Online learning in Reproducing Kernel Hilbert Spaces

Pantelis Bouboulis, Member, IEEE, [1]

May 1, 2012

[1]P. Bouboulis is with the Department of Informatics and telecommunications, University of Athens, Greece, e-mail: (see bouboulis.mysch.gr).

Chapter 1

Reproducing Kernel Hilbert Spaces

In kernel-based methods, the notion of the Reproducing Kernel Hilbert Space (RKHS) plays a crucial role. A RKHS is a rich construct (roughly, a space of functions with an inner product), which has been proven to be a very powerful tool. Kernel based methods are utilized in an increasingly large number of scientific areas, especially where non-linear models are required. For example, in pattern analysis, a classification task of a set $\mathcal{X} \subset \mathbb{R}^m$ is usually reformed by mapping the data into a higher dimensional space (possibly of infinite dimension) $\mathcal{H}$, which is a Reproducing Kernel Hilbert Space (RKHS). The advantage of such a mapping is to make the task more tractable, by employing a linear classifier in the feature space $\mathcal{H}$, exploiting Cover's theorem (see [43, 37]). This is equivalent with solving a non-linear problem in the original space.

Therefore, with the use of kernels, a new technique has been introduced to transform certain classes of non-linear tasks to equivalent linear ones, restated in a higher –even infinite– dimensional space, but with avoiding the accompanying computational and generalization theory's (also known as the 'curse of dimensionality') problems, associated with the "traditional" techniques, when the dimensionality of the task increases. Similar approaches have been used in principal components analysis, in Fisher's linear discriminant analysis, in clustering, regression, image processing and in many other subdisciplines. Recently, processing in RKHS is gaining in popularity within the Signal Processing community in the context of adaptive learning.

The introduction of non-linearity is usually introduced via a computationally elegant way known to the machine learning community as the *kernel trick* [36] (the formal definition of the positive definite kernel is given in section 1.2):

> "Given an algorithm, which is formulated in terms of dot products, one can construct an alternative algorithm by replacing each one of the dot products with a positive definite kernel κ."

Although this trick works well for most applications, it conceals the basic mathematical steps that underlie the procedure, which are essential if one seeks a deeper understanding of the problem. These steps are: 1) Map the finite dimensionality input data from the input space X (usually $X \subset \mathbb{R}^\nu$) into a higher dimensionality (possibly infinite) RKHS $\mathcal{H}$ (this is usually called the feature space) and 2) Perform a linear processing (e.g., adaptive filtering) on the mapped data in $\mathcal{H}$. The procedure is equivalent with a non-linear processing (non-linear filtering) in X (see figure 1.1). The specific choice of the kernel κ defines, implicitly, an RKHS with an appropriate inner product. Moreover, the specific choice of the kernel defines the type of nonlinearity that underlies the model to be used.

1.1 A Historical overview

In the past, there have been two trends in the study of these spaces by the mathematicians. The first one originated in the theory of integral equations by J. Mercer [25, 26]. He used the term "positive definite kernel" to characterize a function of two points $\kappa(x, y)$ defined on X^2, which satisfies Mercer's law:

$$\sum_{n,m=1}^{N} a_n a_m \kappa(x_n, x_m) \geq 0, \tag{1.1}$$

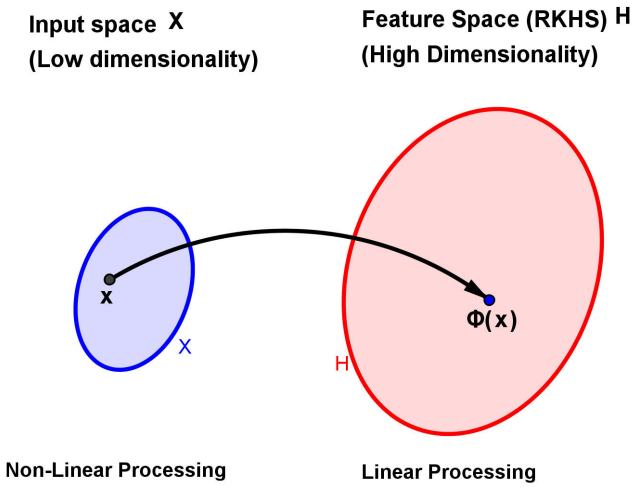


Figure 1.1: Mapping from input space X to feature space $\mathcal{H}$.

for any numbers a_n, a_m and points x_n, x_m. Later on, Moore [27, 28, 29] found that to such a kernel there corresponds a well determined class of functions, $\mathcal{H}$, equipped with a specific inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$, in respect to which the kernel κ possesses the so called reproducing property:

$$f(y) = \langle f, \kappa(\cdot, y) \rangle_{\mathcal{H}}, \tag{1.2}$$

for all functions $f \in \mathcal{H}$ and $y \in X$. Those that followed this trend used to consider a specific given positive definite kernel κ and studied it in itself, or eventually applied it in various domains (such as integral equations, theory of groups, general metric theory, interpolation, e.t.c.). The class $\mathcal{H}$ corresponding to κ was mainly used as a tool of research and it was usually introduced a posteriori. The work of Bochner [5, 6], which introduced the notion of the "positive definite function" in order to apply it in the theory of Fourier transforms, also belongs to the same path as the one followed by Mercer and Moore. These are continuous functions ϕ of one variable such that $\phi(x - y) = \kappa(x, y)$, for some positive definite kernel κ.

On the other hand, those who followed the second trend were primarily interested in the class of functions $\mathcal{H}$, while the associated kernel was employed essentially as a tool in the study of the functions of this class. This trend is traced back to the works of S. Zaremba [47, 48] during the first decade of the 20-th century. He was the first to introduce the notion of a kernel, which corresponds to a specific class of functions and to state its reproducing property. However, he did not develop any general theory, nor did he gave any particular name to the kernels he introduced. In this, second trend, the mathematicians were primarily interested in the study of the class of functions $\mathcal{H}$ and the corresponding kernel κ, which satisfies the reproducing property, was used as a tool in this study. To the same trend belong also the works of Bergman [4] and Aronszajn [2]. Those two trends evolved separately during the first decades of the 20-th century, but soon the links between them were noticed. After the second world war, it was known that the two concepts of defining a kernel, either as a positive definite kernel, or as a reproducing kernel, are equivalent. Furthermore, It was proved that there is a one to one correspondence between the space of positive definite kernels and the space of reproducing kernel Hilbert spaces.

It has to be emphasized that examples of such kernels have been known for a long time prior to the works of Mercer and Zaremba; for example, all the Green's functions of self-adjoint ordinary differential equations belong to this type of kernels. However, the some of the important properties that these kernels possess have only been realized and used in the beginning of the 20-th century and since then have been the focus of research. In the following, we will give a more detailed description of these spaces and establish their main properties, focussing on the essentials that elevate them to such a powerful tool in the context of machine learning. Most of the material presented here can also be found in more detail in several other textbooks, such as the celebrated paper of Aronszajn [2], the excellent introductory text of Paulsen [31] and

the popular books of Schölkoph and Smola [37] and Shawe-Taylor and Cristianini [39]. Here, we attempt to portray both trends and to highlight the important links between them. Although the general theory applies to complex spaces, to keep the presentation as simple as possible, we will mainly focus on real spaces. The complex case will be treated at the end of this section.

1.2 Definition

We begin our study with the classic definitions on positive definite matrices and kernels as they were introduced by Mercer. Given a function $\kappa : X \times X \to \mathbb{R}$ and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in X$ (typically X is a compact subset of $\mathbb{R}^\nu$, $\nu > 0$), the square matrix $\boldsymbol{K} = (K_{n,m})^N$ with elements $K_{n,m} = \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m)$, for $n, m = 1, \ldots, N$, is called the *Gram matrix* (or *kernel matrix*) of κ with respect to $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$. A symmetric matrix $\boldsymbol{K} = (K_{n,m})^N$ satisfying

$$\boldsymbol{c}^T \cdot K \cdot \boldsymbol{c} = \sum_{n=1,m=1}^{N} c_n c_m K_{n,m} \geq 0,$$

for all $\boldsymbol{c} \in \mathbb{R}^N$, $n = 1, \ldots, N$, where the notation $\cdot^T$ denotes the transpose matrix, is called *positive definite*. In matrix analysis literature, this is the definition of a positive semidefinite matrix. However, as positive definite matrices were originally introduced by Mercer and others in this context, we employ the term positive definite, as it was already defined. If the inequality is strict, for all non-zero vectors $\boldsymbol{c} \in \mathbb{R}^N$, the matrix will be called *strictly positive definite*. A function $\kappa : X \times X \to \mathbb{R}$, which for all $N \in \mathbb{N}$ and all $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in X$ gives rise to a positive definite Gram matrix K, is called a *positive definite kernel*. In the following, we will frequently refer to a positive definite kernel simply as *kernel*. We conclude that a positive definite kernel is symmetric and satisfies

$$\sum_{n=1,m=1}^{N} c_n c_m \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m) \geq 0,$$

for all $c \in \mathbb{R}^N$, $n = 1, \ldots, N$, and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in X$. Formally, a Reproducing kernel Hilbert space is defined as follows:

Definition 1.2.1 (Reproducing Kernel Hilbert Space)**.** *Consider a linear class $\mathcal{H}$ of real valued functions, f, defined on a set X. Suppose, further, that in $\mathcal{H}$ we can define an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ with corresponding norm $\| \cdot \|_{\mathcal{H}}$ and that $\mathcal{H}$ is complete with respect to that norm, i.e., $\mathcal{H}$ is a Hilbert space. We call $\mathcal{H}$ a Reproducing Kernel Hilbert Space (RKHS), if there exists a function $\kappa : X \times X \to \mathbb{F}$ with the following two important properties:*

1. *For every $\boldsymbol{x} \in X$, $\kappa(\cdot, \boldsymbol{x})$ belongs to $\mathcal{H}$ (or equivalently κ spans $\mathcal{H}$, i.e., $\mathcal{H} = \overline{\text{span}\{\kappa(\cdot, \boldsymbol{x}),\ \boldsymbol{x} \in X\}}$).*
2. *κ has the so called reproducing property, i.e.,*

$$f(\boldsymbol{x}) = \langle f, \kappa(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}},\ \text{for all } f \in \mathcal{H}, \boldsymbol{x} \in X, \tag{1.3}$$

in particular $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \langle \kappa(\cdot, \boldsymbol{y}), \kappa(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}}$. Furthermore, κ is a positive definite kernel and the mapping $\Phi : X \to \mathcal{H}$, with $\Phi(\boldsymbol{x}) = \kappa(\cdot, \boldsymbol{x})$, for all $\boldsymbol{x} \in X$ is called the feature map of $\mathcal{H}$.

To denote the RKHS associated with a specific kernel κ we will also use the notation $\mathcal{H}(\kappa)$. Note that $\mathcal{H}$ is often called the *feature space* associated with kernel κ. Furthermore, under the aforementioned notations $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \langle \Phi(\boldsymbol{y}), \Phi(\boldsymbol{x}) \rangle_{\mathcal{H}}$, i.e., $\kappa(\boldsymbol{x}, \boldsymbol{y})$ is the inner product of $\Phi(\boldsymbol{y})$ and $\Phi(\boldsymbol{x})$ in the feature space. This is the essence of the kernel trick mentioned at the beginning of section 1. The feature map Φ transforms the data from the low dimensionality space X to the higher dimensionality space $\mathcal{H}$. Linear processing in $\mathcal{H}$ involves inner products in $\mathcal{H}$, which can be calculated via the kernel κ disregarding the actual structure of $\mathcal{H}$. Roughly speaking, one trades nonlinearities, which is often hard to handle, for an increase in the dimensionality of the space.

1.3 Derivation of the Definition

In the following, we consider the definition of a RKHS as a class of functions with specific properties (following the second trend) and show the key ideas that underlie definition 1.2.1. To that end, consider a linear class $\mathcal{H}$ of real valued functions, f, defined on a set X. Suppose, further, that in $\mathcal{H}$ we can define an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ with corresponding norm $\| \cdot \|_{\mathcal{H}}$ and that $\mathcal{H}$ is complete with respect to that norm, i.e., $\mathcal{H}$ is a Hilbert space. Consider, also, a *linear* functional T, from $\mathcal{H}$ into the field $\mathbb{R}$. An important theorem of functional analysis states that such a functional is *continuous*, if and only if it is *bounded*. The space consisting of all continuous linear functionals from $\mathcal{H}$ into the field $\mathbb{R}$ is called the *dual space* of $\mathcal{H}$. In the following, we will frequently refer to the so called *linear evaluation functional* $T_{\boldsymbol{y}}$. This is a special case of a linear functional that satisfies $T_{\boldsymbol{y}}(f) = f(\boldsymbol{y})$, for all $f \in \mathcal{H}$.

