2 Kernels

In Chapter 1, we described how a kernel arises as a similarity measure that can be thought of as a dot product in a so-called feature space. We tried to provide an intuitive understanding of kernels by introducing them as similarity measures, rather than immediately delving into the functional analytic theory of the classes of kernels that actually admit a dot product representation in a feature space.

In the present chapter, we will be both more formal and more precise. We will study the class of kernels k that correspond to dot products in feature spaces $\mathcal{H}$ via a map $\boldsymbol{\Phi}$,

$$\begin{aligned}\boldsymbol{\Phi} &: \mathcal{X} \to \mathcal{H} \\ &x \mapsto \mathbf{x} := \boldsymbol{\Phi}(x),\end{aligned} \tag{2.1}$$

that is,

$$k(x, x') = \langle \boldsymbol{\Phi}(x), \boldsymbol{\Phi}(x') \rangle . \tag{2.2}$$

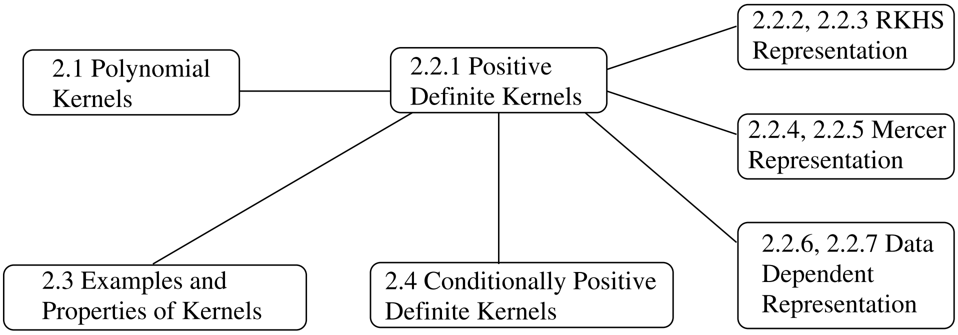
Regarding the input domain $\mathcal{X}$, we need not make assumptions other than it being a set. For instance, we could consider a set of discrete objects, such as strings.

Overview

A natural question to ask at this point is what kind of functions $k(x, x')$ admit a representation of the form (2.2); that is, whether we can always construct a dot product space $\mathcal{H}$ and a map $\boldsymbol{\Phi}$ mapping into it such that (2.2) holds true. We shall begin, however, by trying to give some motivation as to why kernels are at all useful, considering kernels that compute dot products in spaces of monomial features (Section 2.1). Following this, we move on to the questions of how, given a kernel, an associated feature space can be constructed (Section 2.2). This leads to the notion of a Reproducing Kernel Hilbert Space, crucial for the theory of kernel machines. In Section 2.3, we give some examples and properties of kernels, and in Section 2.4, we discuss a class of kernels that can be used as dissimilarity measures rather than as similarity measures.

Prerequisites

The chapter builds on knowledge of linear algebra, as briefly summarized in Appendix B. Apart from that, it can be read on its own; however, readers new to the field will profit from first reading Sections 1.1 and 1.2.


2.1 Polynomial Kernels
2.2.1 Positive Definite Kernels
2.2.2, 2.2.3 RKHS Representation
2.2.4, 2.2.5 Mercer Representation
2.2.6, 2.2.7 Data Dependent Representation
2.3 Examples and Properties of Kernels
2.4 Conditionally Positive Definite Kernels

2.1 Product Features

In this section, we think of $\mathcal{X}$ as a subset of the vector space $\mathbb{R}^N$, ($N \in \mathbb{N}$), endowed with the canonical dot product (1.3).

Monomial Features

Suppose we are given patterns $x \in \mathcal{X}$ where most information is contained in the dth order products (so-called monomials) of entries $[x]_j$ of x,

$$[x]_{j_1} \cdot [x]_{j_2} \cdots [x]_{j_d}, \tag{2.3}$$

where $j_1, \ldots, j_d \in \{1, \ldots, N\}$. Often, these monomials are referred to as *product features*. These features form the basis of many practical algorithms; indeed, there is a whole field of pattern recognition research studying *polynomial classifiers* [484], which is based on first extracting product features and then applying learning algorithms to these features. In other words, the patterns are preprocessed by mapping into the feature space $\mathcal{H}$ of all products of d entries. This has proven quite effective in visual pattern recognition tasks, for instance. To understand the rationale for doing this, note that visual patterns are usually represented as vectors whose entries are the pixel intensities. Taking products of entries of these vectors then corresponds to taking products of pixel intensities, and is thus akin to taking logical "and" operations on the pixels. Roughly speaking, this corresponds to the intuition that, for instance, a handwritten "8" constitutes an eight if there is a top circle *and* a bottom circle. With just one of the two circles, it is not half an "8," but rather a "0." Nonlinearities of this type are crucial for achieving high accuracies in pattern recognition tasks.

Let us take a look at this feature map in the simple example of two-dimensional patterns, for which $\mathcal{X} = \mathbb{R}^2$. In this case, we can collect all monomial feature extractors of degree 2 in the nonlinear map

$$\Phi : \mathbb{R}^2 \to \mathcal{H} = \mathbb{R}^3, \tag{2.4}$$

$$([x]_1, [x]_2) \mapsto ([x]_1^2, [x]_2^2, [x]_1[x]_2). \tag{2.5}$$

This approach works fine for small toy examples, but it fails for realistically sized problems: for N-dimensional input patterns, there exist

$$N_{\mathcal{H}} = \binom{d+N-1}{d} = \frac{(d+N-1)!}{d!(N-1)!} \tag{2.6}$$

different monomials (2.3) of degree d, comprising a feature space $\mathcal{H}$ of dimension $N_{\mathcal{H}}$. For instance, 16×16 pixel input images and a monomial degree $d = 5$ thus yield a dimension of almost 10^{10}.

In certain cases described below, however, there exists a way of *computing dot products* in these high-dimensional feature spaces without explicitly mapping into the spaces, by means of kernels nonlinear in the input space $\mathbb{R}^N$. Thus, if the subsequent processing can be carried out using dot products exclusively, we are able to deal with the high dimension.

We now describe how dot products in polynomial feature spaces can be computed efficiently, followed by a section in which we discuss more general feature spaces. In order to compute dot products of the form $\langle \Phi(x), \Phi(x') \rangle$, we employ kernel representations of the form

Kernel

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle , \tag{2.7}$$

which allow us to compute the value of the dot product in $\mathcal{H}$ without having to explicitly compute the map Φ.

What does k look like in the case of polynomial features? We start by giving an example for $N = d = 2$, as considered above [561]. For the map

$$\Phi : ([x]_1, [x]_2) \mapsto ([x]_1^2, [x]_2^2, [x]_1[x]_2, [x]_2[x]_1), \tag{2.8}$$

(note that for now, we have considered $[x]_1[x]_2$ and $[x]_2[x]_1$ as separate features; thus we are looking at *ordered* monomials) dot products in $\mathcal{H}$ take the form

$$\langle \Phi(x), \Phi(x') \rangle = [x]_1^2[x']_1^2 + [x]_2^2[x']_2^2 + 2[x]_1[x]_2[x']_1[x']_2 = \langle x, x' \rangle^2 . \tag{2.9}$$

In other words, the desired kernel k is simply the square of the dot product in input space. The same works for arbitrary $N, d \in \mathbb{N}$ [62]: as a straightforward generalization of a result proved in the context of polynomial approximation [412, Lemma 2.1], we have:

Proposition 2.1 *Define C_d to map $x \in \mathbb{R}^N$ to the vector $C_d(x)$ whose entries are all possible dth degree ordered products of the entries of x. Then the corresponding kernel computing the dot product of vectors mapped by C_d is*

$$k(x, x') = \langle C_d(x), C_d(x') \rangle = \langle x, x' \rangle^d . \tag{2.10}$$

Polynomial Kernel

Proof We directly compute

$$\langle C_d(x), C_d(x') \rangle = \sum_{j_1=1}^{N} \dots \sum_{j_d=1}^{N} [x]_{j_1} \cdot \dots \cdot [x]_{j_d} \cdot [x']_{j_1} \cdot \dots \cdot [x']_{j_d} \tag{2.11}$$

$$= \sum_{j_1=1}^{N} [x]_{j_1} \cdot [x']_{j_1} \dots \sum_{j_d=1}^{N} [x]_{j_d} \cdot [x']_{j_d} = \left(\sum_{j=1}^{N} [x]_j \cdot [x']_j \right)^d = \langle x, x' \rangle^d . \qquad \blacksquare$$

Note that we used the symbol C_d for the feature map. The reason for this is that we would like to reserve Φ_d for the corresponding map computing *unordered* product features. Let us construct such a map Φ_d, yielding the same value of the dot product. To this end, we have to compensate for the multiple occurrence of certain monomials in C_d by scaling the respective entries of Φ_d with the square roots of their numbers of occurrence. Then, by this construction of Φ_d, and (2.10),

$$\langle \Phi_d(x), \Phi_d(x') \rangle = \langle C_d(x), C_d(x') \rangle = \langle x, x' \rangle^d . \tag{2.12}$$

For instance, if n of the j_i in (2.3) are equal, and the remaining ones are different, then the coefficient in the corresponding component of Φ_d is $\sqrt{(d-n+1)!}$. For the general case, see Problem 2.2. For Φ_2, this simply means that [561]

$$\Phi_2(x) = ([x]_1^2, [x]_2^2, \sqrt{2}\,[x]_1[x]_2). \tag{2.13}$$

The above reasoning illustrates an important point pertaining to the construction of feature spaces associated with kernel functions. Although they map into different feature spaces, Φ_d and C_d are both valid instantiations of feature maps for $k(x, x') = \langle x, x' \rangle^d$.

To illustrate how monomial feature kernels can significantly simplify pattern recognition tasks, let us consider a simple toy example.

Toy Example

Example 2.2 (Monomial Features in 2-D Pattern Recognition) *In the example of Figure 2.1, a non-separable problem is reduced to the construction of a separating hyperplane by preprocessing the input data with Φ_2. As we shall see in later chapters, this has advantages both from the* computational *point of view (there exist efficient algorithms for computing the hyperplane) and from the* statistical *point of view (there exist guarantees for how well the hyperplane will generalize to unseen test points).*

In more realistic cases, e.g., if x represents an image with the entries being pixel values, polynomial kernels $\langle x, x' \rangle^d$ enable us to work in the space spanned by products of any d pixel values — provided that we are able to do our work solely in terms of dot products, without any explicit usage of a mapped pattern $\Phi_d(x)$. Using kernels of the form (2.10), we can take higher-order statistics into account, without the combinatorial explosion (2.6) of time and memory complexity which accompanies even moderately high N and d.

To conclude this section, note that it is possible to modify (2.10) such that it maps into the space of all monomials *up to* degree d, by defining $k(x, x') = (\langle x, x' \rangle + 1)^d$ (Problem 2.17). Moreover, in practice, it is often useful to multiply the kernel by a scaling factor c to ensure that its numeric range is within some bounded interval, say $[-1, 1]$. The value of c will depend on the dimension and range of the data.

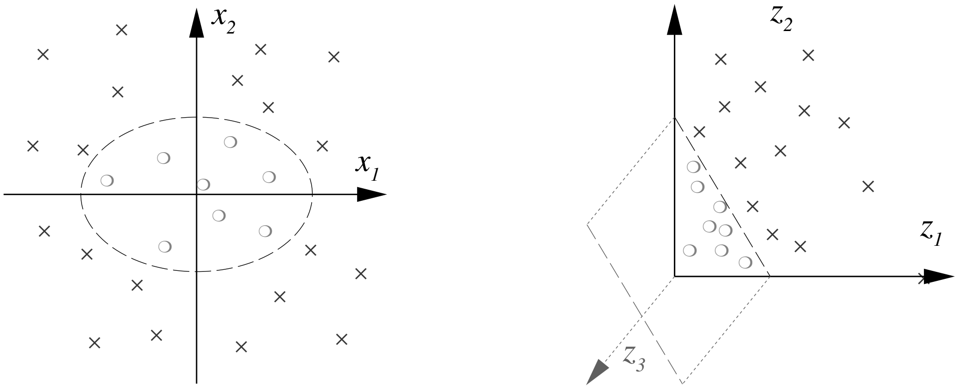


Figure 2.1 Toy example of a binary classification problem mapped into feature space. We assume that the true decision boundary is an ellipse in input space (left panel). The task of the learning process is to estimate this boundary based on empirical data consisting of training points in both classes (crosses and circles, respectively). When mapped into feature space via the nonlinear map $\Phi_2(x) = (z_1, z_2, z_3) = ([x]_1^2, [x]_2^2, \sqrt{2}\,[x]_1[x]_2)$ (right panel), the ellipse becomes a hyperplane (in the present simple case, it is parallel to the z_3 axis, hence all points are plotted in the (z_1, z_2) plane). This is due to the fact that ellipses can be written as linear equations in the entries of (z_1, z_2, z_3). Therefore, in feature space, the problem reduces to that of estimating a hyperplane from the mapped data points. Note that via the polynomial kernel (see (2.12) and (2.13)), the dot product in the three-dimensional space can be computed without computing Φ_2. Later in the book, we shall describe algorithms for constructing hyperplanes which are based on dot products (Chapter 7).

2.2 The Representation of Similarities in Linear Spaces

In what follows, we will look at things the other way round, and start with the kernel rather than with the feature map. Given some kernel, can we construct a feature space such that the kernel computes the dot product in that feature space; that is, such that (2.2) holds? This question has been brought to the attention of the machine learning community in a variety of contexts, especially during recent years [4, 152, 62, 561, 480]. In functional analysis, the same problem has been studied under the heading of *Hilbert space representations* of kernels. A good monograph on the theory of kernels is the book of Berg, Christensen, and Ressel [42]; indeed, a large part of the material in the present chapter is based on this work. We do not aim to be fully rigorous; instead, we try to provide insight into the basic ideas. As a rule, all the results that we state without proof can be found in [42]. Other standard references include [16, 455].

