measure the kernel mean embedding coincides with the canonical feature map and equation (3.4.6) coincides with the reproducing property in the RKHS $\mathcal{H}_k$. However, (3.4.6) needs to be interpreted as a measure-theoretic point of view of the reproducing property, in that the map μ_{δ_x} does not act on x, but on δ_x, a probability measure assigning mass 1 to the set $\{x\} \subset \Sigma$. Figure 3.4.1 shows this interpretation.*

Kernel mean embedding of general probability measures. If we consider a probability measure $P \in \mathcal{P}$, its kernel mean embedding is given by the map

$$\mu_P : \mathcal{P} \to \mathcal{H}_k, \tag{3.4.9}$$

$$P \mapsto \int k(x, \cdot)\, \mathrm{d}P(x) \tag{3.4.10}$$

and the following provides conditions under which the embedding μ_P exists and belongs to $\mathcal{H}_k$:

[8] To be intended as Bochner integral.

Lemma 10. *If* $\int \sqrt{k(x,x)}\,\mathrm{d}P(x) < \infty$ *then* $\mu_P \in \mathcal{H}_k$ *and* $\int f(x)\,\mathrm{d}P(x) = \langle f, \mu_P\rangle_{\mathcal{H}_k}$.

Proof. Consider the linear operator $Lf = \int f(x)\,\mathrm{d}P(x)$. Then

$$|Lf| = \left|\int f(x)\,\mathrm{d}P(x)\right| \leq \int |f(x)|\,\mathrm{d}P(x) \tag{3.4.11}$$

$$= \int |\langle f, k(x,\cdot)\rangle_{\mathcal{H}_k}|\,\mathrm{d}P(x) \tag{3.4.12}$$

$$\leq \int \|f\|_{\mathcal{H}_k}\|k(x,\cdot)\|_{\mathcal{H}(k)}\mathrm{d}P(x) \tag{3.4.13}$$

$$= \|f\|_{\mathcal{H}(k)} \int \sqrt{k(x,x)}\mathrm{d}P(x) \tag{3.4.14}$$

where (3.4.11) is Jensen's inequality, (3.4.12) is the reproducing property, (3.4.13) is Cauchy-Schwarz and (3.4.14) is again the reproducing property. This shows that L is a bounded linear operator from $\mathcal{H}_k$ to $\mathbb{R}$. Thus, from the Riesz representation theorem, there exists $h \in \mathcal{H}_k$ such that $Lf = \langle f, h\rangle_{\mathcal{H}_k}$. Taking $f = k(y,\cdot)$ for some $y \in X$ and using the reproducing property leads to $\int k(y,x)\,\mathrm{d}P(x) = Lf = \langle f, h\rangle_{\mathcal{H}(k)} = h(y)$, so that $h = \int k(x,\cdot)\,\mathrm{d}P(x)$, and so $\mu_P = h \in \mathcal{H}(k)$ with $Lf = \langle f, \mu_P\rangle_{\mathcal{H}_k}$, as claimed. □

Remark 3.4.2. *(Reproducing property of the expectation operation in the RKHS). As for the Dirac measure, Lemma 10 states that it is possible to compute* $\mathbb{E}_P[f] = \int f(x)\,\mathrm{d}P(x)$*, the expectation with respect to* P *of any function* $f \in \mathcal{H}_k$*, as the scalar product* $\langle f, \mu_P\rangle_{\mathcal{H}_k} = \langle f, \mathbb{E}_P[k(x,\cdot)]\rangle_{\mathcal{H}_k}$.

Remark 3.4.3. *(Information retained and characteristic kernels). The information retained by the kernel mean embedding of a distribution depends on the choice of kernel* k*. For example (inhomogeneous) polynomial kernels of degree* m *encode information up to the* m*-th moment of* P*. The* Fourier kernel *allows* μ_P *to incorporate full information about the characteristic function of* P*.*[9] *In particular, a kernel* k *is said to be characteristic if the map* $P \mapsto \mu_P$ *is injective. This ensures that*

$$\|\mu_P - \mu_Q\|_{\mathcal{H}_k} = 0 \qquad \textit{if and only} \qquad P = Q, \tag{3.4.15}$$

and there is no information loss when mapping the distribution P *into the RKHS* $\mathcal{H}_k$*, because this contains a sufficiently rich class of functions to represent all higher moments of* P*. For example, Gaussian, Laplace and Matérn kernels are characteristic, while polynomial kernels are not characteristic on* $X = \mathbb{R}^d$.

Remark 3.4.4. *(Mean embedding estimation). It is often not possible to compute the kernel mean embedding* $\mu_P = \int k(x,\cdot)\mathrm{d}P(x)$ *because this is a high-dimensional intractable integral. However, if we have i.i.d. samples* $x_1, \ldots, x_n \sim P$*, then the empirical mean embedding estimator*

$$\hat{\mu}_P = \frac{1}{n}\sum_{i=1}^{n} k(x_i,\cdot) \tag{3.4.16}$$

is unbiased and it converges to μ_P *by the law of large numbers. A wide range of alternative estimators is also available, aimed at improving certain characteristics of the estimate.*

3.4.2 Discrepancies based on kernel mean embeddings

The kernel mean embedding can be used to define a metric for probability distributions which is an important tool for problems in statistics and machine learning. The metric defined in terms of mean embeddings can be considered as a particular instance of an integral probability metric [14].

[9] The Fourier kernel is not positive definite, but the mean embedding is well defined.

Definition 7. (Integral probability metric). Given two probability measures P and Q on a measurable space X, and a class of real-valued measurable and functions $\mathcal{F} \subset L^1(P) \cap L^1(Q)$, an integral probability metric (IPM) is defined as

$$\mathrm{IPM}_{\mathcal{F}}(P,Q) = \sup_{f\in\mathcal{F}} \left| \int f(x)\mathrm{d}P(x) - \int f(y)\mathrm{d}Q(y) \right|. \tag{3.4.17}$$

Remark 3.4.5. *(Choice of $\mathcal{F}$). The function class $\mathcal{F}$ fully characterizes the* $\mathrm{IPM}_{\mathcal{F}}(P,Q)$*, and its choice is subject to a trade-off. On one hand, $\mathcal{F}$ must be rich enough so that* $\mathrm{IPM}_{\mathcal{F}}(P,Q) = 0$ *if and only if $P = Q$.*[10] *On the other hand, the larger the function class $\mathcal{F}$, the more difficult it is to compute or even estimate* $\mathrm{IPM}_{\mathcal{F}}(P,Q)$*. It is thus important to work with a class $\mathcal{F}$ that provides* meaningful and computable *IPMs. Examples of choices of $\mathcal{F}$ lead the IPM to be the* total variation distance, the Kolmogorov distance and the Wasserstein distance*, all widely used in machine learning and statistics.*

Maximum mean discrepancy. We give the following definition and alternative expression of metric based on choosing the class of test functions $\mathcal{F}$ to be an RKHS.

Definition 8. (Maximum mean discrepancy). When the supremum in (3.4.17) is taken over functions in the unit ball of an RKHS $\mathcal{H}_k$, i.e., $\mathcal{F} := \{f \in \mathcal{H}_k : \|f\|_{\mathcal{H}_k} \leq 1\}$, the resulting IPM is known as the maximum mean discrepancy (MMD), which we denote by $\mathrm{MMD}(P,Q)$.