We call $\mathcal{H}$ a *Reproducing Kernel Hilbert Space (RKHS)* on X over $\mathbb{R}$, if for *every* $\boldsymbol{y} \in X$, the linear evaluation functional, $T_{\boldsymbol{y}}$, is continuous. We will prove that such a space is related to a positive definite kernel, thus providing the first link between the two trends. Subsequently, we will prove that any positive definite kernel defines implicitly a RKHS, providing the second link and concluding the equivalent definition of RKHS (definition 1.2.1), which is usually used in the machine learning literature. The following theorem establishes an important connection between a Hilbert space H and its dual space.

Theorem 1.3.1 (Riesz Representation)**.** *Let $\mathcal{H}$ be a general Hilbert space and let $\mathcal{H}^*$ denote its dual space. Every element Φ of $\mathcal{H}^*$ can be uniquely expressed in the form:*

$$\Phi(f) = \langle f, \phi \rangle_{\mathcal{H}},$$

for some $\phi \in \mathcal{H}$. Moreover, $\|\Phi\|_{\mathcal{H}^*} = \|\phi\|_{\mathcal{H}}$.

Following the Riesz representation theorem, we have that for *every* $\boldsymbol{y} \in X$, there exists a *unique* element $\kappa_{\boldsymbol{y}} \in \mathcal{H}$, such that for every $f \in \mathcal{H}$, $f(\boldsymbol{y}) = T_{\boldsymbol{y}}(f) = \langle f, \kappa_{\boldsymbol{y}} \rangle_{\mathcal{H}}$. The function $\kappa_{\boldsymbol{y}}$ is called the reproducing kernel for the point $\boldsymbol{y}$ and the function $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \kappa_{\boldsymbol{y}}(\boldsymbol{x})$ is called the *reproducing kernel* of $\mathcal{H}$. In addition, note that $\langle \kappa_{\boldsymbol{y}}, \kappa_{\boldsymbol{x}} \rangle_{\mathcal{H}} = \kappa_{\boldsymbol{y}}(\boldsymbol{x}) = \kappa(\boldsymbol{x}, \boldsymbol{y})$ and $\|T_{\boldsymbol{y}}\|^2_{\mathcal{H}^*} = \|\kappa_{\boldsymbol{y}}\|^2_{\mathcal{H}} = \langle \kappa_{\boldsymbol{y}}, \kappa_{\boldsymbol{y}} \rangle_{\mathcal{H}} = \kappa(\boldsymbol{y}, \boldsymbol{y})$.

Proposition 1.3.1. *The reproducing kernel of $\mathcal{H}$ is symmetric, i.e., $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \kappa(\boldsymbol{y}, \boldsymbol{x})$.*

Proof. Observe that $\langle \kappa_{\boldsymbol{y}}, \kappa_{\boldsymbol{x}} \rangle_{\mathcal{H}} = \kappa_{\boldsymbol{y}}(\boldsymbol{x}) = \kappa(\boldsymbol{x}, \boldsymbol{y})$ and $\langle \kappa_{\boldsymbol{x}}, \kappa_{\boldsymbol{y}} \rangle_{\mathcal{H}} = \kappa_{\boldsymbol{x}}(\boldsymbol{y}) = \kappa(\boldsymbol{y}, \boldsymbol{x})$. As the inner product of $\mathcal{H}$ is symmetric (i.e., $\langle \kappa_{\boldsymbol{y}}, \kappa_{\boldsymbol{x}} \rangle_{\mathcal{H}} = \langle \kappa_{\boldsymbol{x}}, \kappa_{\boldsymbol{y}} \rangle_{\mathcal{H}}$) the result follows. □

In the following, we will frequently identify the function $\kappa_{\boldsymbol{y}}$ with the notation $\kappa(\cdot, \boldsymbol{y})$. Thus, we write the reproducing property of $\mathcal{H}$ as:

$$f(\boldsymbol{y}) = \langle f, \kappa(\cdot, \boldsymbol{y}) \rangle_{\mathcal{H}}, \tag{1.4}$$

for any $f \in \mathcal{H}$, $\boldsymbol{y} \in X$. Note that due to the uniqueness provided by the Riesz representation theorem, κ is the unique function that satisfies the reproducing property. The following proposition establishes the first link between the positive definite kernels and the reproducing kernels.

Proposition 1.3.2. *The reproducing kernel of $\mathcal{H}$ is a positive definite kernel.*

Proof. Consider $N > 0$, the real numbers $a_1, a_2, \ldots a_N$ and the elements, $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N \in X$. Then

$$\begin{aligned}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m) &= \sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m \langle \kappa(\cdot, \boldsymbol{x}_m), \kappa(\cdot, \boldsymbol{x}_n) \rangle_{\mathcal{H}} = \sum_{n=1}^{N} a_n \left\langle \sum_{m=1}^{N} a_m \kappa(\cdot, \boldsymbol{x}_m), \kappa(\cdot, \boldsymbol{x}_n) \right\rangle_{\mathcal{H}} \\ &= \left\langle \sum_{m=1}^{N} a_m \kappa(\cdot, \boldsymbol{x}_m), \sum_{n=1}^{N} a_n \kappa(\cdot, \boldsymbol{x}_n) \right\rangle_{\mathcal{H}} = \left\| \sum_{n=1}^{N} a_n \kappa(\cdot, \boldsymbol{x}_n) \right\|^2_{\mathcal{H}} \geq 0.\end{aligned}$$

Combining proposition 1.3.1 and the previous result, we complete the proof. □

Remark 1.3.1. *Generally, for a reproducing kernel, the respective Gram matrix is strictly positive definite. For if not, then there must exist at least one non zero vector* $\boldsymbol{a}$ *such that* $\left\|\sum_{n=1}^{N} a_n \kappa(\cdot, \boldsymbol{x}_n)\right\|_{\mathcal{H}}^2 = 0$. *Hence, for every* $f \in \mathcal{H}$ *we have that* $\sum_n a_n f(\boldsymbol{x}_n) = \langle f, \sum_n a_n \kappa(\cdot, \boldsymbol{x}_n)\rangle_{\mathcal{H}} = 0$. *Thus, in this case there is an equation of linear dependence between the values of every function in* $\mathcal{H}$ *at some finite set of points. Such examples do exist (e.g. Sobolev spaces), but in most cases the reproducing kernels define Gram matrices that are always strictly positive and invertible!*

The following proposition establishes a very important fact; any RKHS, $\mathcal{H}$, can be generated by the respective reproducing kernel κ. Note that the overbar denotes the closure of a set (i.e., if A is a subset of $\mathcal{H}$, $\bar{A}$ is the closure of A).

Proposition 1.3.3. *Let* $\mathcal{H}$ *be a RKHS on the set* X *with reproducing kernel* κ. *Then the linear span of the functions* $\kappa(\cdot, \boldsymbol{x})$, $x \in X$ *is dense in* $\mathcal{H}$, *i.e.,* $\mathcal{H} = \overline{\text{span}\{\kappa(\cdot, \boldsymbol{x}),\ \boldsymbol{x} \in X\}}$.

Proof. We will prove that the only function of $\mathcal{H}$ orthogonal to $A = \text{span}\{\kappa(\cdot, \boldsymbol{x}),\ x \in X\}$ is the zero function. Let f be such a function. Then, as f is orthogonal to A, we have that $f(\boldsymbol{x}) = \langle f, \kappa(\cdot, \boldsymbol{x})\rangle_{\mathcal{H}} = 0$, for every $\boldsymbol{x} \in X$. This holds true if and only if $f = 0$. Thus $A^{\perp} = \overline{A^{\perp}} = \{0\}$. Suppose that there is $f \in \mathcal{H}$ such that $f \notin \overline{A}$. As $\overline{A}$ is a closed (convex) subspace of $\mathcal{H}$, there is a $g \in \overline{A}$ which minimizes the distance between f and points in $\overline{A}$ (theorem of best approximation). For the same g we have that $f - g \perp \overline{A}$. Thus, the non-zero function $h = f - g$ is orthogonal to $\overline{A}$. However, we proved that there isn't any non-zero vector orthogonal to A. This leads us to conclude that $\overline{A} = \mathcal{H}$. □

In the following we give some important properties of the specific spaces.

Proposition 1.3.4 (Norm convergence implies point-wise convergence)**.** *Let* $\mathcal{H}$ *be a RKHS on* X *and let* $\{f_n\}_{n\in\mathbb{N}} \subseteq \mathcal{H}$. *If* $\lim_n \|f_n - f\|_{\mathcal{H}} = 0$, *then* $f(\boldsymbol{x}) = \lim_n f_n(\boldsymbol{x})$, *for every* $\boldsymbol{x} \in X$. *Conversely, if for any sequence* $\{f_n\}_{n\in\mathbb{N}}$ *of a Hilbert space* $\mathcal{H}$, *such that* $\lim_n \|f_n - f\|_{\mathcal{H}} = 0$ *we have also that* $f(\boldsymbol{x}) = \lim_n f_n(\boldsymbol{x})$, *then* $\mathcal{H}$ *is a RKHS.*

Proof. For every $\boldsymbol{x} \in X$ we have that

$$|f_n(\boldsymbol{x}) - f(\boldsymbol{x})|_{\mathcal{H}} = |\langle f_n, \kappa(\cdot, \boldsymbol{x})\rangle_{\mathcal{H}} - \langle f, \kappa(\cdot, \boldsymbol{x})\rangle_{\mathcal{H}}| = |\langle f_n - f, \kappa(\cdot, \boldsymbol{x})\rangle_{\mathcal{H}}| \leq \|f_n - f\|_{\mathcal{H}} \cdot \|\kappa(\cdot, \boldsymbol{x})\|_{\mathcal{H}}.$$

As $\lim_n \|f_n - f\| = 0$, we have that $\lim_n |f_n(\boldsymbol{x}) - f(\boldsymbol{x})| = 0$, for every $\boldsymbol{x} \in X$. Hence $f(\boldsymbol{x}) = \lim_n f_n(\boldsymbol{x})$, for every $\boldsymbol{x} \in X$.

For the converse, consider the evaluation functional $T_{\boldsymbol{y}} : \mathcal{H} \to \mathbb{R}$, $T_{\boldsymbol{y}}(f) = f(\boldsymbol{y})$ for some $\boldsymbol{y} \in \mathcal{H}$. We will prove that $T_{\boldsymbol{y}}$ is continuous for all $\boldsymbol{y} \in \mathcal{H}$. To this end, consider a sequence $\{f_n\}_{n\in\mathbb{N}}$ of $\mathcal{H}$, with the property $\lim_n \|f_n - f\|_{\mathcal{H}} = 0$, i.e., f_n converges to f in the norm. Then $|T_{\boldsymbol{y}}(f_n) - T_{\boldsymbol{y}}(f)| = |f_n(\boldsymbol{y}) - f(\boldsymbol{y})| \to 0$, as $f(\boldsymbol{x}) = \lim_n f_n(\boldsymbol{x})$. Thus $T_{\boldsymbol{y}}(f) = \lim_n T_{\boldsymbol{y}}(f_n)$ for all $\boldsymbol{y} \in X$ and all converging sequences $\{f_n\}_{n\in\mathbb{N}}$ of $\mathcal{H}$. □

Proposition 1.3.5 (Different RKHS's cannot have the same reproducing kernel)**.** *Let* $\mathcal{H}_1, \mathcal{H}_2$ *be RKHS's on* X *with reproducing kernels* κ_1, κ_2. *If* $\kappa_1(\boldsymbol{x}, \boldsymbol{y}) = \kappa_2(\boldsymbol{x}, \boldsymbol{y})$, *for all* $\boldsymbol{x}, \boldsymbol{y} \in X$, *then* $\mathcal{H}_1 = \mathcal{H}_2$ *and* $\|f\|_{\mathcal{H}_1} = \|f\|_{\mathcal{H}_2}$ *for every* f.