There is one more aspect in which this section differs from the previous one: the latter dealt with vectorial data, and the domain $\mathcal{X}$ was assumed to be a subset of $\mathbb{R}^N$. By contrast, the results in the current section hold for data drawn from domains which need no structure, other than their being nonempty sets. This generalizes kernel learning algorithms to a large number of situations where a vectorial representation is not readily available, and where one directly works with pairwise distances or similarities between non-vectorial objects [246, 467, 154, 210, 234, 585]. This theme will recur in several places throughout the book, for instance in Chapter 13.

2.2.1 Positive Definite Kernels

We start with some basic definitions and results. As in the previous chapter, indices i and j are understood to run over $1, \ldots, m$.

Gram Matrix

Definition 2.3 (Gram Matrix) *Given a function $k : \mathcal{X}^2 \to \mathbb{K}$ (where $\mathbb{K} = \mathbb{C}$ or $\mathbb{K} = \mathbb{R}$) and patterns $x_1, \ldots, x_m \in \mathcal{X}$, the $m \times m$ matrix K with elements*

$$K_{ij} := k(x_i, x_j) \tag{2.14}$$

is called the Gram matrix *(or* kernel matrix*) of k with respect to $x_1, \ldots, x_m$.*

PD Matrix

Definition 2.4 (Positive Definite Matrix) *A complex $m \times m$ matrix K satisfying*

$$\sum_{i,j} c_i \bar{c}_j K_{ij} \geq 0 \tag{2.15}$$

for all $c_i \in \mathbb{C}$ is called positive definite.[1] *Similarly, a real symmetric $m \times m$ matrix K satisfying (2.15) for all $c_i \in \mathbb{R}$ is called* positive definite.

Note that a symmetric matrix is positive definite if and only if all its eigenvalues are nonnegative (Problem 2.4). The left hand side of (2.15) is often referred to as the *quadratic form* induced by K.

PD Kernel

Definition 2.5 ((Positive Definite) Kernel) *Let $\mathcal{X}$ be a nonempty set. A function k on $\mathcal{X} \times \mathcal{X}$ which for all $m \in \mathbb{N}$ and all $x_1 \ldots, x_m \in \mathcal{X}$ gives rise to a positive definite Gram matrix is called a* positive definite (pd) kernel*. Often, we shall refer to it simply as a* kernel.

Remark 2.6 (Terminology) *The term* kernel *stems from the first use of this type of function in the field of integral operators as studied by Hilbert and others [243, 359, 112]. A function k which gives rise to an operator T_k via*

$$(T_k f)(x) = \int_{\mathcal{X}} k(x, x') f(x') \, dx' \tag{2.16}$$

is called the kernel of T_k.

In the literature, a number of different terms are used for positive definite kernels, such as reproducing kernel, Mercer kernel, admissible kernel, Support Vector kernel, nonnegative definite kernel, *and* covariance function*. One might argue that the term* positive definite kernel *is slightly misleading. In matrix theory, the term* definite *is sometimes reserved for the case where equality in (2.15) only occurs if $c_1 = \ldots = c_m = 0$.*

[1] The bar in $\bar{c}_j$ denotes complex conjugation; for real numbers, it has no effect.

Simply using the term positive kernel, *on the other hand, could be mistaken as referring to a kernel whose* values *are positive. Finally, the term positive* semi*definite kernel becomes rather cumbersome if it is to be used throughout a book. Therefore, we follow the convention used for instance in [42], and employ the term* positive definite *both for kernels and matrices in the way introduced above. The case where the value* 0 *is only attained if all coefficients are* 0 *will be referred to as* strictly *positive definite.*

We shall mostly use the term kernel. *Whenever we want to refer to a kernel $k(x, x')$ which is not positive definite in the sense stated above, it will be clear from the context.*

The definitions for positive definite kernels and positive definite matrices differ in the fact that in the former case, we are free to choose the points on which the kernel is evaluated — for every choice, the kernel induces a positive definite matrix.

Positive definiteness implies *positivity on the diagonal* (Problem 2.12),

$$k(x, x) \geq 0 \text{ for all } x \in \mathcal{X}, \tag{2.17}$$

and *symmetry* (Problem 2.13),

$$k(x_i, x_j) = \overline{k(x_j, x_i)}. \tag{2.18}$$

To also cover the complex-valued case, our definition of symmetry includes complex conjugation. The definition of symmetry of matrices is analogous; that is, $K_{ij} = \overline{K}_{ji}$.

Real-Valued Kernels

For real-valued kernels it is not sufficient to stipulate that (2.15) hold for real coefficients c_i. To get away with real coefficients only, we must additionally require that the kernel be symmetric (Problem 2.14); $k(x_i, x_j) = k(x_j, x_i)$ (cf. Problem 2.13).

It can be shown that whenever k is a (complex-valued) positive definite kernel, its real part is a (real-valued) positive definite kernel. Below, we shall largely be dealing with real-valued kernels. Most of the results, however, also apply for complex-valued kernels.

Kernels can be regarded as generalized dot products. Indeed, any dot product is a kernel (Problem 2.5); however, linearity in the arguments, which is a standard property of dot products, does not carry over to general kernels. However, another property of dot products, the Cauchy-Schwarz inequality, does have a natural generalization to kernels:

Proposition 2.7 (Cauchy-Schwarz Inequality for Kernels) *If k is a positive definite kernel, and $x_1, x_2 \in \mathcal{X}$, then*

$$|k(x_1, x_2)|^2 \leq k(x_1, x_1) \cdot k(x_2, x_2). \tag{2.19}$$

Proof For sake of brevity, we give a non-elementary proof using some basic facts of linear algebra. The 2×2 Gram matrix with entries $K_{ij} = k(x_i, x_j)$ $(i, j \in \{1, 2\})$ is positive definite. Hence both its eigenvalues are nonnegative, and so is their product, the determinant of K. Therefore

$$0 \leq K_{11}K_{22} - K_{12}K_{21} = K_{11}K_{22} - K_{12}\overline{K}_{12} = K_{11}K_{22} - |K_{12}|^2. \tag{2.20}$$

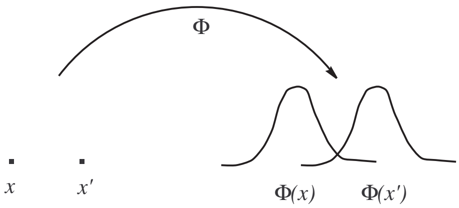


Figure 2.2 One instantiation of the feature map associated with a kernel is the map (2.21), which represents each pattern (in the picture, x or x') by a kernel-shaped *function* sitting on the pattern. In this sense, each pattern is represented by its similarity to *all* other patterns. In the picture, the kernel is assumed to be bell-shaped, e.g., a Gaussian $k(x,x') = \exp(-\|x-x'\|^2/(2\,\sigma^2))$. In the text, we describe the construction of a dot product $\langle .,. \rangle$ on the function space such that $k(x,x') = \langle \Phi(x), \Phi(x') \rangle$.

Substituting $k(x_i, x_j)$ for K_{ij}, we get the desired inequality. ■

We now show how the feature spaces in question are defined by the choice of kernel function.

2.2.2 The Reproducing Kernel Map

Assume that k is a real-valued positive definite kernel, and $\mathcal{X}$ a nonempty set. We define a map from $\mathcal{X}$ into the space of functions mapping $\mathcal{X}$ into $\mathbb{R}$, denoted as $\mathbb{R}^{\mathcal{X}} := \{f : \mathcal{X} \to \mathbb{R}\}$, via

Feature Map

$$\begin{aligned} \Phi : \mathcal{X} &\to \mathbb{R}^{\mathcal{X}} \\ x &\mapsto k(.,x). \end{aligned} \tag{2.21}$$

Here, $\Phi(x)$ denotes the function that assigns the value $k(x',x)$ to $x' \in \mathcal{X}$, i.e., $\Phi(x)(.) = k(.,x)$ (as shown in Figure 2.2).

We have thus turned each pattern into a function on the domain $\mathcal{X}$. In this sense, a pattern is now represented by its similarity to *all* other points in the input domain $\mathcal{X}$. This seems a very rich representation; nevertheless, it will turn out that the kernel allows the computation of the dot product in this representation. Below, we show how to construct a feature space associated with Φ, proceeding in the following steps:

1. Turn the image of Φ into a vector space,
2. define a dot product; that is, a strictly positive definite bilinear form, and
3. show that the dot product satisfies $k(x,x') = \langle \Phi(x), \Phi(x') \rangle$.

We begin by constructing a dot product space containing the images of the input patterns under Φ. To this end, we first need to define a vector space. This is done by taking linear combinations of the form

Vector Space

$$f(.) = \sum_{i=1}^{m} \alpha_i k(.,x_i). \tag{2.22}$$

Here, $m \in \mathbb{N}$, $\alpha_i \in \mathbb{R}$ and $x_1, \ldots, x_m \in \mathcal{X}$ are arbitrary. Next, we define a dot product between f and another function

$$g(.) = \sum_{j=1}^{m'} \beta_j k(., x'_j), \tag{2.23}$$

Dot Product

where $m' \in \mathbb{N}$, $\beta_j \in \mathbb{R}$ and $x'_1, \ldots, x'_{m'} \in \mathcal{X}$, as

$$\langle f, g \rangle := \sum_{i=1}^{m} \sum_{j=1}^{m'} \alpha_i \beta_j k(x_i, x'_j). \tag{2.24}$$

This expression explicitly contains the expansion coefficients, which need not be unique. To see that it is nevertheless well-defined, note that

$$\langle f, g \rangle = \sum_{j=1}^{m'} \beta_j f(x'_j), \tag{2.25}$$

using $k(x'_j, x_i) = k(x_i, x'_j)$. The sum in (2.25), however, does not depend on the particular expansion of f. Similarly, for g, note that

$$\langle f, g \rangle = \sum_{i=1}^{m} \alpha_i g(x_i). \tag{2.26}$$

The last two equations also show that $\langle \cdot, \cdot \rangle$ is bilinear. It is symmetric, as $\langle f, g \rangle = \langle g, f \rangle$. Moreover, it is positive definite, since positive definiteness of k implies that for any function f, written as (2.22), we have

$$\langle f, f \rangle = \sum_{i,j=1}^{m} \alpha_i \alpha_j k(x_i, x_j) \geq 0. \tag{2.27}$$

The latter implies that $\langle \cdot, \cdot \rangle$ is actually itself a positive definite kernel, defined on our space of functions. To see this, note that given functions $f_1, \ldots, f_n$, and coefficients $\gamma_1, \ldots, \gamma_n \in \mathbb{R}$, we have

$$\sum_{i,j=1}^{n} \gamma_i \gamma_j \langle f_i, f_j \rangle = \left\langle \sum_{i=1}^{n} \gamma_i f_i, \sum_{j=1}^{n} \gamma_j f_j \right\rangle \geq 0. \tag{2.28}$$

Here, the left hand equality follows from the bilinearity of $\langle \cdot, \cdot \rangle$, and the right hand inequality from (2.27). For the last step in proving that it qualifies as a dot product, we will use the following interesting property of Φ, which follows directly from the definition: for all functions (2.22), we have

$$\langle k(., x), f \rangle = f(x) \tag{2.29}$$

— k is the *representer of evaluation*. In particular,

$$\langle k(., x), k(., x') \rangle = k(x, x'). \tag{2.30}$$

Reproducing Kernel

By virtue of these properties, positive definite kernels k are also called *reproducing kernels* [16, 42, 455, 578, 467, 202]. By (2.29) and Proposition 2.7, we have

$$|f(x)|^2 = |\langle k(., x), f \rangle|^2 \leq k(x, x) \cdot \langle f, f \rangle. \tag{2.31}$$

Therefore, $\langle f, f \rangle = 0$ directly implies $f = 0$, which is the last property that required proof in order to establish that $\langle \cdot, \cdot \rangle$ is a dot product (cf. Section B.2).

The case of complex-valued kernels can be dealt with using the same construction; in that case, we will end up with a complex dot product space [42].

The above reasoning has shown that any positive definite kernel can be thought of as a dot product in another space: in view of (2.21), the reproducing kernel property (2.30) amounts to

$$\langle \Phi(x), \Phi(x') \rangle = k(x, x'). \tag{2.32}$$

Therefore, the dot product space $\mathcal{H}$ constructed in this way is one possible instantiation of the feature space associated with a kernel.

Kernels from Feature Maps

Above, we have started with the kernel, and constructed a feature map. Let us now consider the opposite direction. Whenever we have a mapping Φ from $\mathcal{X}$ into a dot product space, we obtain a positive definite kernel via $k(x, x') := \langle \Phi(x), \Phi(x') \rangle$. This can be seen by noting that for all $c_i \in \mathbb{R}, x_i \in \mathcal{X}, i = 1, \ldots, m$, we have

$$\sum_{i,j} c_i c_j k(x_i, x_j) = \left\langle \sum_i c_i \Phi(x_i), \sum_j c_j \Phi(x_j) \right\rangle = \left\| \sum_i c_i \Phi(x_i) \right\|^2 \geq 0, \tag{2.33}$$

due to the nonnegativity of the norm.

Equivalent Definition of PD Kernels

This has two consequences. First, it allows us to give an equivalent definition of positive definite kernels as functions with the property that there exists a map Φ into a dot product space such that (2.32) holds true. Second, it allows us to construct kernels from feature maps. For instance, it is in this way that powerful linear representations of 3D heads proposed in computer graphics [575, 59] give rise to kernels. The identity (2.32) forms the basis for the kernel trick:

Kernel Trick

Remark 2.8 ("Kernel Trick") *Given an algorithm which is formulated in terms of a positive definite kernel k, one can construct an alternative algorithm by replacing k by another positive definite kernel $\tilde{k}$.*

In view of the material in the present section, the justification for this procedure is the following: effectively, the original algorithm can be thought of as a dot product based algorithm operating on vectorial data $\Phi(x_1), \ldots, \Phi(x_m)$. The algorithm obtained by replacing k by $\tilde{k}$ then is exactly the same dot product based algorithm, only that it operates on $\tilde{\Phi}(x_1), \ldots, \tilde{\Phi}(x_m)$.