Proposition 3.4.1. *The MMD between two probability distributions P and Q can be expressed as the distance in $\mathcal{H}_k$ between the respective mean embeddings:*

$$\mathrm{MMD}(P,Q) = \|\mu_P - \mu_Q\|_{\mathcal{H}_k}. \tag{3.4.18}$$

Proof. In fact:

$$\begin{aligned}
\mathrm{MMD}(P,Q) &= \sup_{\|f\|_{\mathcal{H}_k}\leq 1} \left| \int f(x)\mathrm{d}P(x) - \int f(y)\mathrm{d}Q(y) \right| && (3.4.19)\\
&= \sup_{\|f\|_{\mathcal{H}_k}\leq 1} \left| \langle f, \mu_P - \mu_Q\rangle_{\mathcal{H}_k} \right| && (3.4.20)\\
&= \|\mu_P - \mu_Q\|_{\mathcal{H}_k}, && (3.4.21)
\end{aligned}$$

where in (3.4.20) we used the definition of mean embedding and the linearity of the scalar product, while (3.4.21) follows from Cauchy-Schwarz and the supremum is obtained by $f = (\mu_P - \mu_Q)/\|\mu_P - \mu_Q\|$. □

Corollary 3.4.1. *The MMD between two probability distributions P and Q can thus be expressed in terms of the kernel function k corresponding to the RKHS $\mathcal{H}_k$ as*

$$\mathrm{MMD}^2(P,Q) = \iint k(x,y)\mathrm{d}P(x)\mathrm{d}P(y) - 2\iint k(x,y)\mathrm{d}P(x)\mathrm{d}Q(y) + \iint k(x,y)\mathrm{d}Q(x)\mathrm{d}Q(y). \tag{3.4.22}$$

Proof. The proof is based on computing terms of the form $\langle \mu_P, \mu_Q\rangle_{\mathcal{H}_k}$. Using lemma 10,

$$\langle \mu_P, \mu_Q\rangle_{\mathcal{H}_k} = \mathbb{E}_P[\mu_Q] = \mathbb{E}_P\left[\int k(y,\cdot)\mathrm{d}Q(x)\right] = \iint k(x,y)\mathrm{d}P(x)\mathrm{d}Q(y). \tag{3.4.23}$$

□

The use of the MMD as probability metric depends both both on its ability to separate probability distributions and its computability (or ease to estimate). This is specified in the following remarks.

[10] In this case $\mathcal{F}$ is called *measure determining*.

Remark 3.4.6. *(Relationship to other IPMs). It is possible to prove that the MMD between two probability measures P and Q is upper bounded by the Wesserstein distance and by the total variation distance.*[11] *As a consequence, if P and Q are close in these distances, they are also close in the MMD metric.*

Remark 3.4.7. *(Empirical estimate). To compute the MMD, we need to be able to calculate the integrals in (3.4.22). If this is not possible in closed form, but we can get i.i.d. samples $x = \{x_i\}_{i=1}^n \sim P$ and $y = \{y_j\}_{j=1}^n \sim Q$, then it is possible to construct estimators $\widehat{\mathrm{MMD}}(x, y)$*[12] *of $\mathrm{MMD}(P, Q)$ that are unbiased and enjoy fast convergence rate when compared to estimators of other IPMs. For example, when $X = \mathbb{R}^d$, the MMD estimator based on so called 'U-statistics' is such that*

$$|\widehat{\mathrm{MMD}}^2(x, y) - \mathrm{MMD}^2(P, Q)| \leq c_d(n^{-1/2}), \tag{3.4.24}$$

where c_d is a constant possibly depending on the dimension. This can be contrasted with estimators of the Wasserstein distance, where the bound is $O(n^{-d/2})$. MMD estimators have thus faster asymptotic convergence rate (although they might perform worse in the small sample size regime).

Remark 3.4.8. *(Applications). Some application of MMD include*

- *Two-sample testing for equality of samples: given two i.i.d. samples $\{x_i\}_{i=1}^m \sim P$ and $\{y_j\}_{j=1}^n \sim Q$, we want to test the null hypothesis $H_0 : \|\mu_P - \mu_Q\|_{\mathcal{H}_k} = 0$ against the alternative hypothesis $H_1 : \|\mu_P - \mu_Q\|_{\mathcal{H}_k} \neq 0$;*
- *Goodness-of-fit testing: we want to test if a sample is drawn from a given distribution P (this can be cast in terms of two-sample testing if we can sample from P);*
- *Optimal quantization of a probability measure P: we want to construct an empirical measure $Q_n = \frac{1}{n}\sum_{i=1}^n x_i$ using as few states x_i as possible for a given approximation quality, as measured by $\mathrm{MMD}(P, Q_n)$.*

Remark 3.4.9. *(Measure determining MMD). If the kernel k corresponding to the RKHS $\mathcal{H}_k$ used in the definition of MMD is characteristic, then $\mathcal{H}_k$ is measure determining:*

$$\mathrm{MMD}(P, Q) = 0 \Leftrightarrow P = Q. \tag{3.4.25}$$

Remark 3.4.10. *(Convergence controlling MMD). The use of measure determining MMDs is desirable, but not a strong enough property to justify its use to measure discrepancy between two probability distributions. A stronger property is weak convergence control, namely*

$$\mathrm{MMD}(P, Q_n) \to 0 \qquad \textit{implies that} \qquad Q_n \Rightarrow P, \tag{3.4.26}$$

where $Q_n \Rightarrow P$ means that $\int f \mathrm{d}Q_n \to \int f \mathrm{d}P$ for any bounded continuous function f. Certain choices of X and k guarantee convergence control, see [18].

Kernel Stein discrepancy (KSD). In the previous remarks on computation and use of the MMD between two probability distributions P and Q (or Q_n), we often think of P as the reference distribution, to test against (in two-sample and goodness-of-fit testing) or to try to approximate (in optimal quantization). Sometimes P admits density[13]

$$p(x) = \tilde{p}(x)/Z, \tag{3.4.27}$$

where Z is an intractable normalizing constant (think for example of the case when P is the posterior distribution of a Bayesian parametric model). In this case, it is often both not possible to compute the integrals against P in (3.4.22) and to get samples from P, hence empirical estimators of the MMD. In this situation, Stein discrepancy proves to be useful, and it turns out that it is possible to *kernelize* it. See [1] for a recent review. We start with:

[11] The latter under the condition that $\sup_{x \in X} k(x, x) < C$.

[12] This is a shorthand notation to express the MMD estimator as a function of the samples.

[13] We now assume P to be continuous, but similar considerations can be done for discrete P.

Definition 9. (Stein characterization of a probability measure P). A distribution P is characterised by the pair $(\mathcal{A}_P, \mathcal{G})$, consisting of a Stein Operator $\mathcal{A}_P$ and a Stein Class $\mathcal{G}$, if it holds that (Stein identity)

$$X \sim P \quad \text{iff} \quad \int \mathcal{A}_P g(x) \mathrm{d}P(x) = 0 \quad \forall g \in \mathcal{G}. \tag{3.4.28}$$

Based on this definition, it is possible to define a measure of discrepancy:

Definition 10. (Stein discrepancy). Given a Stein characterisation $(\mathcal{A}_P, \mathcal{G})$ of a probability measure P, the Stein discrepancy between a distribution P and Q is defined as the maximum deviation from the Stein identity

$$\mathrm{SD}(Q, P) = \sup_{g \in \mathcal{G}} \left| \int \mathcal{A}_P g(x) \mathrm{d}Q(x) \right|. \tag{3.4.29}$$

We can rewrite $\mathrm{SD}(Q, P)$ as

$$\mathrm{SD}(Q, P) = \sup_{f \in \mathcal{F}_P = \mathcal{A}_P \mathcal{G}} \left| \int f(x) \mathrm{d}Q(x) \right| \tag{3.4.30}$$

$$= \sup_{f \in \mathcal{F}_P = \mathcal{A}_P \mathcal{G}} \left| \int f(x) \mathrm{d}Q(x) - \int f(x) \mathrm{d}P(x) \right|, \tag{3.4.31}$$

where the last identity holds because the chosen Stein operator and class characterize P. Equation (3.4.31) *resembles* the general definition of IPM (3.4.17), but notice that $\mathrm{SD}(Q, P)$ is not a metric, because it is not symmetric in its arguments. It is thus natural to wonder whether it is possible to compute Stein discrepancies that result in $\mathcal{F}_P = \mathcal{A}_P \mathcal{G}$ being an RKHS, so that the corresponding Stein discrepancy resembles an MMD. This is indeed possible, and it follows from the following two theorems, proved in [4] and [15] respectively:[14]

Theorem 3.4.1. *(Stein characterization in RKHS). Let $k : X \times X \to \mathbb{R}$ be the reproducing kernel of a RKHS $\mathcal{H}_k$ of functions from X to $\mathbb{R}$. Suppose that k is bounded, symmetric, universal[15] and satisfies $\mathbb{E}_P[(\Delta k(x, x))^2] < \infty$. Then P has Stein characterisation $(\mathcal{A}_P, \mathcal{G})$, consisting of*

$$\mathcal{A}_P g = \frac{\nabla(gp)}{p}, \qquad \mathcal{G} = \{g \in \mathcal{H}_k : \|g\|_{\mathcal{H}_k} \leq 1\}. \tag{3.4.32}$$

Theorem 3.4.2. *(Stein discrepancy in RKHS). The functions $\mathcal{A}_P g$ just defined are precisely the elements of the unit ball in the RKHS $\mathcal{H}_{k_P} := \mathcal{A}_P \mathcal{H}_k$ with kernel*

$$\begin{aligned} k_P(x, y) \quad = \quad & \nabla_x \nabla_y k(x, y) + \frac{\nabla_x p(x)}{p(x)} \nabla_y k(x, y) \\ & + \frac{\nabla_y p(y)}{p(y)} \nabla_x k(x, y) + \frac{\nabla_x p(x)}{p(x)} \frac{\nabla_y p(y)}{p(y)} k(x, y) \end{aligned} \tag{3.4.33}$$

In particular, under regularity conditions, $\int f \mathrm{d}P = 0$, $\forall f \in \mathcal{H}_{k_P}$.