Proof. Let $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \kappa_1(\boldsymbol{x}, \boldsymbol{y}) = \kappa_2(\boldsymbol{x}, \boldsymbol{y})$ and $A_i = \text{span}\{\kappa_i(\cdot, \boldsymbol{x}), \boldsymbol{x} \in X\}$, $i = 1, 2$. As shown in proposition 1.3.3, $\mathcal{H}_i = \overline{A_i}$, $i = 1, 2$. Note that for any $f \in A_i$, $i = 1, 2$, we have that $f(\boldsymbol{x}) = \sum_n a_n \kappa_i(\cdot, \boldsymbol{x}_n)$, for some real numbers a_n and thus the values of the function are independent of whether we regard it as in A_1 or A_2. Furthermore, for any $f \in A_i$, $i = 1, 2$, as the two kernels are identical, we have that $\|f\|_{\mathcal{H}_1}^2 = \sum_{n,m} a_n a_m \kappa(\boldsymbol{x}_m, \boldsymbol{x}_n) = \|f\|_{\mathcal{H}_2}^2$. Thus, $\|f\|_{\mathcal{H}_1} = \|f\|_{\mathcal{H}_2}$, for all $f \in A_1 = A_2$.

Finally, we turn our attention to the limit points of A_1 and A_2. If $f \in \mathcal{H}_1$, then there exists a sequence of functions, $\{f_n\}_{n\in\mathbb{N}} \subseteq A_1$ such that $\lim_n \|f - f_n\|_{\mathcal{H}_1} = 0$. Since $\{f_n\}_{n\in\mathbb{N}}$ is a converging sequence, it is Cauchy in A_1 and thus it is also Cauchy in A_2. Therefore, there exists $g \in \mathcal{H}_2$ such that $\lim_n \|g - f_n\|_{\mathcal{H}_2} = 0$. Employing proposition 1.3.4, we take that $f(\boldsymbol{x}) = \lim_n f_n(\boldsymbol{x}) = g(\boldsymbol{x})$. Thus, every f in $\mathcal{H}_1$ is also in $\mathcal{H}_2$ and by analogous argument we can prove that every $g \in \mathcal{H}_2$ is also in $\mathcal{H}_1$. Hence $\mathcal{H}_1 = \mathcal{H}_2$ and as $\|f\|_{\mathcal{H}_1} = \|f\|_{\mathcal{H}_2}$ for all f in a dense subset (i.e., A_1), we have that the norms are equal for every f. To prove the latter, we use the relation $\lim_n \|f_n\|_{\mathcal{H}_i} = \|f\|_{\mathcal{H}_i}$, $i = 1, 2$. □

The following theorem is the converse of proposition 1.3.2. It was proved by Moore and it gives us a characterization of reproducing kernel functions. Also, it provides the second link between the two trends that have been mentioned in section 1.1. Moore's theorem, together with proposition 1.3.2, proposition 1.3.5 and the uniqueness property of the reproducing kernel of a RKHS, establishes a one-to-one correspondence between RKHS's on a set and positive definite functions on the set.

Theorem 1.3.2 (Moore)**.** *Let X be a set and let $\kappa : X \times X \to \mathbb{R}$ be a positive definite kernel. Then there exists a RKHS of functions on X, such that κ is the reproducing kernel of $\mathcal{H}$.*

Proof. We will give only a sketch of the proof. The interested reader is referred to [31]. The first step is to define $A = \text{span}\{\kappa(\cdot, \boldsymbol{x}),\ \boldsymbol{x} \in X\}$ and the linear map $P : A \times A \to \mathbb{R}$ such that

$$P\left(\sum_m a_m \kappa(\cdot, \boldsymbol{y}_m), \sum_n b_n \kappa(\cdot, \boldsymbol{y}_n)\right) = \sum_{n,m} a_m b_n \kappa(\boldsymbol{y}_n, \boldsymbol{y}_m).$$

We prove that P is well defined and that it satisfies the properties of the inner product. Then, given the vector space A and the inner product P, one may complete the space by taking equivalence classes of Cauchy sequences from A to obtain the Hilbert space $\overline{A}$. Finally, the reproducing property of the kernel κ with respect to the inner product P is proved. □

In view of the aforementioned theorems, the definition 1.2.1 of the RKHS given in 1.2, which is usually used in the machine learning literature, follows naturally.

We conclude this section with a short description of the most important points of the theory developed by Mercer in the context of integral operators. Mercer considered integral operators T_κ generated by a kernel κ, i.e., $T_\kappa : L_2(X) \to L_2(X)$, such that $(T_\kappa f)(\boldsymbol{x}) := \int_X \kappa(\boldsymbol{x}, \boldsymbol{y}) f(\boldsymbol{y}) d\boldsymbol{y}$. He concluded the following theorems [25]:

Theorem 1.3.3 (Mercer Kernels are positive definite)**.** *Let $X \subseteq \mathbb{R}^\nu$ be a nonempty set and let $\kappa : X \times X \to \mathbb{R}$ be continuous. Then κ is a positive definite kernel if and only if*

$$\int_a^b \int_a^b f(\boldsymbol{x}) \kappa(\boldsymbol{x}, \boldsymbol{y}) f(\boldsymbol{y}) d\boldsymbol{x} d\boldsymbol{y} \geq 0,$$

for all continuous functions f on X. Moreover, if κ is positive definite, the integral operator $T_\kappa : L_2(X) \to L_2(X) : (T_\kappa f)(\boldsymbol{x}) := \int_X \kappa(\boldsymbol{x}, \boldsymbol{y}) f(\boldsymbol{y}) d\boldsymbol{y}$ is positive definite and if $\psi_i \in L_2(X)$ are the normalized orthogonal eigenfunctions of T_k associated with the eigenvalues $\lambda_i > 0$ then:

$$\kappa(\boldsymbol{x}, \boldsymbol{y}) = \sum_i \lambda_i \psi_i(\boldsymbol{x}) \psi_i(\boldsymbol{y}).$$

Note that the original form of above theorem is more general, involving σ-algebras and probability measures. However, as in the applications concerning this manuscript such general terms are of no importance, we decided to include this simpler form. The previous theorems established that Mercer's kernels, as they are positive definite kernels, are also reproducing kernels. Furthermore, the first part of theorem 1.3.3 provides a useful tool of determining whether a specific function is actually a reproducing kernel.

Before closing this section, we should emphasize that the general theory of RKHS has been developed by the mathematicians to treat complex spaces. However, for the sake of simplicity and clarity, we decided to begin with the simplest real case. Besides, most kernel based methods involve real data sets. Nevertheless, keep in mind that all the theorems presented here can be generalized to treat complex spaces. We will explore this issue further in section 1.8.

1.4 Examples of Kernels

Before proceeding to some more advanced topics in the theory of RKHS, it is important to give some examples of kernels that appear more often in the literature and are used in various applications. Perhaps

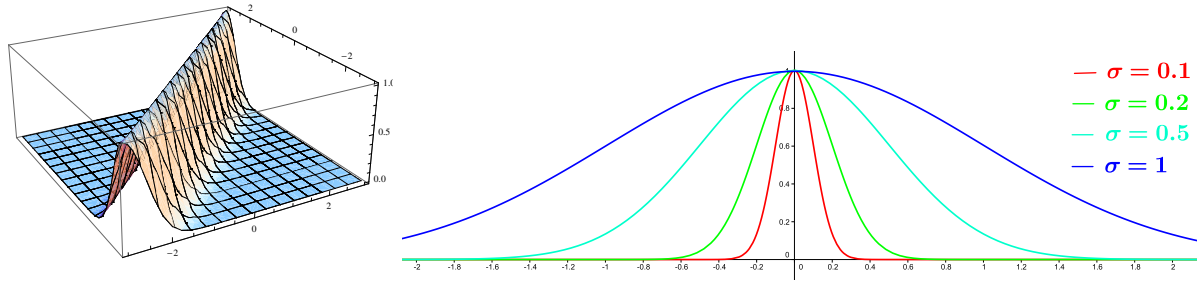
the most widely used reproducing kernel is the Gaussian radial basis function defined on $X \times X$, where $X \subseteq \mathbb{R}^\nu$, as:

$$\kappa_\sigma(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{y}\|^2}{2\sigma^2}\right), \tag{1.5}$$

where $\sigma > 0$. Equivalently the Gaussian RBF function can be defined as:

$$\kappa_t(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-t\|\boldsymbol{x} - \boldsymbol{y}\|^2\right), \tag{1.6}$$

for $t > 0$.



(a) (b)

Figure 1.2: (a) The Gaussian kernel for the case $X = \mathbb{R}$, $\sigma = 0.5$. (b) The element $\Phi(0) = \kappa(\cdot, 0)$ of the feature space induced by the Gaussian kernel for various values of the parameter σ.

Other well-known kernels defined in $X \times X$, $X \subseteq \mathbb{R}^\nu$ are:

- The homogeneous polynomial kernel: $\kappa_d(\boldsymbol{x}, \boldsymbol{y}) = \langle \boldsymbol{x}, \boldsymbol{y} \rangle^d$.
- The inhomogeneous polynomial kernel: $\kappa_d(\boldsymbol{x}, \boldsymbol{y}) = (\langle \boldsymbol{x}, \boldsymbol{y} \rangle + c)^d$, where $c \geq 0$ a constant.
- The spline kernel: $\kappa_p(\boldsymbol{x}, \boldsymbol{y}) = B_{2p+1}(\|\boldsymbol{x} - \boldsymbol{y}\|^2)$, where $B_n = \bigoplus_{i=1}^n I_{[-\frac{1}{2}, \frac{1}{2}]}$.
- The cosine kernel: $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \cos(\angle(\boldsymbol{x}, \boldsymbol{y}))$.
- The Laplacian kernel: $\kappa_t(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-t\|\boldsymbol{x} - \boldsymbol{y}\|\right)$.

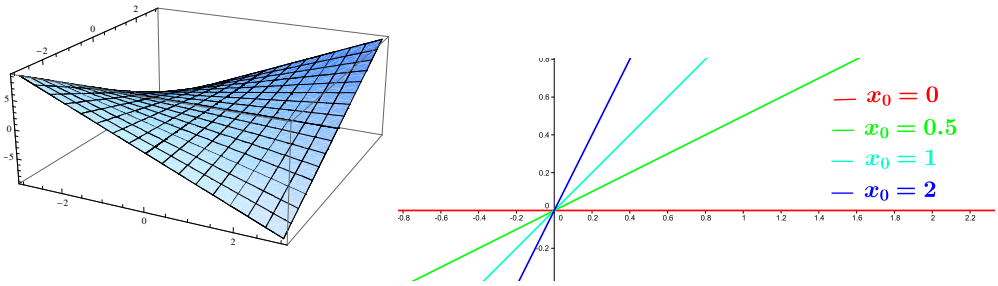
Figures 1.2, 1.3, 1.4, 1.5, 1.6, show some of the aforementioned kernels together with a sample of the elements $\kappa(\cdot, \boldsymbol{x})$ that span the respective RKHS's for the case $X = \mathbb{R}$. Figures 1.7, 1.8, 1.9, show some of the elements $\kappa(\cdot, \boldsymbol{x})$ that span the respective RKHS's for the case $X = \mathbb{R}^2$. Interactive figures regarding the aforementioned examples can be found in `http://bouboulis.mysch.gr/kernels.html`.