The best known application of the kernel trick is in the case where k is the dot product in the input domain (cf. Problem 2.5). The trick is not limited to that case, however: k and $\tilde{k}$ can *both* be nonlinear kernels. In general, care must be exercised in determining whether the resulting algorithm will be useful: sometimes, an algorithm will only work subject to additional conditions on the input data, e.g., the data set might have to lie in the positive orthant. We shall later see that certain kernels induce feature maps which enforce such properties for the mapped data (cf. (2.73)), and that there are algorithms which take advantage of these aspects (e.g., in Chapter 8). In such cases, not every conceivable positive definite kernel will make sense.

Historical Remarks

Even though the kernel trick had been used in the literature for a fair amount of time [4, 62], it took until the mid 1990s before it was explicitly stated that *any* algorithm that only depends on dot products, i.e., any algorithm that is rotationally invariant, can be kernelized [479, 480]. Since then, a number of algorithms have benefitted from the kernel trick, such as the ones described in the present book, as well as methods for clustering in feature spaces [479, 215, 199].

Moreover, the machine learning community took time to comprehend that the definition of kernels on general sets (rather than dot product spaces) greatly extends the applicability of kernel methods [467], to data types such as texts and other sequences [234, 585, 23]. Indeed, this is now recognized as a crucial feature of kernels: they lead to an embedding of general data types in linear spaces.

Not surprisingly, the history of methods for representing kernels in linear spaces (in other words, the mathematical counterpart of the kernel trick) dates back significantly further than their use in machine learning. The methods appear to have first been studied in the 1940s by Kolmogorov [304] for countable $\mathcal{X}$ and Aronszajn [16] in the general case. Pioneering work on linear representations of a related class of kernels, to be described in Section 2.4, was done by Schoenberg [465]. Further bibliographical comments can be found in [42].

We thus see that the mathematical basis for kernel algorithms has been around for a long time. As is often the case, however, the practical importance of mathematical results was initially underestimated.[2]

2.2.3 Reproducing Kernel Hilbert Spaces

In the last section, we described how to define a space of functions which is a valid realization of the feature spaces associated with a given kernel. To do this, we had to make sure that the space is a vector space, and that it is endowed with a dot product. Such spaces are referred to as dot product spaces (cf. Appendix B), or equivalently as *pre-Hilbert* spaces. The reason for the latter is that one can turn them into Hilbert spaces (cf. Section B.3) by a fairly simple mathematical trick. This additional structure has some mathematical advantages. For instance, in Hilbert spaces it is always possible to define projections. Indeed, Hilbert spaces are one of the favorite concepts of functional analysis.

So let us again consider the pre-Hilbert space of functions (2.22), endowed with the dot product (2.24). To turn it into a Hilbert space (over $\mathbb{R}$), one *completes* it in the norm corresponding to the dot product, $\|f\| := \sqrt{\langle f, f \rangle}$. This is done by adding the limit points of sequences that are convergent in that norm (see Appendix B).

[2] This is illustrated by the following quotation from an excellent machine learning textbook published in the seventies (p. 174 in [152]): *"The familiar functions of mathematical physics are eigenfunctions of symmetric kernels, and their use is often suggested for the construction of potential functions. However, these suggestions are more appealing for their mathematical beauty than their practical usefulness."*

RKHS

In view of the properties (2.29) and (2.30), this space is usually called a *reproducing kernel Hilbert space (RKHS).*

In general, an RKHS can be defined as follows.

Definition 2.9 (Reproducing Kernel Hilbert Space) *Let $\mathcal{X}$ be a nonempty set (often called the index set) and by $\mathcal{H}$ a Hilbert space of functions $f : \mathcal{X} \to \mathbb{R}$. Then $\mathcal{H}$ is called a reproducing kernel Hilbert space endowed with the dot product $\langle \cdot, \cdot \rangle$ (and the norm $\|f\| := \sqrt{\langle f, f \rangle}$) if there exists a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with the following properties.*

Reproducing Property

1. *k has the reproducing property*[3]

$$\langle f, k(x, \cdot) \rangle = f(x) \text{ for all } f \in \mathcal{H}; \tag{2.34}$$

in particular,

$$\langle k(x, \cdot), k(x', \cdot) \rangle = k(x, x'). \tag{2.35}$$

Closed Space

2. *k spans $\mathcal{H}$, i.e. $\mathcal{H} = \overline{\operatorname{span}\{k(x, \cdot) | x \in \mathcal{X}\}}$ where $\overline{X}$ denotes the completion of the set X (cf. Appendix B).*

On a more abstract level, an RKHS can be defined as a Hilbert space of functions f on $\mathcal{X}$ such that all evaluation functionals (the maps $f \mapsto f(x')$, where $x' \in \mathcal{X}$) are continuous. In that case, by the Riesz representation theorem (e.g., [429]), for each $x' \in \mathcal{X}$ there exists a unique function of x, called $k(x, x')$, such that

$$f(x') = \langle f, k(., x') \rangle . \tag{2.36}$$

It follows directly from (2.35) that $k(x, x')$ is symmetric in its arguments (see Problem 2.28) and satisfies the conditions for positive definiteness.

Uniqueness of k

Note that the RKHS uniquely determines k. This can be shown by contradiction: assume that there exist two kernels, say k and k', spanning the same RKHS $\mathcal{H}$. From Problem 2.28 we know that both k and k' must be symmetric. Moreover, from (2.34) we conclude that

$$\langle k(x, \cdot), k'(x', \cdot) \rangle_{\mathcal{H}} = k(x, x') = k'(x', x). \tag{2.37}$$

In the second equality we used the symmetry of the dot product. Finally, symmetry in the arguments of k yields $k(x, x') = k'(x, x')$ which proves our claim.

2.2.4 The Mercer Kernel Map

Section 2.2.2 has shown that any positive definite kernel can be represented as a dot product in a linear space. This was done by explicitly constructing a (Hilbert) space that does the job. The present section will construct another Hilbert space.

[3] Note that this implies that each $f \in \mathcal{H}$ is actually a single function whose values at any $x \in \mathcal{X}$ are well-defined. In contrast, L_2 Hilbert spaces usually do not have this property. The elements of these spaces are equivalence classes of functions that disagree only on sets of measure 0; cf. footnote 15 in Section B.3.

One could argue that this is superfluous, given that any two separable Hilbert spaces are isometrically isomorphic, in other words, it is possible to define a one-to-one linear map between the spaces which preserves the dot product. However, the tool that we shall presently use, Mercer's theorem, has played a crucial role in the understanding of SVMs, and it provides valuable insight into the geometry of feature spaces, which more than justifies its detailed discussion. In the SVM literature, the kernel trick is usually introduced via Mercer's theorem.

Mercer's Theorem

We start by stating the version of Mercer's theorem given in [606]. We assume $(\mathcal{X}, \mu)$ to be a finite measure space.[4] The term *almost all* (cf. Appendix B) means *except for sets of measure zero*. For the commonly used Lebesgue-Borel measure, countable sets of individual points are examples of zero measure sets. Note that the integral with respect to a measure is explained in Appendix B. Readers who do not want to go into mathematical detail may simply want to think of the $d\mu(x')$ as a dx', and of $\mathcal{X}$ as a compact subset of $\mathbb{R}^N$. For further explanations of the terms involved in this theorem, cf. Appendix B, especially Section B.3.

Theorem 2.10 (Mercer [359, 307]) *Suppose $k \in L_\infty(\mathcal{X}^2)$ is a symmetric real-valued function such that the integral operator (cf. (2.16))*

$$\begin{aligned} T_k \ &: \ L_2(\mathcal{X}) \to L_2(\mathcal{X}) \\ (T_k f)(x) &:= \int_{\mathcal{X}} k(x, x') f(x') \, d\mu(x') \end{aligned} \tag{2.38}$$

is positive definite*; that is, for all $f \in L_2(\mathcal{X})$, we have*

$$\int_{\mathcal{X}^2} k(x, x') f(x) f(x') \, d\mu(x) d\mu(x') \geq 0. \tag{2.39}$$

Let $\psi_j \in L_2(\mathcal{X})$ be the normalized orthogonal eigenfunctions of T_k associated with the eigenvalues $\lambda_j > 0$, sorted in non-increasing order. Then

1. $(\lambda_j)_j \in \ell_1$,
2. $k(x, x') = \sum_{j=1}^{N_{\mathcal{H}}} \lambda_j \psi_j(x) \psi_j(x')$ *holds for almost all (x, x'). Either $N_{\mathcal{H}} \in \mathbb{N}$, or $N_{\mathcal{H}} = \infty$; in the latter case, the series converges absolutely and uniformly for almost all (x, x').*

For the converse of Theorem 2.10, see Problem 2.23. For a data-dependent approximation and its relationship to kernel PCA (Section 1.7), see Problem 2.26.

From statement 2 it follows that $k(x, x')$ corresponds to a dot product in $\ell_2^{N_{\mathcal{H}}}$, since $k(x, x') = \langle \Phi(x), \Phi(x') \rangle$ with

$$\begin{aligned} \Phi : \mathcal{X} \ &\to \ \ell_2^{N_{\mathcal{H}}} \\ x \ &\mapsto \ (\sqrt{\lambda_j} \psi_j(x))_{j=1,\ldots,N_{\mathcal{H}}}, \end{aligned} \tag{2.40}$$

for almost all $x \in \mathcal{X}$. Note that we use the same Φ as in (2.21) to denote the feature

[4] A finite measure space is a set $\mathcal{X}$ with a σ-algebra (Definition B.1) defined on it, and a measure (Definition B.2) defined on the latter, satisfying $\mu(\mathcal{X}) < \infty$ (so that, up to a scaling factor, μ is a probability measure).

map, although the target spaces are different. However, this distinction is not important for the present purposes — we are interested in the existence of some Hilbert space in which the kernel corresponds to the dot product, and not in what particular representation of it we are using.

In fact, it has been noted [467] that the *uniform* convergence of the series implies that given any $\epsilon > 0$, there exists an $n \in \mathbb{N}$ such that even if $N_{\mathcal{H}} = \infty$, k can be approximated within accuracy ϵ as a dot product in $\mathbb{R}^n$: for almost all $x, x' \in \mathcal{X}$, $|k(x,x') - \langle \Phi^n(x), \Phi^n(x') \rangle| < \epsilon$, where $\Phi^n : x \mapsto (\sqrt{\lambda_1}\psi_1(x), \ldots, \sqrt{\lambda_n}\psi_n(x))$. The feature space can thus always be thought of as finite-dimensional within some accuracy ϵ. We summarize our findings in the following proposition.

Mercer Feature Map

Proposition 2.11 (Mercer Kernel Map) *If k is a kernel satisfying the conditions of Theorem 2.10, we can construct a mapping Φ into a space where k acts as a dot product,*

$$\langle \Phi(x), \Phi(x') \rangle = k(x,x'), \tag{2.41}$$

for almost all $x, x' \in \mathcal{X}$. Moreover, given any $\epsilon > 0$, there exists a map Φ_n into an n-dimensional dot product space (where $n \in \mathbb{N}$ depends on ϵ) such that

$$|k(x,x') - \langle \Phi^n(x), \Phi^n(x') \rangle| < \epsilon \tag{2.42}$$

for almost all $x, x' \in \mathcal{X}$.

Both Mercer kernels and positive definite kernels can thus be represented as dot products in Hilbert spaces. The following proposition, showing a case where the two types of kernels coincide, thus comes as no surprise.

Proposition 2.12 (Mercer Kernels are Positive Definite [359, 42]) *Let $\mathcal{X} = [a,b]$ be a compact interval and let $k : [a,b] \times [a,b] \to \mathbb{C}$ be continuous. Then k is a positive definite kernel if and only if*

$$\int_a^b \int_a^b k(x,x') f(x) f(x')\, dx\, dx' \geq 0 \tag{2.43}$$

for each continuous function $f : \mathcal{X} \to \mathbb{C}$.

Note that the conditions in this proposition are actually more restrictive than those of Theorem 2.10. Using the feature space representation (Proposition 2.11), however, it is easy to see that Mercer kernels are also positive definite (for almost all $x, x' \in \mathcal{X}$) in the more general case of Theorem 2.10: given any $\mathbf{c} \in \mathbb{R}^m$, we have

$$\sum_{i,j} c_i c_j k(x_i, x_j) = \sum_{i,j} c_i c_j \left\langle \Phi(x_i), \Phi(x_j) \right\rangle = \left\| \sum_i c_i \Phi(x_i) \right\|^2 \geq 0. \tag{2.44}$$

Being positive definite, Mercer kernels are thus also reproducing kernels.

We next show how the reproducing kernel map is related to the Mercer kernel map constructed from the eigenfunction decomposition [202, 467]. To this end, let us consider a kernel which satisfies the condition of Theorem 2.10, and construct a dot product $\langle\cdot,\cdot\rangle$ such that k becomes a reproducing kernel for the Hilbert space $\mathcal{H}$ containing the functions

$$f(x) = \sum_{i=1}^{\infty} \alpha_i k(x, x_i) = \sum_{i=1}^{\infty} \alpha_i \sum_{j=1}^{N_{\mathcal{H}}} \lambda_j \psi_j(x) \psi_j(x_i). \tag{2.45}$$

By linearity, which holds for any dot product, we have

$$\langle f, k(., x') \rangle = \sum_{i=1}^{\infty} \alpha_i \sum_{j,n=1}^{N_{\mathcal{H}}} \lambda_j \psi_j(x_i) \langle \psi_j, \psi_n \rangle \lambda_n \psi_n(x'). \tag{2.46}$$

Since k is a Mercer kernel, the ψ_i $(i = 1, \ldots, N_{\mathcal{H}})$ can be chosen to be orthogonal with respect to the dot product in $L_2(\mathcal{X})$. Hence it is straightforward to choose $\langle\cdot,\cdot\rangle$ such that

$$\langle \psi_j, \psi_n \rangle = \delta_{jn} / \lambda_j \tag{2.47}$$

(using the Kronecker symbol δ_{jn}, see (B.30)), in which case (2.46) reduces to the reproducing kernel property (2.36) (using (2.45)). For a coordinate representation in the RKHS, see Problem 2.29.