We can thus define:

Definition 11. (Kernel Stein discrepancy). The kernel Stein discrepancy between two probability measures P and Q is the Stein discrepancy $\mathrm{SD}(Q, P)$ with choice of Stein operator and class as in (3.4.32), and we write

$$\mathrm{KSD}(Q, P) = \sup_{f \in \mathcal{H}_{k_P} = \mathcal{A}_P \mathcal{H}_k} \left| \int f(x) \mathrm{d}Q(x) \right| \tag{3.4.34}$$

[14] We state these for $X = \mathbb{R}$, but similar theorems hold for $X = \mathbb{R}^d$.

[15] See e.g. [18] for a definition. For example, when X is compact, a kernel is universal if the corresponding RKHS $\mathcal{H}_k$ is dense in the space of bounded continuous functions.

Remark 3.4.11. *(Alternative expression KSD). Using a similar argument as in corollary 3.4.1 and the fact that* $\int f \mathrm{d}P = 0$, $\forall f \in \mathcal{H}_{k_P}$, *we can write*

$$\mathrm{KSD}^2(Q, P) = \iint k_P(x, y) \mathrm{d}Q(x) \mathrm{d}Q(y) \tag{3.4.35}$$

Remark 3.4.12. *(Computable KSD). Notice that the kernel* k_P *in* (3.4.33) *depends only on evaluations of the* base kernel k *and its derivatives, and on* $\frac{\nabla p}{p} = \nabla \log p = \nabla \log \tilde{p}$, *so it does not require evaluations of the normalizing constant* Z *in* (3.4.27). *We are thus often in a setting where we can evaluate* k_P. *For computing the KSD we also need to be able to compute the integral of this new kernel* (3.4.35). *This might not be possible in closed form, but in common applications of the KSD (such as quantization of* P*) we have that* Q *is an empirical measure* $Q = Q_n = \frac{1}{n} \sum_{i=1}^n x_i$, *so that*

$$\mathrm{KSD}^2(Q_n, P) = \frac{1}{n^2} \sum_{i,j=1}^n k_P(x_i, x_j). \tag{3.4.36}$$

Remark 3.4.13. *(Convergence controlling KSD). Under conditions on the measure* P *(that is* P *is distantly dissipative - a relaxation of log-concavity) and for certain choices of the base kernel* k *(e.g. when* k *is the inverse multi-quadric kernel), the KSD is convergence controlling:*

$$\mathrm{KSD}(P, Q_n) \to 0 \qquad \textit{implies that} \qquad Q_n \Rightarrow P, \tag{3.4.37}$$

and it is thus suitable for machine learning and statistics tasks such as assessing quality of a sample or optimal quantization of P.

Chapter 4

Gaussian Processes

4.1 Motivation

In chapters 1 and 2 our goal was to fit a function to data while limiting complexity to avoid overfitting. We showed that the representer theorem provides the unique and deterministic optimal solution f^* to KRR. In this chapter we aim at incorporating a measure of uncertainty in the solution to a regression problem: this is an important shift of paradigm, based on recognizing that there might be many nearly as optimal solutions as f^*[1]. In particular, we present *Gaussian process (GP) regression*, a tool for modelling random functions via the *Bayes' update*: a prior belief on a function is updated to a posterior belief through observations. In this setting, uncertainty corresponds to the posterior covariance function, and we will draw some connections between the solution to GP and KRR regression, following the excellent review in [10]. We start with a simple example of Bayes' update, where the quantity treated as random is a parameter in a parametric family.

Bayesian parameter inference Let $x = x_1, \dots, x_n$ be realizations of the real-valued random variables $X_1, \dots, X_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mu, \sigma^2)$. We assume that the variance σ^2 is known and our goal is to infer μ. In the frequentist framework μ is assumed to be a fixed but unknown quantity and inference is solved by producing point estimators. For example, the sample mean is the maximum likelihood estimator $\hat{\mu} = \bar{x}$, which maximizes the *likelihood* $p(x|\mu)$, that is, the joint probability density function of the sample studied as a function of μ,

$$p(x|\mu) = \prod_{i=1}^{n} p(x_i|\mu). \tag{4.1.1}$$

In the Bayesian framework we instead assume that the parameter μ is unknown but random, so we can assign a probability distribution to it. Uncertainty about μ can be formulated *before and after* observing the realizations x. The former corresponds to assigning to μ a *prior* distribution $p(\mu)$, which encodes our belief about the parameter, for example elicited from experts in a certain applied domain.[2] Once an experiment is performed and the sample x is obtained, the prior belief can be updated through Bayes' theorem, to obtain the *posterior* distribution $p(\mu|x)$,

$$p(\mu|x) = \frac{p(x|\mu)p(\mu)}{p(x)} \propto p(x|\mu)p(\mu), \tag{4.1.2}$$

where $p(x|\mu)$ is the likelihood defined in (4.1.1) and $p(x) = \int p(x|\mu)p(\mu)\mathrm{d}\mu$ is the marginal likelihood, a normalizing constant that can (often) be omitted because it does not depend on μ. The shape of the posterior distribution represents uncertainty on μ after observing the data. Point estimators can

[1] This is even more important for non-convex problems that might not have a unique minimizer.

[2] There is a degree of freedom in setting the prior distribution. A sensible choice is required especially in a small data regime, because it affects the inference, while the effect of prior choice becomes negligible as $n \to \infty$.

be obtained also in the Bayesian framework by taking a decision theoretic approach and minimizing expected loss functions (where expectations are computed against the posterior). The posterior mode, also called *maximum a posteriori* (MAP) is

$$\mu_{\text{MAP}} \in \arg\max_{\mu\in\mathbb{R}} p(\mu|x) = \arg\max_{\mu\in\mathbb{R}} \left(\log p(x|\mu) + \log p(\mu)\right) \tag{4.1.3}$$

and it can be considered as a regularized maximum likelihood estimator, where the prior acts as as regularizer.

In our example we can choose $p(\mu) = \mathcal{N}(\mu_0, \sigma_0^2)$, [3] where e.g. μ_0 and σ_0^2 are the mean and variance hyperparameters of the prior. Then by rearranging the terms and completing the square it is possible to prove that

$$p(\mu|x) = \mathcal{N}(\mu_{\text{post}}, \sigma^2_{\text{post}}) = \mathcal{N}(n\tau\bar{x} + \tau_0\mu_0, (n\tau + \tau_0)^{-2}), \tag{4.1.4}$$

where we defined the precision parameters $\tau = 1/\sigma$ and $\tau_0 = 1/\sigma_0$. The posterior mean is a weighted average of prior mean and sample mean (the maximum likelihood estimate), and the posterior precision is the sum of prior and data precisions, so that uncertainty about μ, quantified by the posterior variance, has decreased after observing the data. In this example posterior mean, mode and median coincide.

4.2 Gaussian processes

We can extend the motivating example to a more ambitious task: perform Bayesian inference on a function, rather then just a parameter, thus setting the target of inference to be an infinite-dimensional stochastic process. *Gaussian processes* (GPs) extend the definition of a Gaussian random variables to random functions, as follows.