1.5 Properties of RKHS

In this section, we will refer to some more advanced topics on the theory of RKHS, which are useful for a deeper understanding of the underlying theory and show why RKHS's constitute such a powerful tool. We begin our study with some properties of RKHS's and conclude with the basic theorems that enable us to generate new kernels. As we work in Hilbert spaces, the two *Parseval's identities* are an extremely helpful tool. When $\{e_s : s \in S\}$ (where S is an arbitrary set) is an orthonormal basis for a Hilbert space H, then for any $h \in H$ we have that:

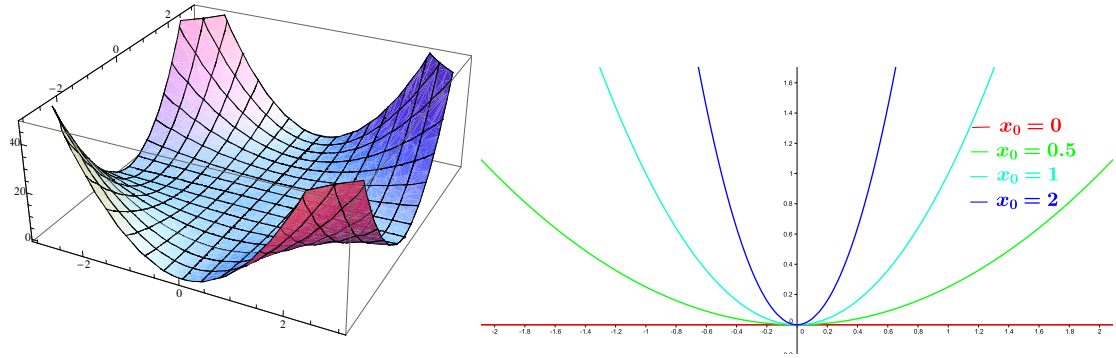
$$h = \sum_{s \in S} \langle h, e_s \rangle e_s, \tag{1.7}$$

$$\|h\|^2 = \sum_{s \in S} |\langle h, e_s \rangle|^2. \tag{1.8}$$



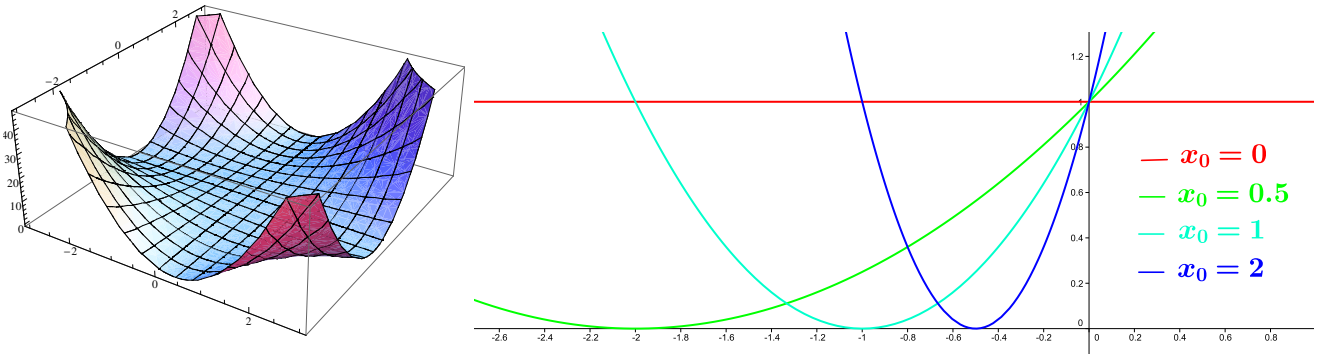
(a) (b)

Figure 1.3: (a) The homogeneous polynomial kernel for the case $X = \mathbb{R}$, $d = 1$. (b) The element $\Phi(x_0) = \kappa(\cdot, x_0)$ of the feature space induced by the homogeneous polynomial kernel ($d = 1$) for various values of x_0.



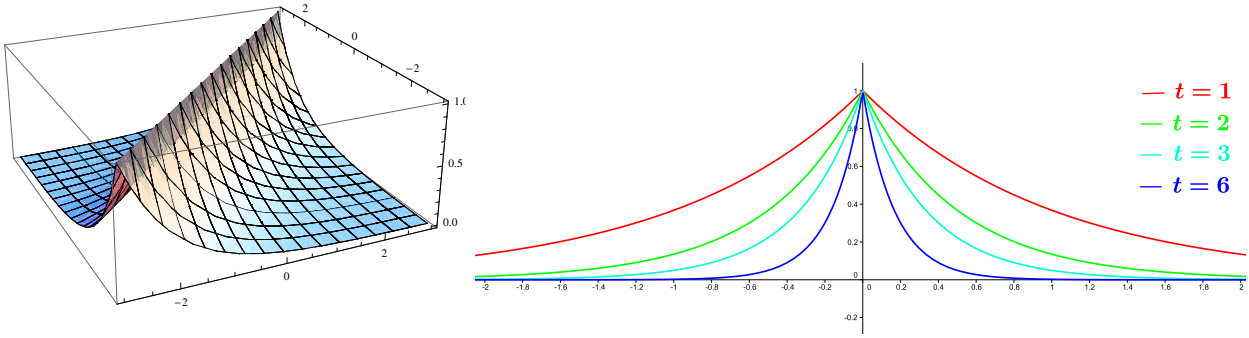
(a) (b)

Figure 1.4: (a) The homogeneous polynomial kernel for the case $X = \mathbb{R}$, $d = 2$. (b) The element $\Phi(x_0) = \kappa(\cdot, x_0)$ of the feature space induced by the homogeneous polynomial kernel ($d = 2$) for various values of x_0.



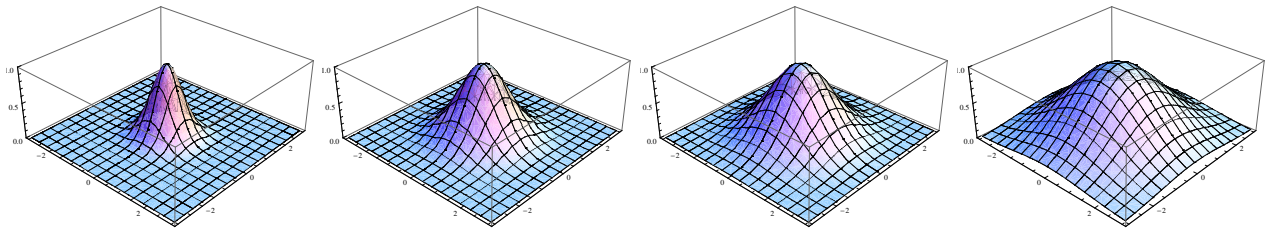
(a) (b)

Figure 1.5: (a) The inhomogeneous polynomial kernel for the case $X = \mathbb{R}$, $d = 2$. (b) The element $\Phi(x_0) = \kappa(\cdot, x_0)$ of the feature space induced by the inhomogeneous polynomial kernel ($d = 2$) for various values of x_0.



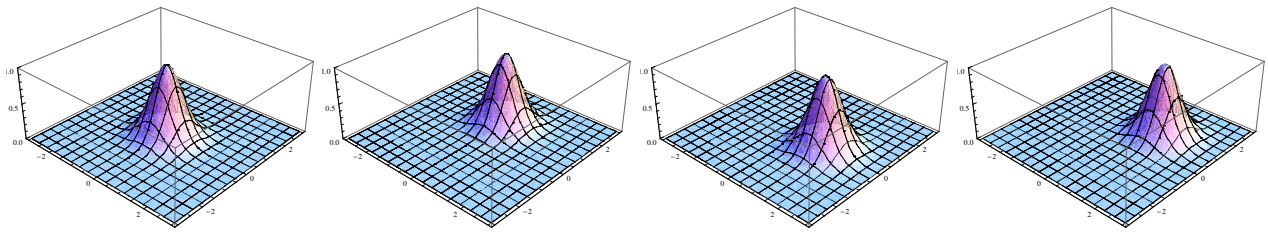
(a) (b)

Figure 1.6: (a) The Laplacian kernel for the case $X = \mathbb{R}$, $t = 1$. (b) The element $\Phi(0) = \kappa(\cdot, 0)$ of the feature space induced by the Laplacian kernel for various values of the parameter t.



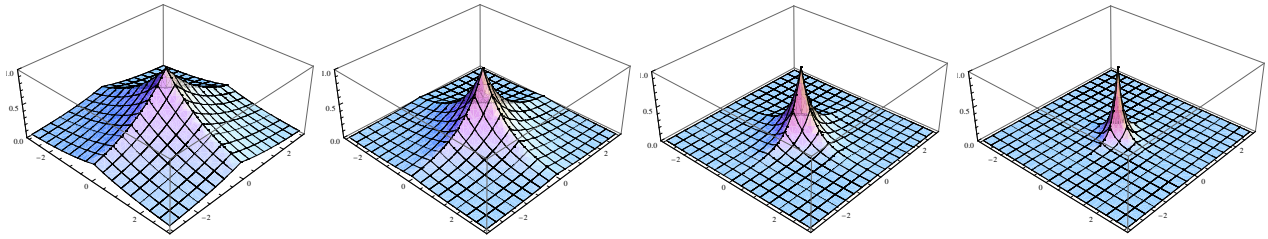
(a) (b) (c) (d)

Figure 1.7: The element $\Phi(0) = \kappa(\cdot, 0)$ of the feature space induced by the Gaussian kernel ($X = \mathbb{R}^2$) for various values of the parameter σ. (a) $\sigma = 0.5$, (b) $\sigma = 0.8$, (c) $\sigma = 1$, (d) $\sigma = 1.5$.



(a) (b) (c) (d)

Figure 1.8: The element $\Phi(x_0) = \kappa(\cdot, x_0)$ of the feature space induced by the Gaussian kernel ($X = \mathbb{R}^2$) with $\sigma = 0.5$. (a) $x_0 = (0, 0)^T$, (b) $x_0 = (0, 1)^T$, (c) $x_0 = (1, 0)^T$, (d) $x_0 = (1, 1)^T$.



(a) (b) (c) (d)

Figure 1.9: The element $\Phi(0) = \kappa(\cdot, 0)$ of the feature space induced by the Laplacian kernel ($X = \mathbb{R}^2$) for various values of the parameter t. (a) $t = 0.5$, (b) $t = 1$, (c) $t = 2$, (d) $t = 4$.

Note that these two identities hold for a general arbitrary set S (not necessarily ordered). The convergence in this case is defined somewhat differently. We say that $h = \sum_{s\in S} h_s$, if for any $\epsilon > 0$, there exists a finite subset $F_0 \subseteq F$, such that for any finite set F: $F_0 \subseteq F \subseteq S$, we have that $\|h - \sum_{s\in S} h_s\| < \epsilon$.

Proposition 1.5.1 (Cauchy-Schwarz Inequality)**.** *If κ is a reproducing kernel on X then*

$$\|\kappa(\boldsymbol{x}, \boldsymbol{y})\|^2 \leq \kappa(\boldsymbol{x}, \boldsymbol{x}) \cdot \kappa(\boldsymbol{y}, \boldsymbol{y}).$$

Proof. The proof is straightforward, as $\kappa(\boldsymbol{x}, \boldsymbol{y})$ is the inner product $\langle \Phi(\boldsymbol{y}), \Phi(\boldsymbol{x})\rangle_{\mathcal{H}}$ of the space $\mathcal{H}(\kappa)$. □

Theorem 1.5.1. *Every finite dimensional class of functions defined on X, equipped with an inner product, is a RKHS. Let $h_1, \ldots, h_N$ constitute a basis of the space and the inner product is defined as follows*

$$\langle f, g\rangle = \sum_{n,m=1}^{N} \alpha_{n,m}\gamma_n\zeta_m,$$

for $f = \sum_{n=1}^{N} \gamma_n h_n$ and $g = \sum_{n=1}^{N} \zeta_n h_n$. Let $A = (\alpha_{n,m})^N$, and $B = (\beta_{n,m})^N$ be its inverse. Then the kernel of the RKHS is given by

$$\kappa(\boldsymbol{x}, \boldsymbol{y}) = \sum_{n,m=1}^{N} \beta_{n,m} h_n(\boldsymbol{x}) h_m(\boldsymbol{y}), \tag{1.9}$$

Proof. The reproducing property is immediately verified by equation 1.9:

$$\begin{aligned}\langle f, \kappa(\cdot, \boldsymbol{x})\rangle_{\mathcal{H}} &= \left\langle \sum_{n=1}^{N} \gamma_n h_n, \sum_{n,m=1}^{N} \beta_{n,m} h_m(\boldsymbol{x}) \cdot h_m \right\rangle_{\mathcal{H}} = \sum_{n,m=1}^{N} \alpha_{n,m}\gamma_n \sum_{k=1}^{N} \beta_{m,k} h_k(\boldsymbol{x}) \\ &= \sum_{n,k=1}^{N} \left(\sum_{m=1}^{N} \alpha_{n,m}\beta_{m,k} \right) \gamma_n h_k(\boldsymbol{x}) = \sum_{n=1}^{N} \gamma_n h_n(\boldsymbol{x}) \\ &= f(x).\end{aligned}$$

□

The following theorem gives the kernel of a RKHS (of finite or infinite dimension) in terms of the elements of an orthonormal basis.