Equivalence of Feature Spaces

The above connection between the Mercer kernel map and the RKHS map is instructive, but we shall rarely make use of it. In fact, we will usually *identify* the different feature spaces. Thus, to avoid confusion in subsequent chapters, the following comments are necessary. As described above, there are different ways of constructing feature spaces for any given kernel. In fact, they can even differ in terms of their dimensionality (cf. Problem 2.22). The two feature spaces that we will mostly use in this book are the RKHS associated with k (Section 2.2.2) and the Mercer ℓ_2 feature space. We will mostly use the same symbol $\mathcal{H}$ for all feature spaces that are associated with a given kernel. This makes sense provided that everything we do, at the end of the day, reduces to dot products. For instance, let us assume that Φ_1, Φ_2 are maps into the feature spaces $\mathcal{H}_1, \mathcal{H}_2$ respectively, both associated with the kernel k; in other words,

$$k(x, x') = \langle \Phi_i(x), \Phi_i(x') \rangle_{\mathcal{H}_i}\,, \text{ for } i = 1, 2. \tag{2.48}$$

Then it will usually *not* be the case that $\Phi_1(x) = \Phi_2(x)$; due to (2.48), however, we always have $\langle \Phi_1(x), \Phi_1(x') \rangle_{\mathcal{H}_1} = \langle \Phi_2(x), \Phi_2(x') \rangle_{\mathcal{H}_2}$. Therefore, as long as we are only interested in dot products, the two spaces can be considered identical.

An example of this identity is the so-called large margin regularizer that is usually used in SVMs, as discussed in the introductory chapter (cf. also Chapters 4 and 7),

$$\langle \mathbf{w}, \mathbf{w} \rangle\,, \text{ where } \mathbf{w} = \sum_{i=1}^{m} \alpha_i \Phi(x_i). \tag{2.49}$$

No matter whether Φ is the RKHS map $\Phi(x_i) = k(., x_i)$ (2.21) or the Mercer map $\Phi(x_i) = (\sqrt{\lambda_j} \psi_j(x))_{j=1,\ldots,N_{\mathcal{H}}}$ (2.40), the value of $\|\mathbf{w}\|^2$ will not change.

This point is of great importance, and we hope that all readers are still with us.

It is fair to say, however, that Section 2.2.5 can be skipped at first reading.

2.2.5 The Shape of the Mapped Data in Feature Space

Using Mercer's theorem, we have shown that one can think of the feature map as a map into a high- or infinite-dimensional Hilbert space. The argument in the remainder of the section shows that this typically entails that the mapped data $\Phi(\mathcal{X})$ lie in some box with rapidly decaying side lengths [606]. By this we mean that the range of the data decreases as the dimension index j increases, with a rate that depends on the size of the eigenvalues.

Let us assume that for all $j \in \mathbb{N}$, we have $\sup_{x\in\mathcal{X}} \lambda_j |\psi_j(x)|^2 < \infty$. Define the sequence

$$l_j := \sup_{x\in\mathcal{X}} \lambda_j |\psi_j(x)|^2. \tag{2.50}$$

Note that if

$$C_k := \sup_j \sup_{x\in\mathcal{X}} |\psi_j(x)| \tag{2.51}$$

exists (see Problem 2.24), then we have $l_j \leq \lambda_j C_k^2$. However, if the λ_j decay rapidly, then (2.50) can be finite even if (2.51) is not.

By construction, $\Phi(\mathcal{X})$ is contained in an axis parallel parallelepiped in $\ell_2^{N_{\mathcal{H}}}$ with side lengths $2\sqrt{l_j}$ (cf. (2.40)).[5]

Consider an example of a common kernel, the Gaussian, and let μ (see Theorem 2.10) be the Lebesgue measure. In this case, the eigenvectors are sine and cosine functions (with supremum one), and thus the sequence of the l_j coincides with the sequence of the eigenvalues λ_j. Generally, whenever $\sup_{x\in\mathcal{X}} |\psi_j(x)|^2$ is finite, the l_j decay as fast as the λ_j. We shall see in Sections 4.4, 4.5 and Chapter 12 that for many common kernels, this decay is very rapid.

It will be useful to consider operators that map $\Phi(\mathcal{X})$ into balls of some radius R centered at the origin. The following proposition characterizes a class of such operators, determined by the sequence $(l_j)_{j\in\mathbb{N}}$. Recall that $\mathbb{R}^{\mathbb{N}}$ denotes the space of all real sequences.

Proposition 2.13 (Mapping $\Phi(\mathcal{X})$ into ℓ_2) *Let S be the diagonal map*

$$\begin{array}{rlll} S: & \mathbb{R}^{\mathbb{N}} & \to & \mathbb{R}^{\mathbb{N}} \\ & (x_j)_j & \mapsto & S(x_j)_j = (s_j x_j)_j, \end{array} \tag{2.52}$$

where $(s_j)_j \in \mathbb{R}^{\mathbb{N}}$. If $\left(s_j\sqrt{l_j}\right)_j \in \ell_2$, then S maps $\Phi(\mathcal{X})$ into a ball centered at the origin whose radius is $R = \left\|\left(s_j\sqrt{l_j}\right)_j\right\|$.

[5] In fact, it is sufficient to use the essential supremum in (2.50). In that case, subsequent statements also only hold true almost everywhere.

Proof Suppose $\left(s_j\sqrt{l_j}\right)_j \in \ell_2$. Using the Mercer map (2.40), we have

$$\|S\Phi(x)\|^2 = \sum_{j\in\mathbb{N}} s_j^2 \lambda_j |\psi_j(x)|^2 \leq \sum_{j\in\mathbb{N}} s_j^2 l_j = R \tag{2.53}$$

for any $x \in \mathcal{X}$. Hence $S\Phi(\mathcal{X}) \subseteq \ell_2$. ■

The converse is not necessarily the case. To see this, note that if $\left(s_j\sqrt{l_j}\right)_j \notin \ell_2$, amounting to saying that

$$\sum_j s_j^2 \sup_{x\in\mathcal{X}} \lambda_j |\psi_j(x)|^2 \tag{2.54}$$

is not finite, then there need not always exist an $x \in \mathcal{X}$ such that $S\Phi(x) = \left(s_j\sqrt{\lambda_j}\psi_j(x)\right)_j \notin \ell_2$, i.e., that

$$\sum_j s_j^2 \lambda_j |\psi_j(x)|^2 \tag{2.55}$$

is not finite.

To see how the freedom to rescale $\Phi(\mathcal{X})$ effectively restricts the class of functions we are using, we first note that everything in the feature space $\mathcal{H} = \ell_2^{N_\mathcal{H}}$ is done in terms of dot products. Therefore, we can compensate any invertible symmetric linear transformation of the data in $\mathcal{H}$ by the inverse transformation on the set of admissible weight vectors in $\mathcal{H}$. In other words, for any invertible symmetric operator S on $\mathcal{H}$, we have $\left\langle S^{-1}\mathbf{w}, S\Phi(x)\right\rangle = \langle \mathbf{w}, \Phi(x)\rangle$ for all $x \in \mathcal{X}$.

As we shall see below (cf. Theorem 5.5, Section 12.4, and Problem 7.5), there exists a class of generalization error bound that depends on the radius R of the smallest sphere containing the data. If the $(l_i)_i$ decay rapidly, we are not actually "making use" of the whole sphere. In this case, we may construct a diagonal scaling operator S which inflates the sides of the above parallelepiped as much as possible, while ensuring that it is still contained within a sphere of the original radius R in $\mathcal{H}$ (Figure 2.3). By effectively reducing the size of the function class, this will provide a way of strengthening the bounds. A similar idea, using kernel PCA (Section 14.2) to determine empirical scaling coefficients, has been successfully applied by [101].

We conclude this section with another useful insight that characterizes a property of the feature map Φ. Note that most of what was said so far applies to the case where the input domain $\mathcal{X}$ is a general set. In this case, it is not possible to make nontrivial statements about continuity properties of Φ. This changes if we assume $\mathcal{X}$ to be endowed with a notion of closeness, by turning it into a so-called topological space. Readers not familiar with this concept will be reassured to hear that Euclidean vector spaces are particular cases of topological spaces.

Continuity of Φ

Proposition 2.14 (Continuity of the Feature Map [402]) *If $\mathcal{X}$ is a topological space and k is a continuous positive definite kernel on $\mathcal{X} \times \mathcal{X}$, then there exists a Hilbert space $\mathcal{H}$ and a continuous map $\Phi : \mathcal{X} \to \mathcal{H}$ such that for all $x, x' \in \mathcal{X}$, we have $k(x, x') = \langle \Phi(x), \Phi(x')\rangle$.*

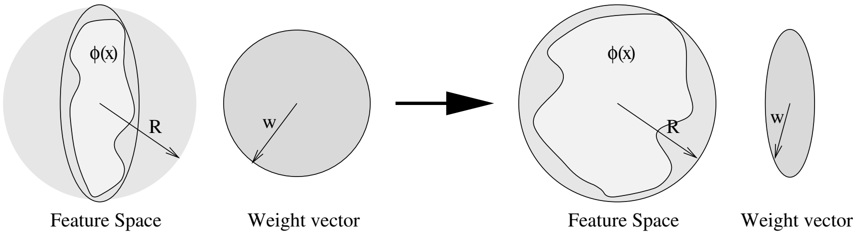


Figure 2.3 Since everything is done in terms of dot products, scaling up the data by an operator S can be compensated by scaling the weight vectors with S^{-1} (cf. text). By choosing S such that the data are still contained in a ball of the same radius R, we effectively reduce our function class (parametrized by the weight vector), which can lead to better generalization bounds, depending on the kernel inducing the map Φ.

2.2.6 The Empirical Kernel Map

The map Φ, defined in (2.21), transforms each input pattern into a function on $\mathcal{X}$, that is, into a potentially infinite-dimensional object. For any given set of points, however, it is possible to approximate Φ by only evaluating it on these points (cf. [232, 350, 361, 547, 474]):

Empirical Kernel Map

Definition 2.15 (Empirical Kernel Map) *For a given set* $\{z_1, \ldots, z_n\} \subset \mathcal{X}$, $n \in \mathbb{N}$, *we call*

$$\Phi_n : \mathbb{R}^N \to \mathbb{R}^n \text{ where } x \mapsto k(.,x)|_{\{z_1,\ldots,z_n\}} = (k(z_1,x),\ldots,k(z_n,x))^\top \tag{2.56}$$

the empirical kernel map *w.r.t.* $\{z_1, \ldots, z_n\}$.

As an example, consider first the case where k is a positive definite kernel, and $\{z_1, \ldots, z_n\} = \{x_1, \ldots, x_m\}$; we thus evaluate $k(.,x)$ on the training patterns. If we carry out a linear algorithm in feature space, then everything will take place in the linear span of the mapped training patterns. Therefore, we can represent the $k(.,x)$ of (2.21) as $\Phi_m(x)$ without losing information. The dot product to use in that representation, however, is not simply the canonical dot product in $\mathbb{R}^m$, since the $\Phi(x_i)$ will usually not form an orthonormal system. To turn Φ_m into a feature map associated with k, we need to endow $\mathbb{R}^m$ with a dot product $\langle \cdot, \cdot \rangle_m$ such that

$$k(x,x') = \langle \Phi_m(x), \Phi_m(x') \rangle_m . \tag{2.57}$$

To this end, we use the ansatz $\langle \cdot, \cdot \rangle_m = \langle \cdot, M \cdot \rangle$, with M being a positive definite matrix.[6] Enforcing (2.57) on the training patterns, this yields the self-consistency condition [478, 512]

$$K = KMK, \tag{2.58}$$

[6] Every dot product in $\mathbb{R}^m$ can be written in this form. We do not require strict definiteness of M, as the null space can be projected out, leading to a lower-dimensional feature space.

where K is the Gram matrix. The condition (2.58) can be satisfied for instance by the (pseudo-)inverse $M = K^{-1}$. Equivalently, we could have incorporated this rescaling operation, which corresponds to a Kernel PCA "whitening" ([478, 547, 474], cf. Section 11.4), directly into the map, by whitening (2.56) to get

Kernel PCA Map

$$\Phi_m^w : x \mapsto K^{-\frac{1}{2}}(k(x_1, x), \ldots, k(x_m, x)). \tag{2.59}$$

This simply amounts to dividing the eigenvector basis vectors of K by $\sqrt{\lambda_i}$, where the λ_i are the eigenvalues of K.[7] This parallels the rescaling of the eigenfunctions of the integral operator belonging to the kernel, given by (2.47). It turns out that this map can equivalently be performed using kernel PCA feature extraction (see Problem 14.8), which is why we refer to this map as the *kernel PCA map*.

Note that we have thus constructed a data-dependent feature map into an m-dimensional space which satisfies $\langle \Phi_m^w(x), \Phi_m^w(x') \rangle = k(x, x')$, i.e., we have found an m-dimensional feature space associated with the given kernel. In the case where K is invertible, $\Phi_m^w(x)$ computes the coordinates of $\Phi(x)$ when represented in a basis of the m-dimensional subspace spanned by $\Phi(x_1), \ldots, \Phi(x_m)$.

For data sets where the number of examples is smaller than their dimension, it can actually be computationally attractive to carry out Φ_m^w explicitly, rather than using kernels in subsequent algorithms. Moreover, algorithms which are not readily "kernelized" may benefit from explicitly carrying out the kernel PCA map.