Definition 12 (Gaussian process)**.** Let $X \neq \emptyset$ be a set and consider the functions $m : X \to \mathbb{R}$ and $k : X \times X \to \mathbb{R}$, called mean and covariance function, respectively. A random function $f : X \to \mathbb{R}$ is called Gaussian process with mean m and covariance k, and we write $\mathcal{GP}(m, k)$, if, for any $n \in \mathbb{N}$ and for any choice of points $\mathcal{D} = \{x_1, \dots, x_n\}, x_i \in X,\ i = 1, \dots n$, the vector of function evaluations $(f(x_1), \dots, f(x_n))^T \in \mathbb{R}^n$ is Gaussian distributed with mean

$$m_{\mathcal{D}} = (m(x_1), \dots, m(x_n))^T \in \mathbb{R}^n, \tag{4.2.1}$$

and covariance matrix

$$k_{\mathcal{D}\mathcal{D}} = \begin{pmatrix} k(x_1, x_1) & \dots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \dots & k(x_n, x_n) \end{pmatrix} \in \mathbb{R}^{n\times n}. \tag{4.2.2}$$

Remark 4.2.1. *(Abstract view on definition of Gaussian process). Definition 12 can be viewed as an extension of the characterization of finite-dimensional Gaussianity to functions. In fact, recall that if Y is a random variable in $\mathbb{R}^d$, then Y is multivariate Gaussian if and only if all one-dimensional projections of Y are scalar Gaussian random variables, that is iff*

$$\langle Y, v\rangle_{\mathbb{R}^d} = Y^T v \sim \mathcal{N}(\mu_v, \sigma_v^2) \qquad \textit{for any } v \in \mathbb{R}^d. \tag{4.2.3}$$

Then, given that we can view functions $f : X \to \mathbb{R}$ as infinite-dimensional vectors, the evaluation of f at a finite number of points $x_1, \dots, x_n \in X$ amounts to projecting an infinite-dimensional vector on a subspace of dimension n.

Remark 4.2.2. *(One-to-one correspondence with kernels). Since the covariance matrix $k_{\mathcal{D}\mathcal{D}}$ has to be symmetric and positive semi-definite for any choice of $\mathcal{D}$, it follows that the function $k : X \times X \to \mathbb{R}$*

[3] This is a shorthand notation to indicate that μ is random variable a priori Gaussian distributed.

in definition 12 ought to be a positive definite kernel, recall remark 1.2.1. Conversely, it is possible to prove[4] *that for every function $m : X \to \mathbb{R}$ and positive definite kernel $k : X \times X \to \mathbb{R}$ there exists a unique Gaussian process $\mathcal{GP}(m, k)$, so we can in fact establish a one-to-one correspondence between positive definite kernels and Gaussian processes. For this reason the function k in definition 12 is often called 'covariance kernel'.* [5]

Remark 4.2.3. *(Properties of GP sample paths). The GP mean function and covariance kernel induce certain properties, such as smoothness of sample realizations of the GP. In particular, it is natural to wonder whether GP realizations lie in the RKHS $\mathcal{H}_k$ corresponding to the covariance kernel k if $m \in \mathcal{H}_k$. It turns out that this is not the case if $\mathcal{H}_k$ is infinite dimensional, and GP sample paths fall outside of $\mathcal{H}_k$ almost surely, even if the mean function belongs to it. However, it is possible to prove that they actually lie on certain RKHSs $\mathcal{H}_k^\theta$, $\theta \in (0, 1)$, defined as* powers of *$\mathcal{H}_k$. The characterization of $\mathcal{H}_k^\theta$ goes through Mercer's theorem characterization of $\mathcal{H}_k$. Here it will be sufficient to notice that that the following inclusion holds*

$$\mathcal{H}_k = \mathcal{H}_k^1 \subset \mathcal{H}_k^\theta \subset \mathcal{H}_k^{\theta'} \subset L^2(\nu), \qquad \text{for all } 1 > \theta > \theta' > 0, \tag{4.2.4}$$

where ν is a finite Borel measure with support X and $L^2(\nu) = \{f : X \to \mathbb{R} : \int f(x)^2 \mathrm{d}\nu(x) < \infty\}$. Thus $\mathcal{H}_k^\theta$ gets larger (less smooth) as θ decreases, where θ needs to satisfy summability criteria related to Mercer's theorem for $\mathcal{H}_k$. In certain cases (e.g when k is a square-exponential kernel) $\mathcal{H}_k^{\theta'}$ is only infinitesimally larger than $\mathcal{H}_k$ and it is possible to take $\theta \to 1$; in other cases (e.g. when k is a Matérn kernel) there is a sharp gap between the two spaces (for example GP realizations when the RKHS of the covariance kernel is the Sobolev space of order 1 coincide with Brownian motion sample paths). See [10, Chapter 4].

4.3 GP-regression

As in the KRR setting, we assume to be given data $(x_i, y_i)_{i=1}^N$, with $x_i \in X$ and $y_i \in \mathbb{R}$. Our goal is to give a description of the data by means of a function $f : X \to \mathbb{R}$. In chapter 2 we studied how a solution to this problem can be found by setting an optimization problem that trades off data-fidelity and function complexity, by imposing f to belong to an RKHS and have finite norm. We now take a Bayesian approach and model f as a random function (stochastic process). In particular we need to specify

Prior distribution for f: a natural choice is given by modelling $f \sim \mathcal{GP}(m, k)$ a priori, for some mean function $m : X \to \mathbb{R}$ and covariance kernel $k : X \times X \to R$. This allows to incorporate certain properties assumed a prior on f, such as periodicity and (non-)smoothness, see remarks 4.2.3 and 4.3.6.

Likelihood. We assume that the data generating process is the function f observed in zero-mean Gaussian additive noise, so that for any $i = 1, \dots, N$

$$y_i = f(x_i) + \xi_i, \qquad \xi_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2). \tag{4.3.1}$$

We assume that σ^2 is known. The one-point data likelihood is thus given by

$$p(y_i|f) = \mathcal{N}(y_i|f(x_i), \sigma^2) \propto \exp\left(-\frac{(y_i - f(x_i))^2}{2\sigma^2}\right), \tag{4.3.2}$$

[4] This is not trivial (and in fact somewhat surprising: an infinite dimensional object is characterised by finite dimensional projections), and it goes through the Kolmogorov extension theorem, that guarantees existence of stochastic process with given *consistent* finite dimensional projections.

[5] Notice that the matrix $k_{\mathcal{D}\mathcal{D}}$ can be singular, in which case the corresponding finite dimensional Gaussian distribution is well defined, but it does not have a density function (with respect to the Lebesgue measure).

and the complete likelihood is

$$p(y_1, \dots, y_N | f) = \prod_{i=1}^{N} \mathcal{N}(y_i | f(x_i), \sigma^2). \tag{4.3.3}$$

Following Bayes theorem (applied to distributions with infinite dimensional support) we have that the chosen prior and likelihood are *conjugate*. In fact the following holds:

Theorem 4.3.1. *(Posterior in GP regression). The posterior in the Bayesian learning problem specified above is the Gaussian process f_y,*

$$f_y \sim \mathcal{GP}(\bar{m}, \bar{k}), \tag{4.3.4}$$

with updated mean function $\bar{m} : X \to \mathbb{R}$ and covariance kernel $\bar{k} : X \times X \to \mathbb{R}$, where

$$\bar{m}(x) = m(x) + k_{x\mathcal{D}}(k_{\mathcal{D}\mathcal{D}} + \sigma^2 I_{NN})^{-1}(y - m_{\mathcal{D}}), \qquad x \in X \tag{4.3.5}$$

$$\bar{k}(x, x') = k(x, x') - k_{x\mathcal{D}}(k_{\mathcal{D}\mathcal{D}} + \sigma^2 I_{NN})^{-1} k_{\mathcal{D}x'}, \qquad x, x' \in X \tag{4.3.6}$$

and

$$k_{\mathcal{D}x} = k_{x\mathcal{D}}^T = (k(x_1, x), \dots, k(x_N, x))^T \in \mathbb{R}^N, \tag{4.3.7}$$

and $k_{\mathcal{D}\mathcal{D}}$ as in (4.2.2), $m_{\mathcal{D}}$ as in (4.2.1), $y = (y_1, \dots, y_N)^T$ and σ^2 is the noise level assumed in the likelihood.

Remark 4.3.1. *(Proof without Bayes' theorem). Theorem 4.3.1 can be proved also without Bayes' theorem, relying only on the properties of conditional distributions in a Gaussian random vector,[6] Kolmogorov extension theorem and the definition of a GP. In fact applying Bayes' theorem to stochastic processes is involved and it requires working with the Radon-Nikodym derivative for performing a change of measure from prior to posterior.*

Remark 4.3.2. *(Connection to the Kalman Filter). In the case when the underlying set X is finite, the GP posterior coincides with the Kalman update.*

Remark 4.3.3. *(Predictions). The posterior mean and covariance kernel can be used for predictions at a new data point $\tilde{x}$. In fact, the prediction at the new point is $\tilde{y} = \bar{m}(\tilde{x}) = \mathbb{E}\left[f_y(x) | (x_i, y_i)_{i=1}^N\right]$, while the variance of $f_y(\tilde{x}) = \bar{k}(\tilde{x}, \tilde{x})$ quantifies uncertainty about the prediction. See Figure 4.3.2.*

Remark 4.3.4. *(Hyperparameter tuning). The GP regression setting requires choosing the values of the hyperparameters appearing in the prior mean m and covariance k, as well as the noise variance σ^2. A common approach is selecting values that maximize the marginal likelihood of the data (this is the denominator of Bayes' theorem, which we did not write explicitly for GPs, but conceptually corresponds to the term $p(x)$ in (4.1.2)). The importance of hyper-parameter turning can be seen in the light of remarks 4.2.3 and 4.3.6.*

4.3.1 Relationship between KRR and GP regression

From inspection of (4.3.5) we see the following:

Theorem 4.3.2. *If we choose $m(x) = 0$ and the same kernel for both KRR and GP regression, then the solution to the KRR problem coincides with the posterior mean $\bar{m}$ (4.3.5) if*

$$\sigma^2 = N\lambda, \tag{4.3.8}$$

where σ^2 is the noise in the observations, N is the number of datapoints, and λ is the regularization parameter in KRR (balancing overfitting and underfitting).