Theorem 1.5.2. *Let $\mathcal{H}$ be a RKHS on X with reproducing kernel κ. If $\{e_s : s \in S \subset \mathbb{N}\}$ is an orthonormal basis for $\mathcal{H}$, then $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \sum_{s\in S} e_s(\boldsymbol{y}) e_s(\boldsymbol{x})$, where this series converges pointwise.*

Proof. For any $\boldsymbol{y} \in X$ we have that $\langle \kappa(\cdot, \boldsymbol{y}), e_s\rangle_{\mathcal{H}} = \langle e_s, \kappa(\cdot, \boldsymbol{y})\rangle_{\mathcal{H}} = e_s(\boldsymbol{y})$. Hence, employing Parseval's identity (1.7), we have that $\kappa(\cdot, \boldsymbol{y}) = \sum_{s\in S} e_s(\boldsymbol{y}) e_s(\cdot)$, where these sums converge in the norm on $\mathcal{H}$. Since the sums converge in the norm, they converge at every point. Hence, $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \sum_{s\in S} e_s(\boldsymbol{y}) e_s(\boldsymbol{x})$. □

Proposition 1.5.2. *If $\mathcal{H}$ is a RKHS on X with respective kernel κ then every closed subspace $\mathcal{F} \subseteq \mathcal{H}$ is also a RKHS. In addition, if $\mathcal{F}_1(\kappa_1)$ and $\mathcal{F}_2(\kappa_2)$ are complementary subspaces of $\mathcal{H}$ then $\kappa = \kappa_1 + \kappa_2$.*

Proposition 1.5.3. *Let $\mathcal{H}$ be a RKHS on X with kernel κ and $\{g_n\}$ is an orthonormal system in $\mathcal{H}$. Then for any sequence of numbers $\{a_n\}$ such that $\sum_n a_n^2 < \infty$ (i.e., $\{a_n\} \in \ell_2$) we have*

$$\sum_n |a_n||g_n(\boldsymbol{x})| \leq \kappa(\boldsymbol{x}, \boldsymbol{x})^{\frac{1}{2}} \left(\sum_n |a_n|^2 \right)^{\frac{1}{2}}.$$

Proof. We have seen that $g_n(\boldsymbol{y}) = \langle g_n, \kappa(\cdot, \boldsymbol{y})\rangle$ and that $\|\kappa(\cdot, \boldsymbol{y})\|^2_{\mathcal{H}} = \kappa(\boldsymbol{y}, \boldsymbol{y})$. Thus, considering that g_n's are orthonormal and taking the Parseval's identity (1.8) for $\kappa(\cdot, \boldsymbol{y})$ with respect to the orthonormal basis we have:

$$\sum_n |g_n(\boldsymbol{y})|^2 = \sum_n |\langle g_n, \kappa(\cdot, \boldsymbol{y})\rangle_{\mathcal{H}}|^2 = \|\kappa(\cdot, \boldsymbol{y})\|^2_{\mathcal{H}} = \kappa(\boldsymbol{y}, \boldsymbol{y}).$$

Therefore, applying the Cauchy-Schwartz inequality we take

$$\sum_n |a_n||g_n(\boldsymbol{x})| \leq \left(\sum_n |a_n|^2\right)^{\frac{1}{2}} \left(\sum_n |g_n(\boldsymbol{x})|^2\right)^{\frac{1}{2}} \leq \kappa(\boldsymbol{x}, \boldsymbol{x})^{\frac{1}{2}} \left(\sum_n |a_n|^2\right)^{\frac{1}{2}}.$$

□

Theorem 1.5.3 (Representer Theorem)**.** *Denote by $\Omega : [0, +\infty) \to \mathbb{R}$ a strictly monotonic increasing function, by X a nonempty set and by $\mathcal{L} : X \times \mathbb{R}^2 \to \mathbb{R} \cup \{\infty\}$ an arbitrary loss function. Then each minimizer $f \in \mathcal{H}$ of the regularized minimization problem:*

$$\min_f \mathcal{L}((\boldsymbol{x}_1, y_1, f(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_N, y_N, f(\boldsymbol{x}_N)) + \Omega(\|f\|^2_{\mathcal{H}}),$$

admits a representation of the form $f = \sum_{n=1}^N a_n \kappa(\cdot, \boldsymbol{x}_n)$.

Proof. We may decompose each $f \in \mathcal{H}$ into a part contained in the span of the kernels centered at the training points, i.e., $\kappa(\cdot, \boldsymbol{x}_1), \ldots, \kappa(\cdot, \boldsymbol{x}_N)$, (which is a closed linear subspace) and a part in the orthogonal complement of the previous span. Thus each f can be written as:

$$f = \sum_{n=1}^N a_n \kappa(\cdot, \boldsymbol{x}_n) + f_\perp.$$

Applying the reproducing property and considering that $\langle f_\perp, \kappa(\cdot, \boldsymbol{x}_n)\rangle_{\mathcal{H}} = 0$, for $n = 1, \ldots, N$, we take:

$$f(\boldsymbol{x}_n) = \langle f, \kappa(\cdot, \boldsymbol{x}_n)\rangle_{\mathcal{H}} = \sum_{i=1}^N a_i \kappa(\boldsymbol{x}_n, \boldsymbol{x}_i) + \langle f_\perp, \kappa(\cdot, \boldsymbol{x}_n)\rangle_{\mathcal{H}} = \sum_{i=1}^N a_i \kappa(\boldsymbol{x}_n, \boldsymbol{x}_i).$$

Thus, the value of the loss function $\mathcal{L}$ depends only on the part contained in the span of the kernels centered at the training points, i.e., on $a_1, \ldots, a_N$. Furthermore, for all $f_\perp$ we have:

$$\Omega(\|f\|^2) = \Omega\left(\left\|\sum_{n=1}^N a_n \kappa(\cdot, \boldsymbol{x}_n)\right\|^2 + \|f_\perp\|^2_{\mathcal{H}}\right) \geq \Omega\left(\left\|\sum_{n=1}^N a_n \kappa(\cdot, \boldsymbol{x}_n)\right\|^2\right).$$

Thus, for any fixed $a_1, \ldots, a_n$ the value of the cost function is minimized for $f_\perp = 0$. Hence, the solution of the minimization task will have to obey this property too. □

Examples of loss functions $\mathcal{L}$ as the ones mentioned in Theorem 1.5.3 are for example the MSE:

$$\mathcal{L}((\boldsymbol{x}_1, y_1, f(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_N, y_N, f(\boldsymbol{x}_N)) = \sum_{n=1}^N (f(\boldsymbol{x}_n) - y_n)^2,$$

and the l_1 mean error

$$\mathcal{L}((\boldsymbol{x}_1, y_1, f(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_N, y_N, f(\boldsymbol{x}_N)) = \sum_{n=1}^N |f(\boldsymbol{x}_n) - y_n|.$$

The aforementioned theorem is of great importance to practical applications. Although one might be trying to solve an optimization task in an infinite dimensional RKHS $\mathcal{H}$ (such as the one that generated by the

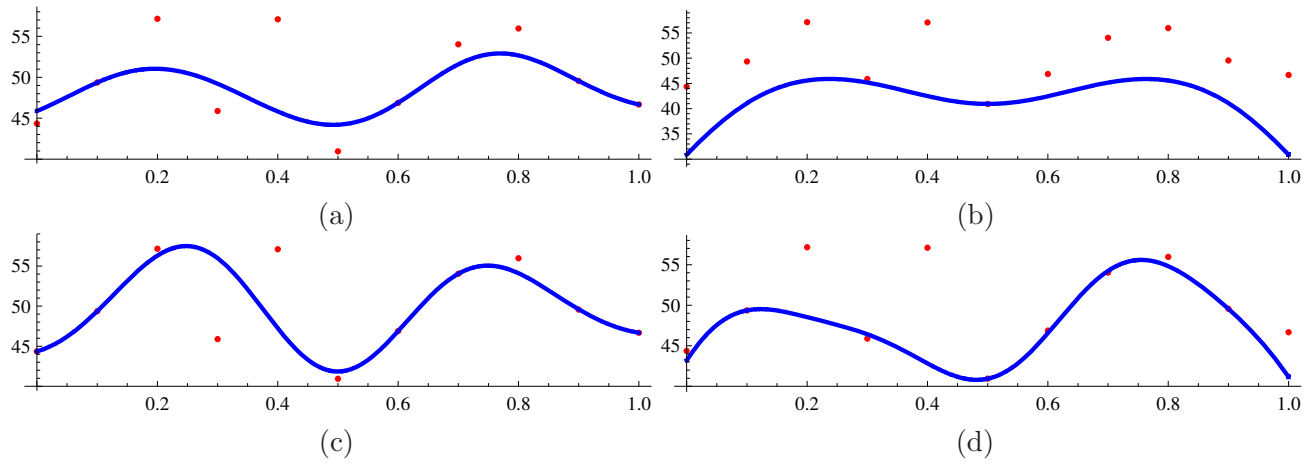


Figure 1.10: Solving the regression problem $\min_f \frac{1}{N}\sum_{n=1}^{N}(y_n - f(x_n))^2 + \lambda\|f\|_{\mathcal{H}}^2$, on a set of 11 points (a), (c) with a bias, i.e., f admits the form of (1.10) and (b), (d) without a bias, i.e., f admits the form of (1.11). In (a) and (b) we set $\sigma = 0.15$, $\lambda = 0.007$. In (c) and (d) we set $\sigma = 0.15$, $\lambda = 0.001$. Observe that for $\lambda = 0.001$, the unbiased solution takes values significantly lower compared to the values of the training points. For the smaller $\lambda = 0.001$, the difference between (c) and (d) is reduced (compared to the case $\lambda = 0.007$). However, one may observe that the unbiased solution (d) is not as smooth as the biased solution (c), especially near 0 and 1.

Gaussian kernel), the Representer Theorem states that the solution of the problem lies in the span of N particular kernels, those centered on the training points.

In practice, we often include a bias factor to the solution of kernel-based regularized minimization tasks, that is, we assume that f admits a representation of the form

$$f = \sum_{n=1}^{N} \theta_n \kappa(\cdot, \boldsymbol{x}_n) + b, \tag{1.10}$$

where $b \in \mathbb{R}$. This has been shown to improve the performance of the respective algorithms [43, 36], for two main reasons. Firstly, the introduction of the bias, b, enlarges the family of functions in which we search for a solution, thus leading to potentially better estimations. Moreover, as the regularization factor $\Omega(\|f\|_{\mathcal{H}}^2)$ penalizes the values of f at the training points, the resulting solution tends to take values as close to zero as possible, for large values of λ (see figure 1.10). The use of the bias factor is theoretically justified by the semi-parametric representer theorem.

Theorem 1.5.4 (Semi-parametric Representer Theorem)**.** *Suppose that in addition to the assumptions of theorem 1.5.3, we are given a set of M real valued functions $\{\psi_m\}_{m=1}^{M} : X \to \mathbb{R}$, with the property that the $N \times M$ matrix $(\psi_m(\boldsymbol{x}_n))_{n,m}$ has rank M. Then any $\tilde{f} := f + h$, with $f \in \mathcal{H}$ and $h \in \text{span}\{\psi_m;\ m = 1, \ldots, M\}$, solving*

$$\min_{\tilde{f}} \quad \mathcal{L}((\boldsymbol{x}_1, y_1, \tilde{f}(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_N, y_N, \tilde{f}(\boldsymbol{x}_N)) + \Omega(\|f\|_{\mathcal{H}}^2),$$

admits a representation of the form

$$\tilde{f} = \sum_{n=1}^{N} \theta_n \kappa(\cdot, \boldsymbol{x}_n) + \sum_{m=1}^{M} b_m \psi_m(\cdot), \tag{1.11}$$

with $\theta_n \in \mathbb{R}$, $b_m \in \mathbb{R}$, for all $n = 1, dots, N$, $m = 1, \ldots, M$.

The following results can be used for the construction of new kernels.