We end this section with two notes which illustrate why the use of (2.56) need not be restricted to the special case we just discussed.

- *More general kernels.* When using non-symmetric kernels k in (2.56), together with the canonical dot product, we effectively work with the positive definite matrix $K^\top K$. Note that each positive definite matrix can be written as $K^\top K$. Therefore, working with positive definite kernels leads to an equally rich set of nonlinearities as working with an empirical kernel map using general non-symmetric kernels. If we wanted to carry out the whitening step, we would have to use $(K^\top K)^{-1/4}$ (cf. footnote 7 concerning potential singularities).

- *Different evaluation sets.* Things can be sped up by using expansion sets of the form $\{z_1, \ldots, z_n\}$, mapping into an n-dimensional space, with $n < m$, as done in [100, 228]. In that case, one modifies (2.59) to

$$\Phi_n^w : x \mapsto K_n^{-\frac{1}{2}}(k(z_1, x), \ldots, k(z_n, x)), \tag{2.60}$$

where $(K_n)_{ij} := k(z_i, z_j)$. The expansion set can either be a subset of the training set,[8] or some other set of points. We will later return to the issue of how to choose

[7] It is understood that if K is singular, we use the pseudo-inverse of $K^{1/2}$ in which case we get an even lower dimensional subspace.

[8] In [228] it is recommended that the size n of the expansion set is chosen large enough to ensure that the smallest eigenvalue of K_n is larger than some predetermined $\epsilon > 0$. Alternatively, one can start off with a larger set, and use kernel PCA to *select* the most important components for the map, see Problem 14.8. In the kernel PCA case, the map (2.60) is com-

the best set (see Section 10.2 and Chapter 18). As an aside, note that in the case of Kernel PCA (see Section 1.7 and Chapter 14 below), one does not need to worry about the whitening step in (2.59) and (2.60): using the canonical dot product in $\mathbb{R}^m$ (rather than $\langle \cdot, \cdot \rangle$) will simply lead to diagonalizing K^2 instead of K, which yields the same eigenvectors with squared eigenvalues. This was pointed out by [350, 361]. The study [361] reports experiments where (2.56) was employed to speed up Kernel PCA by choosing $\{z_1, \ldots, z_n\}$ as a subset of $\{x_1, \ldots, x_m\}$.

2.2.7 A Kernel Map Defined from Pairwise Similarities

In practice, we are given a finite amount of data $x_1, \ldots, x_m$. The following simple observation shows that even if we do not want to (or are unable to) analyze a given kernel k analytically, we can still compute a map Φ such that k corresponds to a dot product in the linear span of the $\Phi(x_i)$:

Proposition 2.16 (Data-Dependent Kernel Map [467]) *Suppose the data $x_1, \ldots, x_m$ and the kernel k are such that the kernel Gram matrix $K_{ij} = k(x_i, x_j)$ is positive definite. Then it is possible to construct a map Φ into an m-dimensional feature space $\mathcal{H}$ such that*

$$k(x_i, x_j) = \left\langle \Phi(x_i), \Phi(x_j) \right\rangle . \tag{2.61}$$

Conversely, given an arbitrary map Φ into some feature space $\mathcal{H}$, the matrix $K_{ij} = \left\langle \Phi(x_i), \Phi(x_j) \right\rangle$ is positive definite.

Proof First assume that K is positive definite. In this case, it can be diagonalized as $K = SDS^\top$, with an orthogonal matrix S and a diagonal matrix D with nonnegative entries. Then

$$k(x_i, x_j) = (SDS^\top)_{ij} = \left\langle S_i, DS_j \right\rangle = \left\langle \sqrt{D} S_i, \sqrt{D} S_j \right\rangle , \tag{2.62}$$

where we have defined the S_i as the rows of S (note that the columns of S would be K's eigenvectors). Therefore, K is the Gram matrix of the vectors $\sqrt{D_{ii}} \cdot S_i$.[9] Hence the following map Φ, defined on $x_1, \ldots, x_m$ will satisfy (2.61)

$$\Phi : x_i \mapsto \sqrt{D_{ii}} \cdot S_i . \tag{2.63}$$

Thus far, Φ is only defined on a set of points, rather than on a vector space. Therefore, it makes no sense to ask whether it is linear. We can, however, ask whether it can be *extended* to a linear map, provided the x_i are elements of a vector space. The answer is that if the x_i are linearly dependent (which is often the case), then this will not be possible, since a linear map would then typically be over-

puted as $D_n^{-1/2} U_n^\top (k(z_1, x), \ldots, k(z_n, x))$, where $U_n D_n U_n^\top$ is the eigenvalue decomposition of K_n. Note that the columns of U_n are the eigenvectors of K_n. We discard all columns that correspond to zero eigenvalues, as well as the corresponding dimensions of D_n. To *approximate* the map, we may actually discard all eigenvalues smaller than some $\epsilon > 0$.

[9] In fact, every positive definite matrix is the Gram matrix of some set of vectors [46].

determined by the m conditions (2.63).

For the converse, assume an arbitrary $\boldsymbol{\alpha} \in \mathbb{R}^m$, and compute

$$\sum_{i,j=1}^{m} \alpha_i \alpha_j K_{ij} = \left\langle \sum_{i=1}^{m} \alpha_i \Phi(x_i), \sum_{j=1}^{m} \alpha_j \Phi(x_j) \right\rangle = \left\| \sum_{i=1}^{m} \alpha_i \Phi(x_i) \right\|^2 \geq 0. \tag{2.64}$$

■

In particular, this result implies that given data $x_1, \ldots, x_m$, and a kernel k which gives rise to a positive definite matrix K, it is always possible to construct a feature space $\mathcal{H}$ of dimension at most m that we are implicitly working in when using kernels (cf. Problem 2.32 and Section 2.2.6).

If we perform an algorithm which requires k to correspond to a dot product in some other space (as for instance the SV algorithms described in this book), it is possible that even though k is not positive definite in general, it still gives rise to a positive definite Gram matrix K with respect to the training data at hand. In this case, Proposition 2.16 tells us that nothing will go wrong during training when we work with these data. Moreover, if k leads to a matrix with some small negative eigenvalues, we can add a small multiple of some strictly positive definite kernel k' (such as the identity $k'(x_i, x_j) = \delta_{ij}$) to obtain a positive definite matrix. To see this, suppose that $\lambda_{\min} < 0$ is the minimal eigenvalue of k's Gram matrix. Note that being strictly positive definite, the Gram matrix K' of k' satisfies

$$\min_{\|\boldsymbol{\alpha}\|=1} \langle \boldsymbol{\alpha}, K'\boldsymbol{\alpha} \rangle \geq \lambda'_{\min} > 0, \tag{2.65}$$

where $\lambda'_{\min}$ denotes its minimal eigenvalue, and the first inequality follows from Rayleigh's principle (B.57). Therefore, provided that $\lambda_{\min} + \lambda\lambda'_{\min} \geq 0$, we have

$$\langle \boldsymbol{\alpha}, (K + \lambda K')\boldsymbol{\alpha} \rangle = \langle \boldsymbol{\alpha}, K\boldsymbol{\alpha} \rangle + \lambda \langle \boldsymbol{\alpha}, K'\boldsymbol{\alpha} \rangle \geq \|\boldsymbol{\alpha}\|^2 \left(\lambda_{\min} + \lambda\lambda'_{\min}\right) \geq 0 \tag{2.66}$$

for all $\boldsymbol{\alpha} \in \mathbb{R}^m$, rendering $(K + \lambda K')$ positive definite.

2.3 Examples and Properties of Kernels

For the following examples, let us assume that $\mathcal{X} \subset \mathbb{R}^N$. Besides homogeneous polynomial kernels (cf. Proposition 2.1),

Polynomial

$$k(x, x') = \langle x, x' \rangle^d, \tag{2.67}$$

Boser, Guyon, and Vapnik [62, 223, 561] suggest the usage of Gaussian radial basis function kernels [26, 4],

Gaussian

$$k(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\,\sigma^2}\right), \tag{2.68}$$

Sigmoid

where $\sigma > 0$, and sigmoid kernels,

$$k(x, x') = \tanh(\kappa \langle x, x' \rangle + \vartheta), \tag{2.69}$$

where $\kappa > 0$ and $\vartheta < 0$. By applying Theorem 13.4 below, one can check that the latter kernel is not actually positive definite (see Section 4.6 and [85, 511] and the discussion in Example 4.25). Curiously, it has nevertheless successfully been used in practice. The reasons for this are discussed in [467].

Inhomogeneous Polynomial

Other useful kernels include the inhomogeneous polynomial,

$$k(x, x') = \left(\langle x, x' \rangle + c\right)^d, \tag{2.70}$$

($d \in \mathbb{N}, c \geq 0$) and the B_n-spline kernel [501, 572] (I_X denoting the indicator (or characteristic) function on the set X, and $\otimes$ the convolution operation, $(f \otimes g)(x) := \int f(x')g(x' - x)dx'$),

B_n-Spline of Odd Order

$$k(x, x') = B_{2p+1}(\|x - x'\|) \text{ with } B_n := \bigotimes_{i=1}^{n} I_{[-\frac{1}{2},\frac{1}{2}]}. \tag{2.71}$$

The kernel computes B-splines of order $2p + 1$ ($p \in \mathbb{N}$), defined by the $(2p + 1)$-fold convolution of the unit interval $[-1/2, 1/2]$. See Section 4.4.1 for further details and a regularization theoretic analysis of this kernel.

Invariance of Kernels

Note that all these kernels have the convenient property of unitary invariance, $k(x, x') = k(Ux, Ux')$ if $U^\top = U^{-1}$, for instance if U is a rotation. If we consider complex numbers, then we have to use the adjoint $U^* := \overline{U}^\top$ instead of the transpose.

RBF Kernels

Radial basis function (RBF) kernels are kernels that can be written in the form

$$k(x, x') = f(d(x, x')), \tag{2.72}$$

where d is a metric on $\mathcal{X}$, and f is a function on $\mathbb{R}_0^+$. Examples thereof are the Gaussians and B-splines mentioned above. Usually, the metric arises from the dot product; $d(x, x') = \|x - x'\| = \sqrt{\langle x - x', x - x' \rangle}$. In this case, RBF kernels are unitary invariant, too. In addition, they are translation invariant; in other words, $k(x, x') = k(x + x_0, x' + x_0)$ for all $x_0 \in \mathcal{X}$.

In some cases, invariance properties alone can distinguish particular kernels: in Section 2.1, we explained how using polynomial kernels $\langle x, x' \rangle^d$ corresponds to mapping into a feature space whose dimensions are spanned by all possible dth order monomials in input coordinates. The different dimensions are scaled with the square root of the number of ordered products of the respective d entries (e.g., $\sqrt{2}$ in (2.13)). These scaling factors precisely ensure invariance under the group of all orthogonal transformations (rotations and mirroring operations). In many cases, this is a desirable property: it ensures that the results of a learning procedure do not depend on which orthonormal coordinate system (with fixed origin) we use for representing our input data.

Proposition 2.17 (Invariance of Polynomial Kernels [480]) *Up to a scaling factor, the kernel $k(x, x') = \langle x, x' \rangle^d$ is the only kernel inducing a map into a space of* all *monomials of degree d which is invariant under orthogonal transformations of $\mathbb{R}^N$.*

Properties of RBF Kernels

Some interesting additional structure exists in the case of a Gaussian RBF kernel k (2.68). As $k(x, x) = 1$ for all $x \in \mathcal{X}$, each mapped example has unit length, $\|\Phi(x)\| =$

1 (Problem 2.18 shows how to achieve this for general kernels). Moreover, as $k(x, x') > 0$ for all $x, x' \in \mathcal{X}$, all points lie inside the same orthant in feature space. To see this, recall that for unit length vectors, the dot product (1.3) equals the cosine of the enclosed angle. We obtain

$$\cos(\angle(\Phi(x), \Phi(x'))) = \langle \Phi(x), \Phi(x') \rangle = k(x, x') > 0, \tag{2.73}$$

which amounts to saying that the enclosed angle between any two mapped examples is smaller than $\pi/2$.

The above seems to indicate that in the Gaussian case, the mapped data lie in a fairly restricted area of feature space. However, in another sense, they occupy a space which is as large as possible:

Theorem 2.18 (Full Rank of Gaussian RBF Gram Matrices [360]) *Suppose that* $x_1, \ldots, x_m \subset \mathcal{X}$ *are distinct points, and* $\sigma \neq 0$. *The matrix* K *given by*

$$K_{ij} := \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right) \tag{2.74}$$

has full rank.

Infinite-Dimensional Feature Space

In other words, the points $\Phi(x_1), \ldots, \Phi(x_m)$ are linearly independent (provided no two x_i are the same). They span an m-dimensional subspace of $\mathcal{H}$. Therefore a Gaussian kernel defined on a domain of infinite cardinality, with no a priori restriction on the number of training examples, produces a feature space of *infinite* dimension. Nevertheless, an analysis of the shape of the mapped data in feature space shows that capacity is distributed in a way that ensures smooth and simple estimates whenever possible (see Section 12.4).

The examples given above all apply to the case of vectorial data. Let us next give an example where $\mathcal{X}$ is *not* a vector space [42].

Proposition 2.19 (Similarity of Probabilistic Events) *If* $(\mathcal{X}, \mathcal{C}, \mathrm{P})$ *is a probability space with* σ*-algebra* $\mathcal{C}$ *and probability measure* P, *then*

$$k(A, B) = \mathrm{P}(A \cap B) - \mathrm{P}(A)\mathrm{P}(B) \tag{2.75}$$

is a positive definite kernel on $\mathcal{C} \times \mathcal{C}$.