[6] In a way similar to the closed form computations in the motivating example (4.1.4).

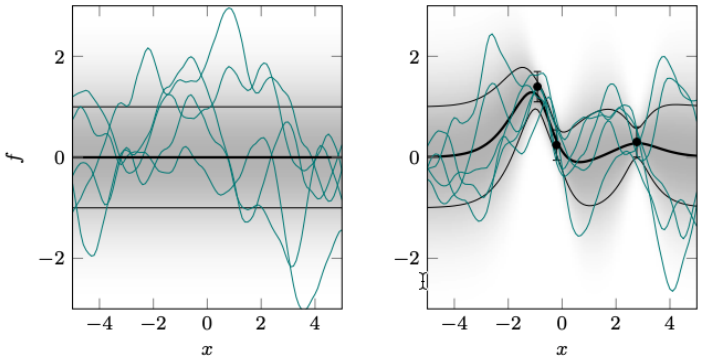


Figure 4.3.1: Left: GP samples drawn from the prior distribution. Right: GP samples drawn from the posterior distribution after three datapoints have been observed. The prior and posterior mean are shown as solid lines and the shaded regions denotes twice the standard prior and posterior deviation at each input value. Image credit: [10].

Remark 4.3.5. *(Trade off data-regularization). Condition (4.3.8) is intuitive in that it represents the equivalence between data noise variance σ^2 in GP regression and regularization level λ in KRR. In fact, if λ is small in KRR, then there is little regularization in the KRR problem, so the solution prioritizes adherence to data (and little importance is given to function smoothness). Accordingly, if σ^2 is small in GP regression, then we believe that the there is little noise in the data, and the likelihood dominates the prior in the posterior formulation.*

Remark 4.3.6. *(Posterior mean in prior RKHS). A consequence of theorem 4.3.2 is that the posterior mean function $\bar{m}$ belongs to the RKHS corresponding to the prior covariance kernel $\mathcal{H}_k$. Thus when specifying the prior covariance kernel we can impose certain properties of the posterior mean. However, following arguments similar to those in remark 4.2.3, posterior samples $f \sim \mathcal{GP}(\bar{m}, \bar{k})$ do not belong to $\mathcal{H}_k$.*[7]

4.3.2 GP interpolation

We can consider the data to be noise-free observations of the function $f : X \to \mathbb{R}$ so that for any $i = 1, \dots, N$

$$y_i = f(x_i). \tag{4.3.9}$$

In this case it is not possible to formally apply Bayes' theorem because the likelihood is degenerate, but following Gaussian calculus, and in case the matrix $k_{\mathcal{DD}}$ is invertible,[8] it is possible to establish an expression of the GP posterior distribution formally equivalent to that in Theorem 4.3.1 with $\sigma^2 = 0$.

The noise-free deterministic representation of data suits well the scenario in which the function values $f(x_i)$ represent the outcome of computer experiments. In this case GP interpolation is also called *model emulation* and it is mostly used for cases when the computer model is expensive to run and any operation that involves function evaluations should then be done parsimoniously. Modelling the function f as a Gaussian process allows to do so, by incorporating prior knowledge on the function and refining its characterization as new observations (computer runs) are obtained. We give an example below.

Example 4.3.1. (Bayesian optimization). The goal is to find the minimizer of a a function $V : X \to \mathbb{R}$

$$x^* \in \arg\min_{x \in X} V(x), \tag{4.3.10}$$

[7] They do not belong to $\mathcal{H}_{\bar{k}}$ either. In fact $\mathcal{H}_{\bar{k}} \subset \mathcal{H}_k$, can you prove this as an exercise? Hint: consider the expression of the posterior kernel (4.3.6).

[8] If $k_{\mathcal{DD}}$ is not invertible, then GP interpolation is formally not well defined. In practice, the problem is regularized by assuming the existence of small noise in the output (also called jitter). Thus the examples that we provide for GP interpolation are in practice used in the regression setting and jitter is sometimes interpreted as model discrepancy term.

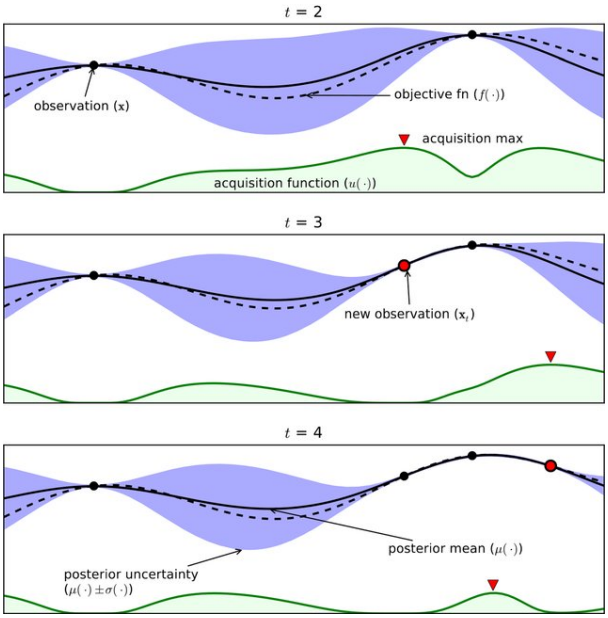


Figure 4.3.2: Illustration of three iterations of Bayesian optimization. The objective function is here shown as dashed line, but it is in practice unknown. The acquisition function is shown on the bottom and it trades off exploitation and exploration of the GP interpolating model. Image credit: [17].

in settings when the evaluation of V is computationally or experimentally expensive. Optimization is performed as follows. Select an initial point x_0 and evaluate $V(x_0)$. Then:

Step 1: Model V by a GP: $V \sim \mathcal{GP}(m, k)$;

Step 2: Use the GP to find a new evaluation point x_i and evaluate $V(x_i)$;

Step 3: Perform Bayes' update using the new datapoint $(x_i, V(x_i))$, to get the GP posterior;

Step 4: Go to Step 1 and iterate until convergence.

Notice that the algorithm requires evaluation of the expensive model, but this is done efficiently. In fact, in *Step 2* the new point x_i is found by maximizing an objective function (acquisition function), which trades off exploitation (high GP mean) and exploration (high GP variance). This does not require iterative evaluations of the function V.

Remark 4.3.7. *(Kernel interpolation). It is possible to determine an analogous interpolation problem to KRR (to be interpreted as the smoothest function in the RKHS that passes through the data), and this is again equivalent to the posterior mean of the GP interpolation problem (provided the matrix $k_{\mathcal{D}\mathcal{D}}$ is invertible).*

Chapter 5

The Neural Tangent Kernel

5.1 Differentiable learning

In this section, we go back to the motivation from Section 1.1: the goal is to find a function f^* that describes available data $(x_i, y_i)_{i=1}^N$. For this, we now consider a parameterisation $(f(\cdot;\theta))_{\theta\in\Theta}$, with $\Theta \subset \mathbb{R}^p$, and intend to find a parameter $\theta^* \in \Theta$ so that $f(x_i;\theta^*) \approx y_i$, $i = 1,\dots,N$. The primary example to consider is when the parameterisation is in terms of a neural network (see below), that is, the parameter space Θ comprises the weights and biases.

Example 5.1.1 (Two-layer neural network)**.** The function class given by

$$f(x;\theta) = \frac{1}{M}\sum_{i=1}^{M} b_i \sigma(a_i \cdot x) \tag{5.1.1}$$

describes neural networks with one hidden layer (M hidden neurons). Here, $\sigma : \mathbb{R} \to \mathbb{R}$ is a nonlinearity, and $\theta = (a_i, b_i)_{i=1}^M$ is the collection of parameters. The hidden layer is parameterised by the weights $a_i \in \mathbb{R}^M$, and the output layer by $b_i \in \mathbb{R}$.