Proposition 1.5.4 (Conformal Transformations). *If $f : X \to R$ is any function, then $\kappa_1(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x})f(\boldsymbol{y})$ is a reproducing kernel. Moreover, if κ is any other reproducing kernel then $\kappa_2(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x})\kappa(\boldsymbol{x}, \boldsymbol{y})f(\boldsymbol{y})$ is also a reproducing kernel.*

Proof. The first part is a direct consequence of theorem 1.5.1. For the second part, consider $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in X$ and $a_1, \ldots, a_N \in \mathbb{R}$. Then

$$\begin{aligned}\sum_{n,m=1}^{N} a_n a_m f(\boldsymbol{x}_n)\kappa(\boldsymbol{x}_n, \boldsymbol{x}_m)f(\boldsymbol{x}_m) &= \sum_{n,m=1}^{N} a_n a_m f(\boldsymbol{x}_n)f(\boldsymbol{x}_m)\langle\Phi(\boldsymbol{x}_m), \Phi(\boldsymbol{x}_n)\rangle_{\mathcal{H}} \\ &= \left\langle \sum_{m}^{N} a_m f(\boldsymbol{x}_m)\Phi(\boldsymbol{x}_m), \sum_{n}^{N} a_n f(\boldsymbol{x}_n)\Phi(\boldsymbol{x}_n) \right\rangle_{\mathcal{H}} \\ &= \left\| \sum_{n}^{N} a_n f(\boldsymbol{x}_n)\Phi(\boldsymbol{x}_n) \right\| \geq 0.\end{aligned}$$

Moreover, as

$$\begin{aligned}\cos\left(\angle(\Phi_2(\boldsymbol{x}), \Phi_2(\boldsymbol{y}))\right) &= \frac{f(\boldsymbol{x})\kappa(\boldsymbol{x}, \boldsymbol{y})f(y)}{\sqrt{f(\boldsymbol{x})\kappa(\boldsymbol{x}, \boldsymbol{x})f(\boldsymbol{x})}\sqrt{(f(\boldsymbol{y})\kappa(\boldsymbol{y}, \boldsymbol{y})f(\boldsymbol{y})}} \\ &= \frac{\kappa(\boldsymbol{x}, \boldsymbol{y})}{\sqrt{\kappa(\boldsymbol{x}, \boldsymbol{x})}\sqrt{(\kappa(\boldsymbol{y}, \boldsymbol{y}))}} = \cos\left(\angle(\Phi(\boldsymbol{x}), \Phi(\boldsymbol{y}))\right),\end{aligned}$$

this transformation of the original kernel, preserves angles in the feature space. □

Theorem 1.5.5 (Restriction of a kernel). *Let $\mathcal{H}$ be a RKHS on X with respective kernel κ. Then κ restricted to the set $X_1 \subset X$ is the reproducing kernel of the class $\mathcal{H}_1$ of all restrictions of functions of $\mathcal{H}$ to the subset X_1. The respective norm of any such restricted function $f_1 \in \mathcal{H}_1$ (originating from $f \in \mathcal{H}$) has norm $\|f_1\|_{\mathcal{H}_1} = \min\{\|f\|_{\mathcal{H}},\ f \in \mathcal{H} : f|_{X_1} = f_1\}$.*

Proposition 1.5.5 (Normalization of a kernel). *Let $\mathcal{H}$ be a RKHS on X with respective kernel κ. Then*

$$\hat{\kappa}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\kappa(\boldsymbol{x}, \boldsymbol{y})}{\sqrt{\kappa(\boldsymbol{x}, \boldsymbol{x})\kappa(\boldsymbol{y}, \boldsymbol{y})}}, \tag{1.12}$$

is also a positive definite kernel on X. Note that $|\hat{\kappa}(\boldsymbol{x}, \boldsymbol{y})| \leq 1$, for all $\boldsymbol{x}, \boldsymbol{y} \in X$.

Proof. Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N \in X$ and $c_1, \ldots, c_N$ be real numbers. Then

$$\begin{aligned}\sum_{n,m=1}^{N} c_n c_m \hat{\kappa}(\boldsymbol{x}_n, \boldsymbol{x}_m) &= \sum_{n,m=1}^{N} c_n c_m \frac{\kappa(\boldsymbol{x}_n, \boldsymbol{x}_m)}{\sqrt{\kappa(\boldsymbol{x}_n, \boldsymbol{x}_n)\kappa(\boldsymbol{x}_m, \boldsymbol{x}_m)}} \\ &= \sum_{n,m=1}^{N} \frac{c_n}{\sqrt{\kappa(\boldsymbol{x}_n, \boldsymbol{x}_n)}} \frac{c_m}{\sqrt{\kappa(\boldsymbol{x}_m, \boldsymbol{x}_m)}} \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m) \geq 0,\end{aligned}$$

as κ is a positive definite kernel. □

Theorem 1.5.6 (Sum of kernels). *Let $\mathcal{H}_1, \mathcal{H}_2$ be two RKHS's on X with respective kernels κ_1, κ_2. Then $\kappa = \kappa_1 + \kappa_2$ is also a reproducing kernel. The corresponding RKHS, $\mathcal{H}$, contains the functions $f = f_1 + f_2$, where $f_i \in \mathcal{H}_i$, $i = 1, 2$. The respective norm is defined by*

$$\|f\|_{\mathcal{H}} = \min\{\|f_1\| + \|f_2\|, \quad \text{for all } f = f_1 + f_2, f_i \in \mathcal{H}_i, i = 1, 2\}.$$

Proof. It is trivial to show that $\kappa_1 + \kappa_2$ is a positive definite kernel. The difficult part is to associate this kernel with the specific RKHS $\mathcal{H}$. Consider the Hilbert space $F = H_1 \times H_2$. The respective inner product and the corresponding norm are defined as

$$\langle (f_1, f_2), (g_1, g_2) \rangle_F = \langle f_1, g_1 \rangle_{\mathcal{H}_1} + \langle f_2, g_2 \rangle_{\mathcal{H}_2},$$
$$\|(f_1, f_2)\|_F^2 = \|f_1\|_{\mathcal{H}_1}^2 + \|f_2\|_{\mathcal{H}_2}^2,$$

for $f_1, g_1, \in \mathcal{H}_1$ and $f_2, g_2 \in \mathcal{H}_2$. If $\mathcal{H}_1$ and $\mathcal{H}_2$ have only 0 in common, it easy to show that there is a one-to-one correspondence between F and $\mathcal{H} = \mathcal{H}_1 + \mathcal{H}_2$, as each $f \in \mathcal{H}$ can be decomposed into two parts (one belonging to $\mathcal{H}_1$ and the other in $\mathcal{H}_2$) uniquely. The difficult part is to discover such a relation, if $\mathcal{H}_0 = \mathcal{H}_1 \cap \mathcal{H}_2$ is larger than $\{0\}$. To make this fact clear, consider this simple example: Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be the linear classes of polynomials of orders up to 1 and up to 2 respectively. Obviously, $\mathcal{H} = \mathcal{H}_1 + \mathcal{H}_2 = \mathcal{H}_2$, as $\mathcal{H}_1 \subset \mathcal{H}_2$. Let $f(x) = x^2 + 5x$, $f \in \mathcal{H}$. Then f can be decomposed into two parts (one belonging to $\mathcal{H}_1$ and the other in $\mathcal{H}_2$) in more than one ways. For example $f(x) = (x^2) + (5x)$, or $f(x) = (x^2 + 4x) + (x)$, or $f(x) = (x^2 + 2x) + (3x)$, e.t.c. Thus, the mapping between $f = f_1 + f_2 \in \mathcal{H}$ and $(f_1, f_2) \in F$ is not one-to-one. However, in such cases, we can still find a smaller subspace of F, which can be identified to $\mathcal{H}$.

To this end, define $F_0 = \{(f, -f),\ f \in \mathcal{H}_0\}$. It is clear that F_0 is a linear subspace of F. We will show that it is a closed one. Consider the converging sequence in F_0: $(f_n, -f_n) \to (\tilde{f}_1, \tilde{f}_2)$. Then $f_n \to \tilde{f}_1$ and $-f_n \to \tilde{f}_2$. Thus $\tilde{f}_1 = -\tilde{f}_2$ and $(\tilde{f}_1, \tilde{f}_2)$ is in F_0. As F_0 is a closed linear subspace of F, we may consider its complementary subspace $F_0^\perp$: $F = F_0 \oplus F_0^\perp$.

As a next step, consider the linear transformation $T : F \to \mathcal{H} : T(f_1, f_2) = f_1 + f_2$. The kernel of this transformation is the subspace F_0. Hence, there is a one-to-one correspondence between $F_0^\perp$ and $\mathcal{H}$. Consider the inverse transformation T^{-1} and let $T^{-1}(f) = (f', f'')$, for $f \in \mathcal{H}$, where $f' \in \mathcal{H}_1$ and $f'' \in \mathcal{H}_2$, i.e., through T^{-1} we decompose f uniquely into two components, one in $\mathcal{H}_1$ and the other in $\mathcal{H}_2$. This decomposition enables us to define an inner product in $\mathcal{H}$, i.e.,

$$\langle f, g \rangle_{\mathcal{H}} = \langle f' + f'', g' + g'' \rangle_{\mathcal{H}} = \langle f', g' \rangle_{\mathcal{H}_1} + \langle f'', g'' \rangle_{\mathcal{H}_2} = \langle (f', f''), (g', g'') \rangle_F,$$

for $f, g \in \mathcal{H}$. To prove that to this $\mathcal{H}$ there corresponds the kernel $\kappa = \kappa_1 + \kappa_2$, we make the following remarks:

1. For every $\boldsymbol{y} \in X$, $\kappa(\cdot, \boldsymbol{y}) = \kappa_1(\cdot, \boldsymbol{y}) + \kappa_2(\cdot, \boldsymbol{y}) \in \mathcal{H}$.

2. For every $y \in X$, let $T^{-1}(\kappa(\cdot, \boldsymbol{y})) = (\kappa'(\cdot, \boldsymbol{y}), \kappa''(\cdot, \boldsymbol{y}))$. Thus

$$\kappa(\boldsymbol{x}, \boldsymbol{y}) = \kappa'(\boldsymbol{x}, \boldsymbol{y}) + \kappa''(\boldsymbol{x}, \boldsymbol{y}) = \kappa_1(\boldsymbol{x}, \boldsymbol{y}) + \kappa_2(\boldsymbol{x}, \boldsymbol{y}),$$

and consequently $\kappa_1(\boldsymbol{x}, \boldsymbol{y}) - k'(\boldsymbol{x}, \boldsymbol{y}) = -(\kappa_2(\boldsymbol{x}, \boldsymbol{y}) - \kappa''(\boldsymbol{x}, \boldsymbol{y}))$. This means that $(\kappa_1(\boldsymbol{x}, \boldsymbol{y}) - k'(\boldsymbol{x}, \boldsymbol{y}), \kappa_2(\boldsymbol{x}, \boldsymbol{y}) - \kappa''(\boldsymbol{x}, \boldsymbol{y})) \in F_0$. Hence, for every $\boldsymbol{y} \in X$ we have

$$\begin{aligned} f(\boldsymbol{y}) =& f'(\boldsymbol{y}) + f''(\boldsymbol{y}) = \langle f', \kappa_1(\cdot, \boldsymbol{y}) \rangle_{\mathcal{H}_1} + \langle f'', \kappa_2(\cdot, \boldsymbol{y}) \rangle_{\mathcal{H}_2} \\ =& \langle (f', f''), (\kappa_1(\cdot, \boldsymbol{y}), \kappa_2(\cdot, \boldsymbol{y})) \rangle_F \\ =& \langle (f', f''), (\kappa'(\cdot, \boldsymbol{y}), \kappa''(\cdot, \boldsymbol{y})) \rangle_F + \langle (f', f''), (\kappa_1(\cdot, \boldsymbol{y}) - \kappa'(\cdot, \boldsymbol{y}), \kappa_2(\cdot, \boldsymbol{y}) - \kappa''(\cdot, \boldsymbol{y}) \rangle_F. \end{aligned}$$

As $(\kappa_1(\boldsymbol{x}, \boldsymbol{y}) - k'(\boldsymbol{x}, \boldsymbol{y}), \kappa_2(\boldsymbol{x}, \boldsymbol{y}) - \kappa''(\boldsymbol{x}, \boldsymbol{y})) \in F_0$ and $(f', f'') \in F_0^\perp$, we conclude that

$$f(y) = \langle (f', f''), (\kappa'(\cdot, y), \kappa''(\cdot, \boldsymbol{y}) \rangle_F = \langle f' + f'', \kappa'(\cdot, \boldsymbol{y}) + \kappa''(\cdot, \boldsymbol{y}) \rangle_{\mathcal{H}} = \langle f, \kappa(\cdot, \boldsymbol{y}) \rangle_{\mathcal{H}}.$$

This is the reproducing property.

Finally, to prove the last part of the theorem, consider again $f \in \mathcal{H}$ and let $f_i \in \mathcal{H}_i$, $i = 1, 2$, such that $f = f_1 + f_2$ and let $f' \in \mathcal{H}_1$ and $f'' \in \mathcal{H}_2$ be the unique decomposition of f through T^{-1}. As $f_1 + f_2 = f' + f''$ we obtain that $f' - f_1 = -(f'' - f_2)$, which implies that $(f' - f_1, f'' - f_2) \in F_0$. Thus, we take:

$$\begin{aligned} \|f_1\|_{\mathcal{H}_1}^2 + \|f_2\|_{\mathcal{H}_2}^2 = \|(f_1, f_2)\|_F^2 =& \|(f', f'')\|_F^2 + \|(f_1 - f', f_2 - f'')\|_F^2 \\ =& \|f'\|_{\mathcal{H}_1}^2 + \|f''\|_{\mathcal{H}_2}^2 + \|(f_1 - f', f_2 - f'')\|_F^2 \\ =& \|f\|_{\mathcal{H}}^2 + \|(f_1 - f', f_2 - f'')\|_F^2. \end{aligned}$$

From the last relation we conclude that $\|f\|^2_{\mathcal{H}} = \|f_1\|^2_{\mathcal{H}_1} + \|f_2\|^2_{\mathcal{H}_2}$, if and only if $f_1 = f'$ and $f_2 = f''$. In this case we take the minimum value of $\|f_1\|^2_{\mathcal{H}_1} + \|f_2\|^2_{\mathcal{H}_2}$, for all possible decompositions $f = f_1 + f_2$. This completes the proof. □

Despite the sum of kernels, other operations preserve reproducing kernels as well. Below, we give an extensive list of such operations. For a description of the induced RKHS and a formal proof (in the cases that are not considered here) the interested reader may refer to [2, 37].