Proof To see this, we define a feature map

$$\Phi : A \mapsto (I_A - \mathrm{P}(A)), \tag{2.76}$$

where I_A is the characteristic function on A. On the feature space, which consists of functions on $\mathcal{X}$ taking values in $[-1, 1]$, we use the dot product

$$\langle f, g \rangle := \int_{\mathcal{X}} f \cdot g \, d\mathrm{P}. \tag{2.77}$$

The result follows by noticing $\langle I_A, I_B \rangle = \mathrm{P}(A \cap B)$ and $\langle I_A, \mathrm{P}(B) \rangle = \mathrm{P}(A)\mathrm{P}(B)$. ■

Further examples include kernels for string matching, as proposed by [585, 234, 23]. We shall describe these, and address the general problem of designing kernel functions, in Chapter 13.

The next section will return to the connection between kernels and feature spaces. Readers who are eager to move on to SV algorithms may want to skip this section, which is somewhat more technical.

2.4 The Representation of Dissimilarities in Linear Spaces

2.4.1 Conditionally Positive Definite Kernels

We now proceed to a larger class of kernels than that of the positive definite ones. This larger class is interesting in several regards. First, it will turn out that some kernel algorithms work with this class, rather than only with positive definite kernels. Second, its relationship to positive definite kernels is a rather interesting one, and a number of connections between the two classes provide understanding of kernels in general. Third, they are intimately related to a question which is a variation on the central aspect of positive definite kernels: the latter can be thought of as dot products in feature spaces; the former, on the other hand, can be embedded as *distance measures* arising from norms in feature spaces.

The present section thus attempts to extend the utility of the kernel trick by looking at the problem of which kernels can be used to compute distances in feature spaces. The underlying mathematical results have been known for quite a while [465]; some of them have already attracted interest in the kernel methods community in various contexts [515, 234].

Clearly, the squared distance $\|\Phi(x) - \Phi(x')\|^2$ in the feature space associated with a pd kernel k can be computed, using $k(x, x') = \langle \Phi(x), \Phi(x') \rangle$, as

$$\|\Phi(x) - \Phi(x')\|^2 = k(x, x) + k(x', x') - 2k(x, x'). \tag{2.78}$$

Positive definite kernels are, however, not the full story: there exists a *larger* class of kernels that can be used as generalized distances, and the present section will describe why and how [468].

Let us start by considering how a dot product and the corresponding distance measure are affected by a translation of the data, $x \mapsto x - x_0$. Clearly, $\|x - x'\|^2$ is translation invariant while $\langle x, x' \rangle$ is not. A short calculation shows that the effect of the translation can be expressed in terms of $\|. - .\|^2$ as

$$\langle (x - x_0), (x' - x_0) \rangle = \frac{1}{2}\left(-\|x - x'\|^2 + \|x - x_0\|^2 + \|x_0 - x'\|^2\right). \tag{2.79}$$

Note that this, just like $\langle x, x' \rangle$, is still a pd kernel: $\sum_{i,j} c_i c_j \langle (x_i - x_0), (x_j - x_0) \rangle = \|\sum_i c_i (x_i - x_0)\|^2 \geq 0$ holds true for any c_i. For any choice of $x_0 \in \mathcal{X}$, we thus get a similarity measure (2.79) associated with the dissimilarity measure $\|x - x'\|$.

This naturally leads to the question of whether (2.79) might suggest a connection that also holds true in more general cases: what kind of nonlinear dissimilarity measure do we have to substitute for $\|. - .\|^2$ on the right hand side of (2.79), to ensure that the left hand side becomes positive definite? To state the answer, we first need to define the appropriate class of kernels.

The following definition differs from Definition 2.4 only in the additional constraint on the sum of the c_i. Below, $\mathbb{K}$ is a shorthand for $\mathbb{C}$ or $\mathbb{R}$; the definitions are the same in both cases.

Definition 2.20 (Conditionally Positive Definite Matrix) *A symmetric $m \times m$ matrix K ($m \geq 2$) taking values in $\mathbb{K}$ and satisfying*

$$\sum_{i,j=1}^{m} c_i \bar{c}_j K_{ij} \geq 0 \text{ for all } c_i \in \mathbb{K}, \ \text{ with } \sum_{i=1}^{m} c_i = 0, \tag{2.80}$$

is called conditionally positive definite (cpd).

Definition 2.21 (Conditionally Positive Definite Kernel) *Let $\mathcal{X}$ be a nonempty set. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{K}$ which for all $m \geq 2, x_1, \ldots, x_m \in \mathcal{X}$ gives rise to a conditionally positive definite Gram matrix is called a* conditionally positive definite (cpd) kernel.

Note that symmetry is also required in the complex case. Due to the additional constraint on the coefficients c_i, it does not follow automatically anymore, as it did in the case of complex positive definite matrices and kernels. In Chapter 4, we will revisit cpd kernels. There, we will actually introduce cpd kernels of different orders. The definition given in the current chapter covers the case of kernels which are cpd *of order* 1.

Connection PD — CPD

Proposition 2.22 (Constructing PD Kernels from CPD Kernels [42]) *Let $x_0 \in \mathcal{X}$, and let k be a symmetric kernel on $\mathcal{X} \times \mathcal{X}$. Then*

$$\tilde{k}(x, x') := \frac{1}{2}(k(x, x') - k(x, x_0) - k(x_0, x') + k(x_0, x_0))$$

is positive definite if and only if k is conditionally positive definite.

The proof follows directly from the definitions and can be found in [42]. This result does generalize (2.79): the negative squared distance kernel is indeed cpd, since $\sum_i c_i = 0$ implies $-\sum_{i,j} c_i c_j \|x_i - x_j\|^2 = -\sum_i c_i \sum_j c_j \|x_j\|^2 - \sum_j c_j \sum_i c_i \|x_i\|^2 + 2\sum_{i,j} c_i c_j \langle x_i, x_j \rangle = 2\sum_{i,j} c_i c_j \langle x_i, x_j \rangle = 2\|\sum_i c_i x_i\|^2 \geq 0$. In fact, this implies that all kernels of the form

$$k(x, x') = -\|x - x'\|^{\beta}, 0 \leq \beta \leq 2 \tag{2.81}$$

are cpd (they are not pd),[10] by application of the following result (note that the case $\beta = 0$ is trivial):

[10] Moreover, they are not cpd if $\beta > 2$ [42].

Proposition 2.23 (Fractional Powers and Logs of CPD Kernels [42]) *If $k : \mathcal{X} \times \mathcal{X} \to (-\infty, 0]$ is cpd, then so are $-(-k)^\alpha$ $(0 < \alpha < 1)$ and $-\ln(1-k)$.*

To state another class of cpd kernels that are not pd, note first that as a trivial consequence of Definition 2.20, we know that (i) sums of cpd kernels are cpd, and (ii) any constant $b \in \mathbb{R}$ is a cpd kernel. Therefore, any kernel of the form $k + b$, where k is cpd and $b \in \mathbb{R}$, is also cpd. In particular, since pd kernels are cpd, we can take any pd kernel and offset it by b, and it will still be at least cpd. For further examples of cpd kernels, cf. [42, 578, 205, 515].

2.4.2 Hilbert Space Representation of CPD Kernels

We now return to the main flow of the argument. Proposition 2.22 allows us to construct the feature map for k from that of the pd kernel $\tilde{k}$. To this end, fix $x_0 \in \mathcal{X}$ and define $\tilde{k}$ according to Proposition 2.22. Due to Proposition 2.22, $\tilde{k}$ is positive definite. Therefore, we may employ the Hilbert space representation $\Phi : \mathcal{X} \to \mathcal{H}$ of $\tilde{k}$ (cf. (2.32)), satisfying $\langle \Phi(x), \Phi(x') \rangle = \tilde{k}(x, x')$; hence,

$$\|\Phi(x) - \Phi(x')\|^2 = \tilde{k}(x, x) + \tilde{k}(x', x') - 2\tilde{k}(x, x'). \tag{2.82}$$

Substituting Proposition 2.22 yields

$$\|\Phi(x) - \Phi(x')\|^2 = -k(x, x') + \frac{1}{2}\left(k(x, x) + k(x', x')\right). \tag{2.83}$$

This implies the following result [465, 42].

Feature Map for CPD Kernels

Proposition 2.24 (Hilbert Space Representation of CPD Kernels) *Let k be a real-valued CPD kernel on $\mathcal{X}$, satisfying $k(x, x) = 0$ for all $x \in \mathcal{X}$. Then there exists a Hilbert space $\mathcal{H}$ of real-valued functions on $\mathcal{X}$, and a mapping $\Phi : \mathcal{X} \to \mathcal{H}$, such that*

$$\|\Phi(x) - \Phi(x')\|^2 = -k(x, x'). \tag{2.84}$$

If we drop the assumption $k(x, x) = 0$, the Hilbert space representation reads

$$\|\Phi(x) - \Phi(x')\|^2 = -k(x, x') + \frac{1}{2}\left(k(x, x) + k(x', x')\right). \tag{2.85}$$

It can be shown that if $k(x, x) = 0$ for all $x \in \mathcal{X}$, then

$$d(x, x') := \sqrt{-k(x, x')} = \|\Phi(x) - \Phi(x')\| \tag{2.86}$$

is a semi-metric: clearly, it is nonnegative and symmetric; additionally, it satisfies the triangle inequality, as can be seen by computing $d(x, x') + d(x', x'') = \|\Phi(x) - \Phi(x')\| + \|\Phi(x') - \Phi(x'')\| \geq \|\Phi(x) - \Phi(x'')\| = d(x, x'')$ [42].

It is a metric if $k(x, x') \neq 0$ for $x \neq x'$. We thus see that we can rightly think of k as the negative of a distance measure.

We next show how to represent *general* symmetric kernels (thus in particular cpd kernels) as symmetric bilinear forms Q in feature spaces. This generalization of the previously known feature space representation for pd kernels comes at a cost: Q will no longer be a dot product. For our purposes, we can get away with this. The result will give us an intuitive understanding of Proposition 2.22: we can then write $\tilde{k}$ as $\tilde{k}(x, x') := Q(\Phi(x) - \Phi(x_0), \Phi(x') - \Phi(x_0))$. Proposition 2.22 thus essentially adds an origin in feature space which corresponds to the image $\Phi(x_0)$ of one point x_0 under the feature map.

Feature Map for General Symmetric Kernels

Proposition 2.25 (Vector Space Representation of Symmetric Kernels) *Let k be a real-valued symmetric kernel on $\mathcal{X}$. Then there exists a linear space $\mathcal{H}$ of real-valued functions on $\mathcal{X}$, endowed with a symmetric bilinear form $Q(.,.)$, and a mapping $\Phi : \mathcal{X} \to \mathcal{H}$, such that $k(x, x') = Q(\Phi(x), \Phi(x'))$.*

Proof The proof is a direct modification of the pd case. We use the map (2.21) and linearly complete the image as in (2.22). Define $Q(f, g) := \sum_{i=1}^{m} \sum_{j=1}^{m'} \alpha_i \beta_j k(x_i, x'_j)$. To see that it is well-defined, although it explicitly contains the expansion coefficients (which need not be unique), note that $Q(f, g) = \sum_{j=1}^{m'} \beta_j f(x'_j)$, independent of the α_i. Similarly, for g, note that $Q(f, g) = \sum_i \alpha_i g(x_i)$, hence it is independent of β_j. The last two equations also show that Q is bilinear; clearly, it is symmetric. ■

Note, moreover, that by definition of Q, k is a reproducing kernel for the feature space (which is not a Hilbert space): for all functions f (2.22), we have $Q(k(., x), f) = f(x)$; in particular, $Q(k(., x), k(., x')) = k(x, x')$.

Rewriting $\tilde{k}$ as $\tilde{k}(x, x') := Q(\Phi(x) - \Phi(x_0), \Phi(x') - \Phi(x_0))$ suggests an immediate generalization of Proposition 2.22: in practice, we might want to choose other points as origins in feature space — points that do not have a pre-image x_0 in the input domain, such as the mean of a set of points (cf. [543]). This will be useful when considering kernel PCA. It is only crucial that the behavior of our reference point under translation is identical to that of individual points. This is taken care of by the constraint on the sum of the c_i in the following proposition.

Matrix Centering

Proposition 2.26 (Exercise 2.23 in [42]) *Let K be a symmetric matrix, $\mathbf{e} \in \mathbb{R}^m$ be the vector of all ones, $\mathbf{1}$ the $m \times m$ identity matrix, and let $\mathbf{c} \in \mathbb{C}^m$ satisfy $\mathbf{e}^*\mathbf{c} = 1$. Then*

$$\tilde{K} := (\mathbf{1} - \mathbf{e}\mathbf{c}^*)K(\mathbf{1} - \mathbf{c}\mathbf{e}^*) \tag{2.87}$$

is positive definite if and only if K is conditionally positive definite.[11]

Proof "$\Longrightarrow$": suppose $\tilde{K}$ is positive definite. Thus for any $\mathbf{a} \in \mathbb{C}^m$ which satisfies $\mathbf{a}^*\mathbf{e} = \mathbf{e}^*\mathbf{a} = 0$, we have $0 \leq \mathbf{a}^*\tilde{K}\mathbf{a} = \mathbf{a}^*K\mathbf{a} + \mathbf{a}^*\mathbf{e}\mathbf{c}^*K\mathbf{c}\mathbf{e}^*\mathbf{a} - \mathbf{a}^*K\mathbf{c}\mathbf{e}^*\mathbf{a} - \mathbf{a}^*\mathbf{e}\mathbf{c}^*K\mathbf{a} = \mathbf{a}^*K\mathbf{a}$. This means that $0 \leq \mathbf{a}^*K\mathbf{a}$, proving that K is conditionally positive definite.