Example 5.1.2 (Kernel machine)**.** According to the representer theorem (Theorem 2.0.1), the solution to the kernel ridge regression problem is given in the form

$$f(x;\theta) = \sum_{i=1}^{N} \theta_i k(x_i, x). \tag{5.1.2}$$

Kernel learning can thus be phrased as the task of determining the coefficients $\theta = (\theta_1,\dots,\theta_N^\top)$.

Example 5.1.3 (Tensor networks)**.** Another interesting class of function approximators is the one given by tensor networks, see, for example, [16].

To find θ^*, we define an optimality criterion (loss function) and optimise θ using gradient-descent type algorithms. For example,

$$\mathcal{L}(f) = \frac{1}{N}\sum_{i=1}^{N} \left(f(x_i) - y_i\right)^2 = \int_{\mathbb{R}^d} \left(f(x) - f^*(x)\right)^2 \rho(\mathrm{d}x) \tag{5.1.3}$$

represents the data term in (2.0.3). In the second formulation in (5.1.3) have introduced the empirical measure $\rho = \frac{1}{N}\sum_{i=1}^N \delta_{x_i}$, as well as the target function f^* satisfying $f^*(x_i) = y_i$. From another (more fundamental) perspective, the middle term in (5.1.3) represents the *empirical risk* (training error), whereas the right-most expression represents the *true risk* (test error). See [5, Chapter I,1] for more information.

Remark 5.1.1. *More generally, we can consider*

$$\mathcal{L}(f) = \int_{\mathbb{R}^d} (f(x) - y)^2 \, \rho(\mathrm{d}x\mathrm{d}y). \tag{5.1.4}$$

This corresponds to the setting when the y-values are observed with noise. Again, see [5, Chapter I,1] for a detailed discussion.

To find θ^*, it is natural to minimise (5.1.3) using gradient-descent type algorithms, given, in their simplest form, as time-discretisations of

$$\partial_t \theta_t = -\nabla_{\theta_t} \mathcal{L}(f(\cdot; \theta_t)). \tag{5.1.5}$$

More generally, (5.1.5) might be replaced by a stochastic version, or augmented with momentum. In the case of (5.1.3), we obtain

$$\partial_t \theta_t = -\nabla_{\theta_t} \left(\frac{1}{N} \sum_{i=1}^{N} (f(x_i; \theta_t) - y_i)^2 \right), \tag{5.1.6}$$

and hope that $\theta^* = \lim_{t \to \infty} \theta_t$ solves the learning problem in an appropriate sense. The process of evolving (5.1.6) is commonly referred to as training.

Linear vs. nonlinear parameterisations. Comparing Example 5.1.1 and 5.1.2, we see that the map $\theta \mapsto f(\cdot, \theta)$ is linear in the second case, and nonlinear in the first case. In the linear case, the map $\theta \mapsto \mathcal{L}(f(\cdot; \theta))$ is convex, and the solution to the dynamics (5.1.6) converges to a global minimum. In the nonlinear case, the dynamics (5.1.6) is much harder to understand. In particular, we can ask the following questions:

1. There might be many local and global minimima, and the loss landscape is typically nonconvex. Will the dynamics converge to a global minimum?
2. If there are multiple global minima, will there be 'good' and 'bad' ones (in terms of generalisation)?

5.2 The neural tangent kernel

To give partial answers to those questions, it turns out to be useful to shift attention to the evolution of $f(\cdot; \theta_t)$. Using the chain rule, we calculate

$$\partial_t f(x; \theta_t) = \nabla_{\theta_t} f(x; \theta_t) \cdot \partial_t \theta_t = -\nabla_{\theta_t} f(x; \theta_t) \cdot \nabla_{\theta_t} \mathcal{L}(f(\cdot; \theta_t)) \tag{5.2.1}$$

and

$$\nabla_\theta \mathcal{L}(f(\cdot; \theta)) = \frac{2}{N} \sum_{i=1}^{N} (f(x_i) - y_i) \, \nabla_\theta f(x_i; \theta). \tag{5.2.2}$$

Writing $f_t(\cdot) := f(\cdot; \theta_t)$ and combining (5.2.1) with (5.2.2), we obtain the key equation

$$\partial_t f_t(\cdot) = -\frac{2}{N} \sum_{i=1}^{N} \nabla_{\theta_t} f(\cdot; \theta_t) \cdot \nabla_{\theta_t} f(x_i; \theta_t) \, (f_t(x_i) - y_i) \tag{5.2.3a}$$

$$= -\frac{2}{N} \sum_{i=1}^{N} H_{\theta_t}(\cdot, x_i) \, (f_t(x_i) - y_i), \tag{5.2.3b}$$

where we have defined the *neural tangent kernel* (NTK)

$$H_\theta(x, y) := \nabla_\theta f(x; \theta) \cdot \nabla_\theta f(y; \theta). \tag{5.2.4}$$

A short calculation shows that H_θ is indeed a positive definite kernel (see Definition 1), for all $\theta \in \Theta$.[1] The equation (5.2.3) is *almost* a closed equation for f, noting that the only nonautonomous dependence is via H though θ_t. We will see below that in certain regimes $\theta_t \approx \text{const.}$ is a good approximation, so that the analysis of (5.2.3) simplifies considerably.

Remark 5.2.1 (A geometric viewpoint)**.** *We can view the loss functional* (5.1.3) *as a mapping* $\mathcal{L} : L^2(\rho) \to \mathbb{R}$ *with Fréchet derivative*

$$D\mathcal{L} = 2(f - f^*). \tag{5.2.5}$$

This can (formally) be verified by calculating

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\Big|_{\varepsilon=0} \mathcal{L}(f + \varepsilon\psi) = \langle D\mathcal{L}, \psi\rangle_{L^2(\rho)}, \tag{5.2.6}$$

for all (reasonable) perturbations ψ*. Using this, we can write* (5.2.3) *succinctly as*

$$\partial_t f = -\int_{\mathbb{R}^d} H_{\theta_t}(\cdot, z) D\mathcal{L}(f)(z)\rho(\mathrm{d}z) = -T_{\theta_t,\rho} D\mathcal{L}(f), \tag{5.2.7}$$

with reference to the integral operator

$$T_{\theta,\rho}\phi(x) = \int_{\mathbb{R}^d} H_\theta(x, y)\phi(y)\,\rho(\mathrm{d}x). \tag{5.2.8}$$

The formulation (5.2.7) *bears similarities to gradient flows on Riemannian manifolds. To be precise, recall that if* (M, g) *is a (finite-dimensional) Riemannian manifold with a smooth potential* $V : M \to \mathbb{R}$*, then we can consider*

$$\frac{\mathrm{d}X}{\mathrm{d}t} = -\mathrm{grad}_g V(X) = -g^{\#}(\mathrm{d}V)(X), \tag{5.2.9}$$

where grad_g *refers to the Riemannian gradient, which can be expressed in terms of the musical isomorphism* $g^{\#} : T^*M \to TM$ *and exterior derivative. In local coordinates,* (5.2.9) *takes the form*

$$\partial_t X^\mu = -g^{\mu\nu}\partial_\nu V(X), \tag{5.2.10}$$

employing Einstein's summation convention and where $g^{\mu\nu}$ *denote the coordinates of the inverse of the metric tensor. Comparing* (5.2.3) *to* (5.2.10)*, we arrive at the following interpretation: The neural network* f *evolves according to a gradient flow dynamics (steepest descent) in a Riemannian manifold, whose metric tensor is given by the inverse of the integral operator defined in* (5.2.8)*.*

To finish this section, we stress the following:

1. The evolution equations (5.2.3) and (5.2.7) are completely general (no reference to neural networks at this point, only to the parameterisation $f(\cdot;\theta)$), and it is exact (no approximation of any kind).

2. However, those equations are not self-contained as the NTK H depends on θ.

In the following two sections we will study two scenarios where H becomes independent of θ, and hence the equations (5.2.3) and (5.2.7) can be studied very effectively.

[1] It is clear that H_θ is symmetric. To check positive definiteness, note that $\sum_{ij} \alpha_i\alpha_j \nabla_\theta f(x_i;\theta) \cdot \nabla_\theta f(x_j;\theta) = \big(\sum_i \alpha_i \nabla_\theta f(x_i;\theta)\big)^2 \geq 0$.