1. If $\kappa(\boldsymbol{x}, \boldsymbol{y})$ is a positive definite kernel on X, then $\lambda\kappa(\boldsymbol{x}, \boldsymbol{y})$ is also a positive definite kernel for any $\lambda \geq 0$. It is obvious that in this case $\mathcal{H}(\lambda\kappa) = \mathcal{H}(\kappa)$, if $\lambda > 0$. If $\lambda = 0$, then $\mathcal{H}(0) = \{0\}$.

2. If $\kappa_1(\boldsymbol{x}, \boldsymbol{y})$ and $\kappa_2(\boldsymbol{x}, \boldsymbol{y})$ are positive definite kernels on X, then $\kappa_1(\boldsymbol{x}, \boldsymbol{y}) + \kappa_2(\boldsymbol{x}, \boldsymbol{y})$ is also a positive definite kernel, as Theorem 1.5.6 established.

3. If $\kappa_1(\boldsymbol{x}, \boldsymbol{y})$ and $\kappa_2(\boldsymbol{x}, \boldsymbol{y})$ are positive definite kernels on X, then $\kappa_1(\boldsymbol{x}, \boldsymbol{y}) \cdot \kappa_2(\boldsymbol{x}, \boldsymbol{y})$ is also a positive definite kernel.

4. If $\kappa_n(\boldsymbol{x}, \boldsymbol{y})$ are positive definite kernels on X, such that $\lim_n \kappa_n(\boldsymbol{x}, \boldsymbol{y}) = \kappa(\boldsymbol{x}, \boldsymbol{y})$, for all $\boldsymbol{x}, \boldsymbol{y} \in X$, then $\kappa(\boldsymbol{x}, \boldsymbol{y})$ is also a positive definite kernel.

5. If $\kappa(\boldsymbol{x}, \boldsymbol{y})$ is a positive definite kernel on X and $p(z)$ is a polynomial with non-negative coefficients, then $p(\kappa(\boldsymbol{x}, \boldsymbol{y}))$ is also a positive definite kernel.

6. If $\kappa(\boldsymbol{x}, \boldsymbol{y})$ is a positive definite kernel on X, then $e^{\kappa(\boldsymbol{x}, \boldsymbol{y})}$ is also a positive definite kernel. To prove this, consider the Taylor expansion formula of e^z, which may be consider as a limit of polynomials with non-negative coefficients.

7. If $\kappa(\boldsymbol{x}, \boldsymbol{y})$ is a positive definite kernel on X and $\Psi : X' \to X$ is a function, then $\kappa(\Psi(\boldsymbol{x}), \Psi(\boldsymbol{y}))$ is a positive definite kernel on X'.

8. If $\kappa_1(\boldsymbol{x}, \boldsymbol{y})$ and $\kappa_2(\boldsymbol{x}', \boldsymbol{y}')$ are positive definite kernels on X and X' respectively, then their tensor product $(\kappa_1 \otimes \kappa_2)(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{x}', \boldsymbol{y}') = \kappa_1(\boldsymbol{x}, \boldsymbol{y})\kappa_2(\boldsymbol{x}', \boldsymbol{y}')$, is a kernel on $X \times X'$.

9. If $\kappa_1(\boldsymbol{x}, \boldsymbol{y})$ and $\kappa_2(\boldsymbol{x}', \boldsymbol{y}')$ are positive definite kernels on X and X' respectively, then their direct sum $(\kappa_1 \oplus \kappa_2)(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{x}', \boldsymbol{y}') = \kappa_1(\boldsymbol{x}, \boldsymbol{y}) + \kappa_2(\boldsymbol{x}', \boldsymbol{y}')$, is a kernel on $X \times X'$.

1.6 Dot product and translation invariant kernels

There are two important classes of kernels that follow certain rules and are widely used in practice. The first one includes the *dot product kernels*, which are functions defined as $\kappa(\boldsymbol{x}, \boldsymbol{y}) = f(\langle \boldsymbol{x}, \boldsymbol{y} \rangle)$, for some real function f. The second class are the *translation invariant kernels*, which are defined as $\kappa(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x} - \boldsymbol{y})$, for some real function f defined on X. The following theorems establish necessary and sufficient conditions for such functions to be reproducing kernels.

Theorem 1.6.1 (Power Series of dot product kernels)**.** *Let $f : \mathbb{R} \to \mathbb{R}$. A function $\kappa(\boldsymbol{x}, \boldsymbol{y}) = f(\langle \boldsymbol{x}, \boldsymbol{y} \rangle)$ defined on X, such that f has the power series expansion $f(t) = \sum_n a_n t^n$, is a positive definite kernel, if and only if we have $a_n \geq 0$ for all n.*

Theorem 1.6.2 (Bochner's - Fourier Criterion for translation invariant kernels)**.** *Let $f : X \to \mathbb{R}$. A function $\kappa(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x} - \boldsymbol{y})$ defined on $X \subseteq \mathbb{R}^\nu$, is a positive definite kernel, if the Fourier transform*

$$F[k](\omega) = (2\pi)^{-\frac{N}{2}} \int_X e^{-i\langle \boldsymbol{\omega}, \boldsymbol{x} \rangle} f(\boldsymbol{x}) d\boldsymbol{x}$$

is non-negative.

Remark 1.6.1. *Bochner's theorem is more general, involving Borel measures and topological spaces. For the sake of simplicity we give only this simple form.*

Employing the tools provided in this section, one can readily prove the positivity of some of the kernels given in section 1.4. For example:

- Homogeneous Polynomial Kernel: As $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ is a positive definite kernel and $p(z) = z^d$ is a polynomial with non-negative coefficients, $p(\langle \boldsymbol{x}, \boldsymbol{y} \rangle) = (\langle \boldsymbol{x}, \boldsymbol{y} \rangle)^d$ is a positive definite kernel.
- Inhomogeneous Polynomial Kernel: As $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ is a positive definite kernel, and $p(z) = (z + c)^d$ is a polynomial with non-negative coefficients (for positive c), $p(\langle \boldsymbol{x}, \boldsymbol{y} \rangle) = (c + \langle \boldsymbol{x}, \boldsymbol{y} \rangle)^d$ is a positive definite kernel.
- The cosine kernel: Note that $\cos(\angle(\boldsymbol{x}, \boldsymbol{y})) = \frac{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}{\|\boldsymbol{x}\| \|\boldsymbol{y}\|}$. Thus the cosine kernel is the normalization of the simple kernel $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$.

To prove that the Gaussian and the Laplacian are positive kernels we need another set of tools. This is the topic of the next section.

1.7 The Gaussian kernel and other translation invariant kernels

As the Gaussian kernel is the most widely used in applications, we dedicate this section to present some of its most important properties. We begin our study showing that the gaussian radial basis function is indeed a reproducing kernel. To this end, we introduce some new notions.

Definition 1.7.1 (Negative Definite Kernel)**.** *Let X be a set. A function $\kappa : X \times X \to \mathbb{R}$ is called a negative definite kernel if it is symmetric, i.e., $\kappa(y, x) = \kappa(x, y)$, and*

$$\sum_{n,m=1}^{N} c_n c_m \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m) \leq 0,$$

for any $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in X$ and $c_1, \ldots, c_N \in \mathbb{R}$, with $\sum_{n=1}^{N} c_n = 0$.

Examples of negative kernels are the constant functions and all functions of the form $-\kappa$, where κ is a positive definite kernel. Furthermore, the following proposition holds:

Proposition 1.7.1. *Let X be a non empty set, the functions $\psi_k : X \times X \to \mathbb{R}$ be negative kernels and $\alpha_k > 0$, for $k \in \mathbb{N}$. Then*

- *Any positive combination of a finite number of negative kernels is also a negative kernel, i.e., $\psi = \sum_k \alpha_k \psi_k$, with $\alpha_1, \ldots, \alpha_n > 0$ is a negative kernel.*
- *The limit of any converging sequence of negative kernels is also a negative kernel, i.e. if $\psi(x, y) = \lim_k \psi_k(x, y)$, for all $x, y \in X$, then ψ is a negative kernel.*

Proof. For the first part, consider the numbers $c_1, \ldots, c_N$ such that $\sum_{n=1}^{N} c_n = 0$, $x_1, \ldots, x_N \in X$ and $K \in \mathbb{N}$. Then

$$\sum_{n,m=1}^{N} c_n c_m \sum_{k=1}^{K} \alpha_k \psi_k(\boldsymbol{x}_n, \boldsymbol{x}_m) = \sum_{k=1}^{K} \alpha_k \sum_{n,m=1}^{N} c_n c_m \psi_k(\boldsymbol{x}_n, \boldsymbol{x}_m) \leq 0.$$

Finally, to prove the second part we take:

$$\sum_{n,m=1}^{N} c_n c_m \psi(\boldsymbol{x}_n, \boldsymbol{x}_m) = \sum_{n,m=1}^{N} c_n c_m \lim_k \psi_k(\boldsymbol{x}_n, \boldsymbol{x}_m) = \lim_k \sum_{n,m=1}^{N} c_n c_m \psi_k(\boldsymbol{x}_n, \boldsymbol{x}_m) \leq 0.$$

□

Lemma 1.7.1. *Let X be a nonempty set, V be a vector space equipped with an inner product and $T : X \to V$. Then the function*

$$\psi(\boldsymbol{x}, \boldsymbol{y}) = \|T(\boldsymbol{x}) - T(\boldsymbol{y})\|_V^2$$

is a negative definite kernel on X.

Proof. Consider the numbers $c_1, \ldots, c_N$ such that $\sum_{n=1}^N c_n = 0$ and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in X$. Then

$$\begin{aligned}
\sum_{n,m=1}^N c_n c_m \|T(\boldsymbol{x}_n) - T(\boldsymbol{x}_m)\|_V^2 &= \sum_{n,m=1}^N c_n c_m \langle T(\boldsymbol{x}_n) - T(\boldsymbol{x}_m), T(\boldsymbol{x}_n) - T(\boldsymbol{x}_m)\rangle_V \\
&= \sum_{n,m=1}^N c_n c_m \left(\|T(\boldsymbol{x}_n)\|_V^2 + \|T(\boldsymbol{x}_m)\|_V^2 - \langle T(\boldsymbol{x}_n), T(\boldsymbol{x}_m)\rangle_V - \langle T(\boldsymbol{x}_m), T(\boldsymbol{x}_n)\rangle_V \right) \\
&= \sum_{m=1}^N c_m \sum_{n=1}^N c_n \|T(\boldsymbol{x}_n)\|_V^2 + \sum_{n=1}^N c_n \sum_{m=1}^N c_m \|T(\boldsymbol{x}_m)\|_V^2 \\
&\quad - \left\langle \sum_{n=1}^N c_n T(\boldsymbol{x}_n), \sum_{m=1}^N c_m T(\boldsymbol{x}_m) \right\rangle_V - \left\langle \sum_{m=1}^N c_m T(\boldsymbol{x}_m), \sum_{n=1}^N c_n T(\boldsymbol{x}_n) \right\rangle_V .
\end{aligned}$$

As $\sum_{n=1}^N c_n = 0$, the first two terms of the summation vanish and we take:

$$\sum_{n,m=1}^N c_n c_m \|T(\boldsymbol{x}_n) - T(\boldsymbol{x}_m)\|_V^2 = -2 \left\| \sum_{n=1}^N c_n T(\boldsymbol{x}_n) \right\|_V^2 \leq 0.$$

Thus ψ is a negative definite kernel. □

Lemma 1.7.2. *Let $\psi : X \times X \to \mathbb{R}$ be a function. Fix $\boldsymbol{x}_0 \in X$ and define*

$$\kappa(\boldsymbol{x}, \boldsymbol{y}) = -\psi(\boldsymbol{x}, \boldsymbol{y}) + \psi(\boldsymbol{x}, \boldsymbol{x}_0) + \psi(\boldsymbol{x}_0, \boldsymbol{y}) - \psi(\boldsymbol{x}_0, \boldsymbol{x}_0).$$

Then ψ is a negative definite kernel if and only if κ is a positive definite kernel.