"$\Longleftarrow$": suppose K is conditionally positive definite. This means that we have to show that $\mathbf{a}^*\tilde{K}\mathbf{a} \geq 0$ for all $\mathbf{a} \in \mathbb{C}^m$. We have

$$\mathbf{a}^*\tilde{K}\mathbf{a} = \mathbf{a}^*(\mathbf{1} - \mathbf{e}\mathbf{c}^*)K(\mathbf{1} - \mathbf{c}\mathbf{e}^*)\mathbf{a} = \mathbf{s}^*K\mathbf{s} \text{ for } \mathbf{s} = (\mathbf{1} - \mathbf{c}\mathbf{e}^*)\mathbf{a}. \tag{2.88}$$

[11] $\mathbf{c}^*$ is the vector obtained by transposing and taking the complex conjugate of $\mathbf{c}$.

All we need to show is $\mathbf{e}^*\mathbf{s} = 0$, since then we can use the fact that K is cpd to obtain $\mathbf{s}^* K\mathbf{s} \geq 0$. This can be seen as follows $\mathbf{e}^*\mathbf{s} = \mathbf{e}^*(\mathbf{1} - \mathbf{c}\mathbf{e}^*)\mathbf{a} = (\mathbf{e}^* - (\mathbf{e}^*\mathbf{c})\mathbf{e}^*)\mathbf{a} = (\mathbf{e}^* - \mathbf{e}^*)\mathbf{a} = 0$. ■

This result directly implies a corresponding generalization of Proposition 2.22:

Kernel Centering

Proposition 2.27 (Adding a General Origin) *Let k be a symmetric kernel, $x_1, \ldots, x_m \in \mathcal{X}$, and let $c_i \in \mathbb{C}$ satisfy $\sum_{i=1}^m c_i = 1$. Then*

$$\tilde{k}(x, x') := \frac{1}{2}\Big(k(x, x') - \sum_{i=1}^m c_i k(x, x_i) - \sum_{i=1}^m c_i k(x_i, x') + \sum_{i,j=1}^m c_i c_j k(x_i, x_j)\Big)$$

is positive definite if and only if k is conditionally positive definite.

Proof Consider a set of $m' \in \mathbb{N}$ points $x'_1, \ldots, x'_{m'} \in \mathcal{X}$, and let K be the $(m + m') \times (m + m')$ Gram matrix based on $x_1, \ldots, x_m, x'_1, \ldots, x'_{m'}$. Apply Proposition 2.26 using $c_{m+1} = \ldots = c_{m+m'} = 0$. ■

Application to SVMs

The above results show that conditionally positive definite kernels are a natural choice whenever we are dealing with a translation invariant problem, such as the SVM: maximization of the margin of separation between two classes of data is independent of the position of the origin. Seen in this light, it is not surprising that the structure of the dual optimization problem (cf. [561]) allows cpd kernels: as noted in [515, 507], the constraint $\sum_{i=1}^m \alpha_i y_i = 0$ projects out the same subspace as (2.80) in the definition of conditionally positive definite matrices.

Application to Kernel PCA

Another example of a kernel algorithm that works with conditionally positive definite kernels is Kernel PCA (Chapter 14), where the data are centered, thus removing the dependence on the origin in feature space. Formally, this follows from Proposition 2.26 for $c_i = 1/m$.

Application to Parzen Windows Classifiers

Let us consider another example. One of the simplest distance-based classification algorithms proceeds as follows. Given m_+ points labelled with $+1$, m_- points labelled with -1, and a mapped test point $\Phi(x)$, we compute the mean squared distances between the latter and the two classes, and assign it to the one for which this mean is smaller;

$$y = \operatorname{sgn}\left(\frac{1}{m_-}\sum_{y_i=-1}\|\Phi(x) - \Phi(x_i)\|^2 - \frac{1}{m_+}\sum_{y_i=1}\|\Phi(x) - \Phi(x_i)\|^2\right). \tag{2.89}$$

We use the distance kernel trick (Proposition 2.24) to express the decision function as a kernel expansion in the input domain: a short calculation shows that

$$y = \operatorname{sgn}\left(\frac{1}{m_+}\sum_{y_i=1} k(x, x_i) - \frac{1}{m_-}\sum_{y_i=-1} k(x, x_i) + b\right), \tag{2.90}$$

with the constant offset

$$b = \frac{1}{2m_-}\sum_{y_i=-1} k(x_i, x_i) - \frac{1}{2m_+}\sum_{y_i=1} k(x_i, x_i). \tag{2.91}$$

Note that for some cpd kernels, such as (2.81), $k(x_i, x_i)$ is always 0, and thus $b = 0$. For others, such as the commonly used Gaussian kernel, $k(x_i, x_i)$ is a nonzero constant, in which case b vanishes provided that $m_+ = m_-$. For normalized Gaussians, the resulting decision boundary can be interpreted as the Bayes decision based on two Parzen window density estimates of the classes; for general cpd kernels, the analogy is merely a formal one; that is, the decision functions take the same form.

Properties of CPD Kernels

Many properties of positive definite kernels carry over to the more general case of conditionally positive definite kernels, such as Proposition 13.1.

Using Proposition 2.22, one can prove an interesting connection between the two classes of kernels:

Proposition 2.28 (Connection PD — CPD [465]) *A kernel k is conditionally positive definite if and only if* $\exp(tk)$ *is positive definite for all $t > 0$.*

Positive definite kernels of the form $\exp(tk)$ ($t > 0$) have the interesting property that their nth root ($n \in \mathbb{N}$) is again a positive definite kernel. Such kernels are called *infinitely divisible*. One can show that, disregarding some technicalities, the logarithm of an infinitely divisible positive definite kernel mapping into $\mathbb{R}_0^+$ is a conditionally positive definite kernel.

2.4.3 Higher Order CPD Kernels

For the sake of completeness, we now present some material which is of interest to one section later in the book (Section 4.8), but not central for the present chapter. We follow [341, 204].

Definition 2.29 (Conditionally Positive Definite Functions of Order q) *A continuous function h, defined on* $[0, \infty)$, *is called conditionally positive definite (cpd) of order q on $\mathbb{R}^N$ if for any distinct points $x_1, \ldots, x_m \in \mathbb{R}^N$, the quadratic form,*

$$\sum_{i,j=1}^{m} \alpha_i \alpha_j h(\|x_i - x_j\|^2), \tag{2.92}$$

is nonnegative, provided that the scalars $\alpha_1, \ldots, \alpha_m$ satisfy $\sum_{i=1}^{m} \alpha_i p(x_i) = 0$, for all polynomials $p(\cdot)$ on $\mathbb{R}^N$ of degree lower than q.

Let Π_q^N denote the space of polynomials of degree lower than q on $\mathbb{R}^N$. By definition, every cpd function h of order q generates a positive definite kernel for SV expansions in the space of functions orthogonal to Π_q^N, by setting $k(x, x') := h(\|x - x'\|^2)$.

There exists also an analogue to the positive definiteness of the integral operator in the conditions of Mercer's theorem. In [157, 341] it is shown that for cpd functions h of order q, we have

$$\int h(\|x - x'\|^2) f(x) f(x') dx dx' \geq 0, \tag{2.93}$$

provided that the projection of f onto Π_q^N is zero.

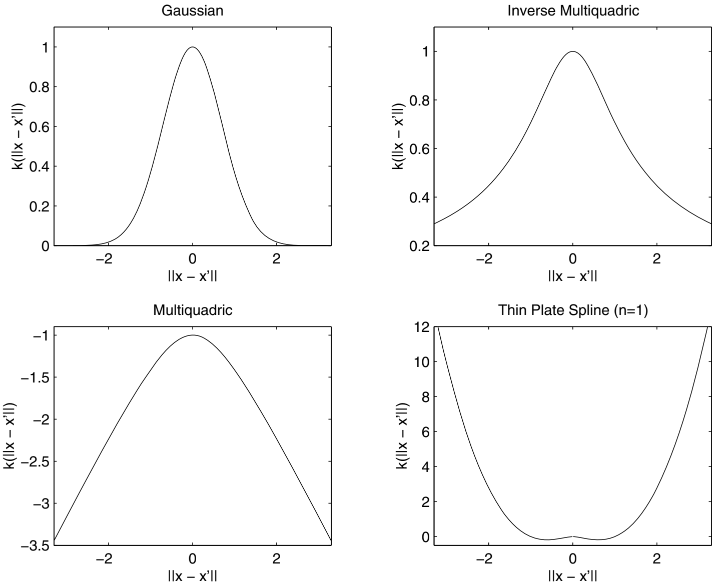


Figure 2.4 Conditionally positive definite functions, as described in Table 2.1. Where applicable, we set the free parameter c to 1; β is set to 2. Note that cpd kernels need not be positive anywhere (e.g., the Multiquadric kernel).

Table 2.1 Examples of Conditionally Positive Definite Kernels. The fact that the exponential kernel is pd (i.e., cpd of order 0) follows from (2.81) and Proposition 2.28.

Kernel	Order	
$e^{-c\|x-x'\|^{\beta}}, 0 \leq \beta \leq 2$	0	Exponential
$\frac{1}{\sqrt{\|x-x'\|^2+c^2}}$	0	Inverse Multiquadric
$-\sqrt{\|x-x'\|^2+c^2}$	1	Multiquadric
$\|x-x'\|^{2n}\ln\|x-x'\|$	n	Thin Plate Spline

Definition 2.30 (Completely Monotonic Functions) *A function $h(x)$ is called completely monotonic of order q if*

$$(-1)^n \frac{d^n}{dx^n} h(x) \geq 0 \textit{ for all } x \in [0, \infty) \textit{ and } n \geq q. \tag{2.94}$$

It can be shown [464, 465, 360] that a function $h(x^2)$ is conditionally positive definite if and only if $h(x)$ is completely monotonic of the same order. This gives a (sometimes simpler) criterion for checking whether a function is cpd or not.

If we use cpd kernels in learning algorithms, we must ensure orthogonality of the estimate with respect to Π_q^N. This is usually done via constraints $\sum_{i=1}^m \alpha_i p(x_i) = 0$ for all polynomials $p(\cdot)$ on $\mathbb{R}^N$ of degree lower than q (see Section 4.8).

2.5 Summary

The crucial ingredient of SVMs and other kernel methods is the so-called kernel trick (see (2.7) and Remark 2.8), which permits the computation of dot products in high-dimensional feature spaces, using simple functions defined on pairs of input patterns. This trick allows the formulation of nonlinear variants of any algorithm that can be cast in terms of dot products, SVMs being but the most prominent example. The mathematical result underlying the kernel trick is almost a century old [359]. Nevertheless, it was only much later that it was exploited by the machine learning community for the analysis [4] and construction of algorithms [62], and that it was described as a general method for constructing nonlinear generalizations of dot product algorithms [480].

The present chapter has reviewed the mathematical theory of kernels. We started with the class of polynomial kernels, which can be motivated as computing a combinatorially large number of monomial features rather efficiently. This led to the general question of which kernel can be used, or: which kernel can be represented as a dot product in a linear feature space. We defined this class and discussed some of its properties. We described several ways how, given such a kernel, one can construct a representation in a feature space. The most well-known representation employs Mercer's theorem, and represents the feature space as an ℓ_2 space defined in terms of the eigenfunctions of an integral operator associated with the kernel. An alternative representation uses elements of the theory of reproducing kernel Hilbert spaces, and yields additional insights, representing the linear space as a space of functions written as kernel expansions. We gave an in-depth discussion of the kernel trick in its general form, including the case where we are interested in dissimilarities rather than similarities; that is, when we want to come up with nonlinear generalizations of distance-based algorithms rather than dot-product-based algorithms.

In both cases, the underlying philosophy is the same: we are trying to express a complex nonlinear algorithm in terms of simple geometrical concepts, and we are then dealing with it in a linear space. This linear space may not always be readily available; in some cases, it may even be hard to construct explicitly. Nevertheless, for the sake of design and analysis of the algorithms, it is sufficient to know that the linear space exists, empowering us to use the full potential of geometry, linear algebra and functional analysis.

2.6 Problems

2.1 (Monomial Features in $\mathbb{R}^2$ •) *Verify the second equality in (2.9).*

2.2 (Multiplicity of Monomial Features in $\mathbb{R}^N$ [515] ••) *Consider the monomial kernel $k(x, x') = \langle x, x' \rangle^d$ (where $x, x' \in \mathbb{R}^N$), generating monomial features of order d. Prove* *that a valid feature map for this kernel can be defined coordinate-wise as*

$$\Phi_{\mathbf{m}}(\mathbf{x}) = \sqrt{\frac{d!}{\prod_{i=1}^{n}[\mathbf{m}]_i!}}\prod_{i=1}^{n}[\mathbf{x}]_i^{[\mathbf{m}]_i} \tag{2.95}$$

for every $\mathbf{m} \in \mathbb{N}^n$, $\sum_{i=1}^{n}[\mathbf{m}]_i = d$ *(i.e., every such* $\mathbf{m}$ *corresponds to one dimension of* $\mathcal{H}$*).*

2.3 (Inhomogeneous Polynomial Kernel ●●) *Prove that the kernel (2.70) induces a feature map into the space of all monomials up to degree d. Discuss the role of c.*

2.4 (Eigenvalue Criterion of Positive Definiteness ●) *Prove that a symmetric matrix is positive definite if and only if all its eigenvalues are nonnegative (see Appendix B).*

2.5 (Dot Products are Kernels ●) *Prove that dot products (Definition B.7) are positive definite kernels.*

2.6 (Kernels on Finite Domains ●●) *Prove that for finite* $\mathcal{X}$*, say* $\mathcal{X} = \{x_1, \ldots, x_m\}$*, k is a kernel if and only if the* $m \times m$ *matrix* $(k(x_i, x_j))_{ij}$ *is positive definite.*

2.7 (Positivity on the Diagonal ●) *From Definition 2.5, prove that a kernel satisfies* $k(x, x) \geq 0$ *for all* $x \in \mathcal{X}$*.*

2.8 (Cauchy-Schwarz for Kernels ●●) *Give an elementary proof of Proposition 2.7.*

Hint: start with the general form of a symmetric 2×2 *matrix, and derive conditions for its coefficients that ensure that it is positive definite.*

2.9 (PD Kernels Vanishing on the Diagonal ●) *Use Proposition 2.7 to prove that a kernel satisfying* $k(x, x) =$ *for all* $x \in \mathcal{X}$ *is identically zero.*

How does the RKHS look in this case? Hint: use (2.31).