5.3 Kernel machines

We return to Example 5.1.2 and compute the NTK according to (5.2.4):

$$H_\theta(u,v) = \sum_{i=1}^{N} k(u,x_i)k(x_i,v) = \int_{\mathbb{R}^d} k(u,x)k(x,v)\rho(\mathrm{d}x). \tag{5.3.1}$$

Here, the key observation is that H does not depend on θ (this is of course a consequence of the fact that the parameterisation $\theta \mapsto f(\cdot;\theta)$ is linear). In the following, we drop the θ from the notation. The evolution equation for f takes the friendly form

$$\partial_t f_t(\cdot) = -\frac{2}{N}\sum_{i=1}^{N} H(\cdot,x_i)\left(f_t(x_i) - f^*(x_i)\right). \tag{5.3.2}$$

To solve this equation and gain more insight, it is convenient to introduce the *residual* $E_t := f_t - f^*$, so that

$$\partial_t E_t = -\frac{2}{N}\sum_{i=1}^{N} H(\cdot,x_i)E_t(x_i), \tag{5.3.3}$$

which is a linear evolution equation in E. Let us summarise the solutions to (5.3.2) and (5.3.3) as follows:

Proposition 5.3.1. *Let $\boldsymbol{H} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{E}_0 \in \mathbb{R}^N$ be given by $\boldsymbol{H}_{ij} = H(x_i,x_j)$ as well as $\boldsymbol{E}_0 = (E_0(x_1),\dots,E_0(x_N)))^\top$. Furthermore, let $\boldsymbol{h}:\mathbb{R}^N \to \mathbb{R}^N$ be given by $\boldsymbol{h} = ((H(\cdot,x_1),\dots,H(\cdot,x_N)))$. Assume that $\boldsymbol{H}$ is invertible. Then the (unique) solutions to (5.3.2) and (5.3.3) are given by*

$$f_t = f_0 + \boldsymbol{h}\cdot\boldsymbol{H}^{-1}\left(\exp\left(-\frac{2t}{N}\boldsymbol{H}\right) - \boldsymbol{I}_{N\times N}\right)\boldsymbol{E}_0, \tag{5.3.4}$$

and

$$E_t = E_0 + \boldsymbol{h}\cdot\boldsymbol{H}^{-1}\left(\exp\left(-\frac{2t}{N}\boldsymbol{H}\right) - \boldsymbol{I}_{N\times N}\right)\boldsymbol{E}_0, \tag{5.3.5}$$

We can take the limit $t\to\infty$ in the explicit solution (5.3.4). Remarkably, we recover the solution to kernel ridge regression problem (see Theorem 2.0.2)!

Corollary 5.3.1. *Assume that $f_0 = 0$ (and still that $\boldsymbol{H}$ is invertible. Then in the long-time limit, we recover the solution to the kernel ridge regression problem (2.0.3), with $\lambda = 0$, that is, $\lim_{t\to\infty} f_t$ is given by (2.0.15) and (2.0.16).*

Before giving the proofs of Proposition 5.3.1 and Corollary 5.3.1, we give a few remarks.

Remark 5.3.1.

1. *Corollary 5.3.1 shows that the solution to the kernel ridge regression problem can be found by integrating the dynamics (5.3.2), and so in some sense (5.3.2) solves the linear system (2.0.16) iteratively (continuously in time). This can indeed be useful computationally when N is very large, because no matrix inverses are required.*

2. *The equation (5.3.3) is interesting, as updates to the residual happen in the subspace*

$$\left\{\sum_{i=1}^{N}\alpha_i H(\cdot,x_i): \quad \alpha_i\in\mathbb{R}\right\} \subset \mathcal{H}_H. \tag{5.3.6}$$

It is also possible to analyse the dynamics (5.3.3) spectrally, by diagonalising the involved linear operator. It can then be seen that high-frequency modes decay first, and regularisation (in the sense of avoiding overfitting) can be obtained by early stopping. This idea is developed more fully in [9, Section 5].

3. *The fact the the dynamics (5.3.2) recovers the minimum-norm interpolation of the data (see (2.0.17)) is an instance of the following phenomenon: Statistics and computation cannot be thought of separately, but have to be considered in tandem (historically, these two subjects tended to be treated separately, but arguably 'machine learning' is to some extent about uniting both disciplines). Here, gradient descent (an algorithm) helps to identify a statistically favourable solution, avoiding overfitting.*

Proof of Proposition 5.3.1 and Corollary 5.3.1. The solutions (5.2.3) and (5.3.5) can be verified by directly taking derivatives. A more constructive approach is to first solve (5.3.3) *partially*, in the sense that we determine the dynamics of the vector $\boldsymbol{E}_t := (E_t(x_1), \dots, E_t(x_N))^\top \in \mathbb{R}^N$. Defining the matrix $\boldsymbol{H} \in \mathbb{R}^{N\times N}$ with entries $\boldsymbol{H}_{ij} = H(x_i, x_j)$, we observe that $\boldsymbol{E}_t$ satisfies the ODE

$$\partial_t \boldsymbol{E}_t = -\frac{2}{N}\boldsymbol{H}\boldsymbol{E}_t, \tag{5.3.7}$$

with solution given by

$$\boldsymbol{E}_t = \exp\left(-\frac{2t}{N}\boldsymbol{H}\right)\boldsymbol{E}_0. \tag{5.3.8}$$

We next introduce the function-valued vector $\boldsymbol{h} = ((H(\cdot, x_1), \dots, H(\cdot, x_N)))$, so that (5.3.3) can be written in the form

$$\partial_t E_t = -\frac{2}{N}\boldsymbol{h}\cdot\boldsymbol{E}_t. \tag{5.3.9}$$

We can then compute

$$E_t = E_0 - \frac{2}{N}\boldsymbol{h}\cdot\int_0^t \boldsymbol{E}_s\,\mathrm{d}s = E_0 - \frac{2}{N}\boldsymbol{h}\cdot\int_0^t \exp\left(-\frac{2t}{N}\boldsymbol{H}\right)\boldsymbol{E}_0\,\mathrm{d}s \tag{5.3.10a}$$

$$= E_0 + \boldsymbol{h}\cdot\boldsymbol{H}^{-1}\left(\exp\left(-\frac{2t}{N}\boldsymbol{H}\right) - \boldsymbol{I}_{N\times N}\right)\boldsymbol{E}_0. \tag{5.3.10b}$$

Now, (5.3.4) directly follow by plugging in $E_t = f_t - f^*$.

For Corollary 5.3.1, we take the limit in (5.3.4) to obtain

$$\lim_{t\to\infty} f_t = \boldsymbol{h}\cdot\boldsymbol{H}^{-1}\boldsymbol{y}, \tag{5.3.11}$$

with $\boldsymbol{y} = (f^*(x_1), \dots, f^*(x_N))$. From (5.3.1) we see that $\boldsymbol{H} = \boldsymbol{K}^2$ and $\boldsymbol{h} = \boldsymbol{k}\boldsymbol{K}$, with $\boldsymbol{K}_{ij} = k(x_i, x_j)$ and $\boldsymbol{k}_i = k(\cdot, x_i)$, and so the result follows by comparison with (2.0.15) and (2.0.16). □

5.4 Lazy training in neural networks

For example 5.1.1 (and, more general neural networks), the dynamics (5.2.3) is more difficult to analyse, since H depends on θ. However, the following was observed in [9]:

In the limit when the hidden layer(s) become infinitely wide, and under appropriate initialisation of the weights, the NTK remains constant during training. As a consequence, neural network training is well-described by (5.3.2), and the network acts like a kernel machine. This observation answers the questions posed at the end of Section 5.1.

Why is this so?

In this limiting regime, it turns out that the individual weights $\theta \in \Theta$ do not move (or move only infinitesimally), but $f(\cdot;\theta)$ does change substantially. This is possible because in the limit there are infinitely many weights that contribute to $f(\cdot;\theta)$. This phenomenon is called lazy training.

In the following, we aim at making these claims (somewhat) precise. For full rigour, we refer to the excellent exposition in [12, Appendix H]. To arrive at the limit, it is clear that (5.1.1) has to be rescaled

appropriately, and so we introduce

$$f^\alpha(x;\theta) := \frac{\alpha}{M}\sum_{i=1}^{M} b_i\sigma(a_i\cdot x). \tag{5.4.1}$$

In the main, two choices for the scale parameter make sense:

1. $\alpha = \mathcal{O}(1)$. This leads to 'law of large number' type results, and a description in terms of the empirical measure $\rho = \frac{1}{M}\sum_{i=1}^{M}\delta_{\theta_i}$ suggests itself. As $M\to\infty$, we may work with measures that have full support on Θ and analyse their dynamics. This is not the kernel regime, but see [12] for details.
2. $\alpha = \mathcal{O}(\sqrt{M})$. This is a 'central limit theorem' type scaling and leads to *lazy training*.

First, let us discuss in which sense the scaling $\alpha = \mathcal{O}(\sqrt{M})$ makes sense:

Lemma 11. *Let $a_i, b_i \sim \mathcal{N}(0,1)$ be sequences of iid normally distributed random variables. Then $f^{\sqrt{M}}$ as defined in (5.4.1) converges (in distribution) to a Gaussian process.*

Proof. Remember Definition 12. We need to fix $(x_1,\dots,x_N)\in\mathbb{R}^{dN}$ and show that the distribution of $(f^{\sqrt{M}}(x_1,\theta),\dots,f^{\sqrt{M}}(x_N,\theta))$ converges to a Gaussian as $M\to\infty$. For this, notice that the (vector-valued) random variables

$$(b_i\sigma(a_i\cdot x_1),\dots,b_i\sigma(a_i\cdot x_N))^\top \in \mathbb{R}^N, \quad i=1,\dots,M, \tag{5.4.2}$$

are iid, and hence indeed by the central limit theorem, their rescaled sum (rescaled as in (5.4.1) converges to an N-dimensional Gaussian. □

Bonus question: What are the mean and covariance functions associated to the limiting Gaussian process?

Next, we go back to the abstract picture and study the training dynamics subject to rescalings (this is again general, and goes beyond the neural network setting). Recall that the training dynamics can be viewed as a coupled system of equations, comprised by (5.1.5), governing the evolution of θ, and (5.2.3), governing of the evolution of f. The second depends on the solution of the first, and is (of course) in some sense redundant, because knowing θ_t, we can directly compute $f(\cdot;\theta_t)$. However, the whole point of the analysis in this section is to decouple these two equations.

We introduce the following new variables:

$$f^\alpha(x;\theta) := \alpha f(x;\theta) \qquad \text{(rescaled parameterisation)} \tag{5.4.3a}$$

$$\tau := \alpha^{-2}t \qquad \text{(short times)} \tag{5.4.3b}$$

In terms of those, the coupled system (5.1.5),(5.2.3) takes the form

$$\partial_\tau\theta_\tau = -\frac{1}{\alpha}\nabla_{\theta_\tau}\mathcal{L}(f^\alpha(\cdot,\theta_\tau)), \tag{5.4.4a}$$

$$\partial_\tau f^\alpha_\tau(\cdot) = -\frac{2}{N}\sum_{i=1}^{N} H_{\theta_\tau}(\cdot,x_i)\left(f^\alpha_\tau(x_i)-y_i\right), \tag{5.4.4b}$$

that is, the second equation is invariant under the rescaling (5.4.3), while the first one picks up a factor of $\frac{1}{\alpha}$. Therefore, indeed, in the limit when $\alpha\to\infty$, it is reasonable to expect that θ_τ remains constant (hence H_θ remains constant), while f_τ evolves according to the kernel machine dynamics (5.3.2). This argument essentially explains the phenomenon of *lazy training*, and more rigorous results can be found in [12, Appendix H] as well as in the original paper [9].

Remark 5.4.1 (Initial conditions)**.** *Notice that while the dynamics* (5.4.4b) *is invariant under the rescaling, the initial condition (suppressed in the notation/discussion) is not! Therefore, this scaling argument is not as straightforward as it appears and depends more delicately on the parameterisation. However, we established in Lemma 11 that in the case of neural networks of the form* (5.1.1)*, the limiting object is a Gaussian process (this will be the initial condition for the dynamics* (5.4.4b)*). Therefore,* f^α *remains bounded in a suitable sense as* $\alpha \to \infty$*, and therefore the argument for lazy training can be substantiated.*

Remark 5.4.2 (Deterministic limit of the NTK)**.** *In the setting of Lemma 11, the initial condition for* (5.4.4b) *will be random (a Gaussian process), see the previous remark. However, the NTK will converge towards a deterministic positive definite kernel as* $\alpha \to \infty$*: this is (roughly speaking) because* $\nabla_\theta f(\cdot;\theta) \approx \mathcal{O}(\sqrt{M})$*, and so the NTK as defined in* (5.2.4) *scales with* $\frac{1}{M}$*, leading to a 'law of large number' scaling and a deterministic limit.*

Further reading. The neural tangent kernel analysis answers some questions: Minimisers of the loss can be found according to (5.3.2), and their statistical properties can be understood as minimum-norm interpolators. However, there are also new questions: Empirically, it has been observed that neural networks often outperform kernel methods. It must therefore be the case that the NTK limiting regime is not completely realised in practice (and this fact should somehow help the performance). One intuitive idea is that the NTK (as in (5.2.3)) evolves during the training in a favourable way, and so neural networks can (maybe) be understood as kernel machine with a positive definite kernel that is adapted according to the task at hand. Understanding this in more detail is very much ongoing research. Here is a (very incomplete) list of suggestions for further reading.

1. the original paper: [9]
2. an excellent overview: [12, Appendix H]
3. Clearly, it is important to study properties of the RKHS associated to the NTK. See [3], "Deep neural tangent kernel and Laplace kernel have the same RKHS"
4. [8]: When do neural networks outperform kernel methods? The name of the paper speaks for itself...
5. [6] data adaptive kernels...

Bibliography

[1] Andreas Anastasiou, Alessandro Barp, François-Xavier Briol, Bruno Ebner, Robert E Gaunt, Fatemeh Ghaderinezhad, Jackson Gorham, Arthur Gretton, Christophe Ley, Qiang Liu, et al. Stein's Method Meets Computational Statistics: A Review of Some Recent Developments. *arXiv preprint arXiv:2105.03481*, 2021.

[2] Francis Bach. Learning theory from first principles. *Draft of a book, version of Sept*, 6:2021, 2021.

[3] Lin Chen and Sheng Xu. Deep neural tangent kernel and laplace kernel have the same rkhs. *arXiv preprint arXiv:2009.10683*, 2020.

[4] Kacper Chwialkowski, Heiko Strathmann, and Arthur Gretton. A kernel test of goodness of fit. In *International conference on machine learning*, pages 2606–2615. PMLR, 2016.

[5] Felipe Cucker and Steve Smale. On the mathematical foundations of learning. *Bulletin of the American mathematical society*, 39(1):1–49, 2002.

[6] Xialiang Dou and Tengyuan Liang. Training neural networks as learning data-adaptive kernels: Provable representation and approximation benefits. *Journal of the American Statistical Association*, 116(535):1507–1520, 2021.

[7] Gregory E Fasshauer and Qi Ye. Reproducing kernels of generalized Sobolev spaces via a Green function approach with distributional operators. *Numerische Mathematik*, 119(3):585–611, 2011.

[8] Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. When do neural networks outperform kernel methods? *Advances in Neural Information Processing Systems*, 33: 14820–14830, 2020.

[9] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[10] 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.

[11] Erwin Kreyszig. *Introductory functional analysis with applications*, volume 1. Wiley New York, 1978.

[12] Song Mei, Theodor Misiakiewicz, and Andrea Montanari. Mean-field theory of two-layers neural networks: dimension-free bounds and kernel limit. In *Conference on Learning Theory*, pages 2388–2464. PMLR, 2019.

[13] Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, Bernhard Schölkopf, et al. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141, 2017.

[14] Alfred Müller. Integral probability metrics and their generating classes of functions. *Advances in Applied Probability*, 29(2):429–443, 1997.

[15] Chris J Oates, Mark Girolami, and Nicolas Chopin. Control functionals for monte carlo integration. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, pages 695–718, 2017.

[16] Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5): 2295–2317, 2011.

[17] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175, 2015.

[18] Bharath K Sriperumbudur, Kenji Fukumizu, and Gert RG Lanckriet. Universality, Characteristic Kernels and RKHS Embedding of Measures. *Journal of Machine Learning Research*, 12(7), 2011.

[19] Ingo Steinwart and Andreas Christmann. *Support vector machines*. Springer Science & Business Media, 2008.

[20] Grace Wahba and Yuedong Wang. *Representer Theorem*, pages 1–11. American Cancer Society, 2019. ISBN 9781118445112.