2.10 (Two Kinds of Positivity ●) *Give an example of a kernel which is positive definite according to Definition 2.5, but not positive in the sense that* $k(x, x') \geq 0$ *for all* x, x'*.*

Give an example of a kernel where the contrary is the case.

2.11 (General Coordinate Transformations ●) *Prove that if* $\sigma : \mathcal{X} \to \mathcal{X}$ *is a bijection, and* $k(x, x')$ *is a kernel, then* $k(\sigma(x), \sigma(x'))$ *is a kernel, too.*

2.12 (Positivity on the Diagonal ●) *Prove that positive definite kernels are positive on the diagonal,* $k(x, x) \geq 0$ *for all* $x \in \mathcal{X}$*. Hint: use* $m = 1$ *in (2.15).*

2.13 (Symmetry of Complex Kernels ●●) *Prove that complex-valued positive definite kernels are symmetric (2.18).*

2.14 (Real Kernels vs. Complex Kernels ●) *Prove that a real matrix satisfies (2.15) for all* $c_i \in \mathbb{C}$ *if and only if it is symmetric and it satisfies (2.15) for* real *coefficients* c_i*.*

Hint: decompose each c_i *in (2.15) into real and imaginary parts.*

2.15 (Rank-One Kernels •) *Prove that if f is a real-valued function on $\mathcal{X}$, then $k(x, x') := f(x)f(x')$ is a positive definite kernel.*

2.16 (Bayes Kernel ••) *Consider a binary pattern recognition problem. Specialize the last problem to the case where $f : \mathcal{X} \to \{\pm 1\}$ equals the Bayes decision function $y(x)$, i.e., the classification with minimal risk subject to an underlying distribution $P(x, y)$ generating the data.*

Argue that this kernel is particularly suitable since it renders the problem linearly separable in a 1D feature space: State a decision function (cf. (1.35)) that solves the problem (hint: you just need one parameter α, and you may set it to 1; moreover, use $b = 0$) [124].

The final part of the problem requires knowledge of Chapter 16: Consider now the situation where some prior $P(f)$ over the target function class is given. What would the optimal kernel be in this case? Discuss the connection to Gaussian processes.

2.17 (Inhomogeneous Polynomials •) *Prove that the inhomogeneous polynomial (2.70) is a positive definite kernel, e.g., by showing that it is a linear combination of homogeneous polynomial kernels with positive coefficients. What kind of features does this kernel compute [561]?*

2.18 (Normalization in Feature Space •) *Given a kernel k, construct a corresponding normalized kernel $\tilde{k}$ by normalizing the feature map $\tilde{\Phi}$ such that for all $x \in \mathcal{X}$, $\|\tilde{\Phi}(x)\| = 1$ (cf. also Definition 12.35). Discuss the relationship between normalization in input space and normalization in feature space for Gaussian kernels and homogeneous polynomial kernels.*

2.19 (Cosine Kernel •) *Suppose $\mathcal{X}$ is a dot product space, and $x, x' \in \mathcal{X}$. Prove that $k(x, x') = \cos(\angle(x, x))$ is a positive definite kernel. Hint: use Problem 2.18.*

2.20 (Alignment Kernel •) *Let $\langle K, K' \rangle_F := \sum_{ij} K_{ij} K'_{ij}$ be the Frobenius dot product of two matrices. Prove that the* empirical alignment *of two Gram matrices [124], $A(K, K') := \langle K, K' \rangle_F / \sqrt{\langle K, K \rangle_F \langle K', K' \rangle_F}$, is a positive definite kernel.*

Note that the alignment can be used for model selection, putting $K'_{ij} := y_i y_j$ (cf. Problem 2.16) and $K_{ij} := \operatorname{sgn}(k(x_i, x_j))$ or $K_{ij} := \operatorname{sgn}(k(x_i, x_j)) - b$ (cf. [124]).

2.21 (Equivalence Relations as Kernels •••) *Consider a similarity measure $k : \mathcal{X} \to \{0, 1\}$ with*

$$k(x, x) = 1 \text{ for all } x \in \mathcal{X}. \tag{2.96}$$

Prove that k is a positive definite kernel if and only if, for all $x, x', x'' \in \mathcal{X}$,

$$k(x, x') = 1 \Longleftrightarrow k(x', x) = 1 \text{ and} \tag{2.97}$$

$$k(x, x') = k(x', x'') = 1 \Longrightarrow k(x, x'') = 1. \tag{2.98}$$

Equations (2.96) to (2.98) amount to saying that $k = I_T$, where $T \subset \mathcal{X} \times \mathcal{X}$ is an equivalence relation.

As a simple example, consider an undirected graph, and let $(x, x') \in T$ whenever x and x' are in the same connected component of the graph. Show that T is an equivalence relation.

Find examples of equivalence relations that lend themselves to an interpretation as similarity measures. Discuss whether there are other relations that one might want to use as similarity measures.

2.22 (Different Feature Spaces for the Same Kernel •) *Give an example of a kernel with two valid feature maps Φ_1, Φ_2, mapping into spaces $\mathcal{H}_1, \mathcal{H}_2$ of different dimensions.*

2.23 (Converse of Mercer's Theorem •) *Prove that if an integral operator kernel k admits a uniformly convergent dot product representation on some compact set $\mathcal{X} \times \mathcal{X}$,*

$$k(x, x') = \sum_{i=1}^{\infty} \psi_i(x)\psi_i(x'), \tag{2.99}$$

then it is positive definite. Hint: show that

$$\int_{\mathcal{X}\times\mathcal{X}} \left(\sum_{i=1}^{\infty} \psi_i(x)\psi_i(x') \right) f(x) f(x')\, dx\, dx' = \sum_{i=1}^{\infty} \left(\int_{\mathcal{X}} \psi_i(x) f(x)\, dx \right)^2 \geq 0.$$

Argue that in particular, polynomial kernels (2.67) satisfy Mercer's conditions.

2.24 (∞-Norm of Mercer Eigenfunctions ••) *Prove that under the conditions of Theorem 2.10, we have, up to sets of measure zero,*

$$\sup_j \left\| \sqrt{\lambda_j}\psi_j \right\|_{\infty} \leq \sqrt{\|k\|_{\infty}} < \infty. \tag{2.100}$$

Hint: note that $\|k\|_{\infty} \geq k(x, x)$ up to sets of measures zero, and use the series expansion given in Theorem 2.10. Show, moreover, that it is not generally the case that

$$\sup_j \|\psi_j\|_{\infty} < \infty. \tag{2.101}$$

Hint: consider the case where $\mathcal{X} = \mathbb{N}$, $\mu(\{n\}) := 2^{-n}$, and $k(i, j) := \delta_{ij}$. Show that

1. *$T_k((a_j)) = (a_j 2^{-j})$ for $(a_j) \in L_2(\mathcal{X}, \mu)$,*
2. *T_k satisfies $\langle (a_j), T_k(a_j) \rangle = \sum_j (a_j 2^{-j})^2 \geq 0$ and is thus positive definite,*
3. *$\lambda_j = 2^{-j}$ and $\psi_j = 2^{j/2} e_j$ form an orthonormal eigenvector decomposition of T_k (here, e_j is the jth canonical unit vector in ℓ_2), and*
4. *$\|\psi_j\|_{\infty} = 2^{j/2} = \lambda_j^{-1/2}$.*

Argue that the last statement shows that (2.101) is wrong and (2.100) is tight.[12]

2.25 (Generalized Feature Maps •••) *Via (2.38), Mercer kernels induce compact (integral) operators. Can you generalize the idea of defining a feature map associated with an*

[12] Thanks to S. Smale and I. Steinwart for this exercise.

operator to more general bounded positive definite operators T? Hint: use the multiplication operator representation of T [467].

2.26 (Nyström Approximation (cf. [603]) •) *Consider the integral operator obtained by substituting the distribution* P *underlying the data into (2.38), i.e.,*

$$(T_k f)(x) = \int_{\mathcal{X}} k(x, x') f(x) \, d\mathrm{P}(x). \tag{2.102}$$

If the conditions of Mercer's theorem are satisfied, then k can be diagonalized as

$$k(x, x') = \sum_{j=1}^{N_{\mathcal{H}}} \lambda_j \psi_j(x) \psi_j(x'), \tag{2.103}$$

where λ_j and ψ_j satisfy the eigenvalue equation

$$\int_{\mathcal{X}} k(x, x') \psi_j(x) \, d\mathrm{P}(x) = \lambda_j \psi_j(x') \tag{2.104}$$

and the orthonormality conditions

$$\int_{\mathcal{X}} \psi_i(x) \psi_j(x) d\mathrm{P}(x) = \delta_{ij}. \tag{2.105}$$

Show that by replacing the integral by a summation over an iid sample $X = \{x_1, \ldots, x_m\}$ from $\mathrm{P}(x)$, one can recover the kernel PCA eigenvalue problem (Section 1.7). Hint: Start by evaluating (2.104) for $x' \in X$, to obtain m equations. Next, approximate the integral by a sum over the points in X, replacing $\int_{\mathcal{X}} k(x, x') \psi_j(x) \, d\mathrm{P}(x)$ by $\frac{1}{m} \sum_{n=1}^{m} k(x_n, x') \psi_j(x_n)$.

Derive the orthogonality condition for the eigenvectors $(\psi_j(x_n))_{n=1,\ldots,m}$ from (2.105).

2.27 (Lorentzian Feature Spaces ••) *If a finite number of eigenvalues is negative, the expansion in Theorem 2.10 is still valid. Show that in this case, k corresponds to a Lorentzian symmetric bilinear form in a space with indefinite signature [467].*

Discuss whether this causes problems for learning algorithms utilizing these kernels. In particular, consider the cases of SV machines (Chapter 7) and Kernel PCA (Chapter 14).

2.28 (Symmetry of Reproducing Kernels •) *Show that reproducing kernels (Definition 2.9) are symmetric. Hint: use (2.35) and exploit the symmetry of the dot product.*

2.29 (Coordinate Representation in the RKHS ••) *Write $\langle \cdot, \cdot \rangle$ as a dot product of coordinate vectors by expressing the functions of the RKHS in the basis $(\sqrt{\lambda_n} \psi_n)_{n=1,\ldots,N_{\mathcal{H}}}$, which is orthonormal with respect to $\langle \cdot, \cdot \rangle$, i.e.,*

$$f(x) = \sum_{n=1}^{N_{\mathcal{H}}} \alpha_n \sqrt{\lambda_n} \psi_n(x). \tag{2.106}$$

Obtain an expression for the coordinates α_n, using (2.47) and $\alpha_n = \left\langle f, \sqrt{\lambda_n} \psi_n \right\rangle$. Show that $\mathcal{H}$ has the structure of a RKHS in the sense that for f and g given by (2.106), and

$$g(x) = \sum_{j=1}^{N_{\mathcal{H}}} \beta_j \sqrt{\lambda_j} \psi_j(x), \tag{2.107}$$

we have $\langle \boldsymbol{\alpha}, \boldsymbol{\beta} \rangle = \langle f, g \rangle$ *. Show, moreover, that* $f(x) = \langle \boldsymbol{\alpha}, \Phi(x) \rangle$ *in* $\mathcal{H}$*. In other words,* $\Phi(x)$ *is the coordinate representation of the kernel as a function of one argument.*

2.30 (Equivalence of Regularization Terms •) *Using (2.36) and (2.41), prove that* $\|\mathbf{w}\|^2$*, where* $\mathbf{w} = \sum_{i=1}^{m} \alpha_i \Phi(x_i)$*, is the same no matter whether* Φ *denotes the RKHS feature map (2.21) or the Mercer feature map (2.40).*

2.31 (Approximate Inversion of Gram Matrices ••) *Use the kernel PCA map (2.59) to derive a method for approximately inverting a large Gram matrix.*

2.32 (Effective Dimension of Feature Space •) *Building on Section 2.2.7, argue that for a finite data set, we are always effectively working in a finite-dimensional feature space.*

2.33 (Translation of a Dot Product •) *Prove (2.79).*

2.34 (Example of a CPD Kernel ••) *Argue that the hyperbolic tangent kernel (2.69) is effectively conditionally positive definite, if the input values are suitably restricted, since it can be approximated by* $k + b$*, where* k *is a polynomial kernel (2.67) and* $b \in \mathbb{R}$*. Discuss how this explains that hyperbolic tangent kernels can be used for SVMs although, as pointed out in number of works (e.g., [86], cf. the remark following (2.69)), they are not positive definite.*

2.35 (Polarization Identity ••) *Prove the* polarization identity*, stating that for any symmetric bilinear form* $\langle \cdot, \cdot \rangle : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$*, we have, for all* $x, x' \in \mathcal{X}$*,*

$$\langle x, x' \rangle = \frac{1}{4} \left(\langle x + x', x + x' \rangle - \langle x - x', x - x' \rangle \right) . \tag{2.108}$$

Now consider the special case where $\langle \cdot, \cdot \rangle$ *is a Euclidean dot product and* $\langle x - x', x - x' \rangle$ *is the squared Euclidean distance between* x *and* x'*. Discuss why the polarization identity does not imply that the value of the dot product can be recovered from the distances alone. What else does one need?*

2.36 (Vector Space Representation of CPD Kernels •••) *Specialize the vector space representation of symmetric kernels (Proposition 2.25) to the case of cpd kernels. Can you identify a subspace on which a cpd kernel is actually pd?*

2.37 (Parzen Windows Classifiers in Feature Space ••) *Assume that* k *is a positive definite kernel. Compare the algorithm described in Section 1.2 with the one of (2.89). Construct situations where the two algorithms give different results. Hint: consider datasets where the class means coincide.*

2.38 (Canonical Distortion Kernel ○○○) *Can you define a kernel based on Baxter's canonical distortion metric [28]?*