Proof. Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in X$. For the if part, consider the numbers $c_1, \ldots, c_N$ such that $\sum_{n=1}^N c_n = 0$. Then

$$\begin{aligned}
\sum_{n,m=1}^N c_n c_m \kappa(\boldsymbol{x}_n, y_m) &= -\sum_{n,m=1}^N c_n c_m \psi(\boldsymbol{x}_n, \boldsymbol{x}_m) + \sum_{n,m=1}^N c_n c_m \psi(\boldsymbol{x}_n, \boldsymbol{x}_0) \\
&\quad + \sum_{n,m=1}^N c_n c_m \psi(\boldsymbol{x}_0, \boldsymbol{x}_m) - \sum_{n,m=1}^N c_n c_m \psi(\boldsymbol{x}_0, \boldsymbol{x}_0) \\
&= -\sum_{n,m=1}^N c_n c_m \psi(\boldsymbol{x}_n, \boldsymbol{x}_m) + \sum_{n}^N c_m \sum_{n}^N c_n \psi(\boldsymbol{x}_n, \boldsymbol{x}_0) \\
&\quad + \sum_{n=1}^N c_n \sum_{m=1}^N c_m \psi(\boldsymbol{x}_0, \boldsymbol{x}_m) - \sum_{m=1}^N c_m \sum_{m=1}^N c_n \psi(\boldsymbol{x}_0, \boldsymbol{x}_0).
\end{aligned}$$

As $\sum_{n=1}^N c_n = 0$ and $\sum_{n,m=1}^N c_n c_m \kappa(\boldsymbol{x}_n, y_m) \geq 0$, we take that $\sum_{n,m=1}^N c_n c_m \psi(\boldsymbol{x}_n, \boldsymbol{y}_m) \leq 0$. Thus ψ is a negative definite kernel.

For the converse, take $c_1, \ldots, c_N \in \mathbb{R}$ and define $c_0 = -\sum_{n=1}^N c_n$. By this simple trick, we generate the numbers $c_0, c_1, \ldots, c_N \in \mathbb{R}$, which have the property $\sum_{n=0}^N c_n = 0$. As ψ is a negative definite kernel, we

take that $\sum_{n,m=0}^{N} c_n c_m \psi(\boldsymbol{x}_n, \boldsymbol{x}_m) \leq 0$, for any $\boldsymbol{x}_0 \in X$. Thus,

$$\begin{aligned}
\sum_{n,m=0}^{N} c_n c_m \psi(\boldsymbol{x}_n, \boldsymbol{x}_m) &= \sum_{n,m=1}^{N} c_n c_m \psi(\boldsymbol{x}_n, \boldsymbol{x}_m) + \sum_{m=1}^{N} c_0 c_m \psi(\boldsymbol{x}_0, \boldsymbol{x}_m) + \sum_{n=1}^{N} c_0 c_n \psi(\boldsymbol{x}_n, \boldsymbol{x}_0) + c_0^2 \psi(\boldsymbol{x}_0, \boldsymbol{x}_0) \\
&= \sum_{n,m=1}^{N} c_n c_m \psi(\boldsymbol{x}_n, \boldsymbol{x}_m) - \sum_{n,m=1}^{N} c_n c_m \psi(\boldsymbol{x}_0, \boldsymbol{x}_m) \\
&\quad - \sum_{n,m=1}^{N} c_n c_m \psi(\boldsymbol{x}_n, \boldsymbol{x}_0) + \sum_{n,m=1}^{N} c_n c_m \psi(\boldsymbol{x}_0, \boldsymbol{x}_0) \\
&= \sum_{n,m=1}^{N} c_n c_m \left(\psi(\boldsymbol{x}_n, \boldsymbol{x}_m) - \psi(\boldsymbol{x}_0, \boldsymbol{x}_m) - \psi(\boldsymbol{x}_n, \boldsymbol{x}_0) + \psi(\boldsymbol{x}_0, \boldsymbol{x}_0)\right) \\
&= - \sum_{n,m=1}^{N} c_n c_m \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m).
\end{aligned}$$

Thus $\sum_{n,m=1}^{N} c_n c_m \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m) \geq 0$ and κ is a positive definite kernel. □

Theorem 1.7.1 (Schoenberg)**.** *Let X be a nonempty set and $\psi : X \times X \to \mathbb{R}$. The function ψ is a negative kernel if and only if $\exp(-t\psi)$ is a positive definite kernel for all $t \geq 0$.*

Proof. For the if part, recall that

$$\psi(\boldsymbol{x}, \boldsymbol{y}) = \lim_{t \to 0} \frac{1 - \exp(-t\psi(\boldsymbol{x}, \boldsymbol{y}))}{t}.$$

As $\exp(-t\psi)$ is positive definite, $-\exp(-t\psi)$ is negative definite and the result follows from Proposition 1.7.1. It suffices to prove the converse for $t = 1$, as if ψ is a negative definite kernel so is $t\psi$, for any $t \geq 0$. Take $\boldsymbol{x}_0 \in X$ and define the positive definite kernel $\kappa(\boldsymbol{x}, \boldsymbol{y}) = -\psi(\boldsymbol{x}, \boldsymbol{y}) + \psi(\boldsymbol{x}, \boldsymbol{x}_0) + \psi(\boldsymbol{x}_0, \boldsymbol{y}) - \psi(\boldsymbol{x}_0, \boldsymbol{x}_0)$ (Lemma 1.7.2). Then

$$e^{-\psi(\boldsymbol{x}, \boldsymbol{y})} = e^{-\psi(\boldsymbol{x}, \boldsymbol{x}_0)} e^{\kappa(\boldsymbol{x}, \boldsymbol{y})} e^{-\psi(\boldsymbol{x}_0, \boldsymbol{y})} e^{\kappa(\boldsymbol{x}_0, \boldsymbol{x}_0)}.$$

Let $f(x) = e^{-\psi(\boldsymbol{x}, \boldsymbol{x}_0)}$. Then, as ψ is a negative kernel and therefore symmetric, one can readily prove that the last relation can be rewritten as

$$e^{-\psi(\boldsymbol{x}, \boldsymbol{y})} = e^{\kappa(\boldsymbol{x}_0, \boldsymbol{x}_0)} \cdot f(\boldsymbol{x}) e^{\kappa(\boldsymbol{x}, \boldsymbol{y})} f(y).$$

Since $e^{\kappa(\boldsymbol{x}_0, \boldsymbol{x}_0)}$ is a positive number, employing the properties of positive kernels given in section 1.5, we conclude that $e^{-\psi(\boldsymbol{x}, \boldsymbol{y})}$ is a positive definite kernel. □

Corollary 1.7.1. *The Gaussian radial basis function is a reproducing kernel.*

Although all properties of positive kernels do not apply to negative kernels as well (for example the product of negative kernels is not a negative kernel), there are some other operations that preserve negativity.

Proposition 1.7.2. *Let $\psi : X \times X \to \mathbb{R}$ be negative definite. In this case:*

1. *If $\psi(\boldsymbol{x}, \boldsymbol{x}) \geq 0$, for all $\boldsymbol{x} \in X$, then $\psi^p(\boldsymbol{x}, \boldsymbol{y})$ is negative definite for any $0 < p \leq 1$.*
2. *If $\psi(\boldsymbol{x}, \boldsymbol{x}) \geq 0$, for all $x \in X$, then $\log(1 + \psi(\boldsymbol{x}, \boldsymbol{y}))$ is negative definite.*
3. *If $\psi : X \times X \to (0, +\infty)$, then $\log \psi(\boldsymbol{y}, \boldsymbol{x})$ is negative definite.*

Proof. We give a brief description of the proofs.

1. We use the formula:

$$\psi(\boldsymbol{x},\boldsymbol{y})^p = \frac{p}{\Gamma(1-p)}\int_0^\infty t^{-p-1}\left(1-e^{-t\psi(\boldsymbol{x},\boldsymbol{y})}\right)dt,$$

where the Gamma function is given by $\Gamma(z)=\int_0^\infty e^{-t}t^z\,dt$. As $e^{-t\psi(\boldsymbol{x},\boldsymbol{y})}$ is positive definite (Theorem 1.7.1) and -1, t^{-p-1} are positive numbers, it is not difficult to prove that the expression inside the integral is negative definite for all $t>0$.

2. Similarly, we use the formula:

$$\log(1+\psi(\boldsymbol{x},\boldsymbol{y})) = \int_0^\infty \frac{e^{-t}}{t}\left(1-e^{-t\psi(\boldsymbol{x},\boldsymbol{y})}\right)dt.$$

3. For any $c>0$, $\log(\psi(\boldsymbol{x},\boldsymbol{y})+1/c) = \log(1+c\psi(\boldsymbol{x},\boldsymbol{y})) - log(c)$. We can prove that the second part is negative definite. Then, by taking the limit $c\to 0$, one completes the proof.

□

As a direct consequence, one can prove that since $\|\boldsymbol{x}-\boldsymbol{y}\|^2$ is a negative kernel, so is $\|\boldsymbol{x}-\boldsymbol{y}\|^{2p}$, for any $0<p\leq 1$. Thus, for any $0<p\leq 2$, $\|\boldsymbol{x}-\boldsymbol{y}\|^p$ is a negative kernel and $\exp(-t\|\boldsymbol{x}-\boldsymbol{y}\|^p)$ is a positive kernel for any $t>0$. Therefore, for $p=2$ we take another proof of the positivity of the Gaussian radial basis function. In addition, for $p=1$ one concludes that the Laplacian radial basis function is also a positive kernel. Moreover, for the Gaussian kernel the following important property has been proved.

Theorem 1.7.2 (Full rank of the Gaussian RBF Gram matrices)**.** *Suppose that $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N\subset X$ are distinct points and $\sigma\neq 0$. The Gram matrix given by*

$$K_{n,m} = \exp\left(-\frac{\|\boldsymbol{x}_n-\boldsymbol{x}_m\|^2}{2\sigma^2}\right),$$

has full rank.

As a consequence, for any choice of discrete points $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N$, we have that $\sum_{m=1}^N a_m\kappa(\boldsymbol{x}_n,\boldsymbol{x}_m)=0$, for all $n=1,2\ldots,N$, if and only if $a_1=\cdots=a_N=0$. However, observe that for any $a_1,\ldots,a_N$

$$\sum_{m=1}^N a_m\kappa(\boldsymbol{x}_n,\boldsymbol{x}_m) = \sum_{m=1}^N a_m\langle\kappa(\cdot,\boldsymbol{x}_m),\kappa(\cdot,\boldsymbol{x}_n)\rangle_{\mathcal{H}} = \left\langle\sum_{m=1}^N a_m\kappa(\cdot,\boldsymbol{x}_m),\kappa(\cdot,\boldsymbol{x}_n)\right\rangle_{\mathcal{H}} = \langle f,\kappa(\cdot,\boldsymbol{x}_n)\rangle_{\mathcal{H}},$$

where $f=\sum_{m=1}^N a_m\kappa(\cdot,\boldsymbol{x}_m)\in\text{span}\{\kappa(\cdot,\boldsymbol{x}_m),\ m=1,\ldots,N\}$. In addition, if for an $f\in\text{span}\{\kappa(\cdot,\boldsymbol{x}_m),\ m=1,\ldots,N\}$ we have that $\langle f,\kappa(\cdot,\boldsymbol{x}_n)\rangle_{\mathcal{H}}=0$ for all $n=1,\ldots,N$, if and only if $f=0$. Hence, if f is orthogonal to all $\Phi(\boldsymbol{x}_n)$, then $f=0$. We conclude that $f=\sum_{m=1}^N a_m\kappa(\cdot,\boldsymbol{x}_m)=0$ if and only if $a_1=\cdots=a_N=0$. Therefore, the points $\Phi(\boldsymbol{x}_m)=\kappa(\cdot,\boldsymbol{x}_m)$, $m=1,\ldots,N$, are linearly independent, provided that no two $\boldsymbol{x}_m$ are the same. *Hence, a Gaussian kernel defined on a domain of infinite cardinality, produces a feature space of infinite dimension.* Moreover, the Gram matrices defined by Gaussian kernels are always strictly positive definite and invertible.

In addition, for every $\boldsymbol{x},\boldsymbol{y}\in X$ we have that $\kappa(\boldsymbol{x},\boldsymbol{x})=1$ and $\kappa(\boldsymbol{x},\boldsymbol{y})\geq 0$. This means that all $\boldsymbol{x}\in X$ are mapped through the feature map Φ to points lying in the surface of the unit sphere of the RKHS $\mathcal{H}$ and that the angle between any two mapped points $\Phi(\boldsymbol{x})$ and $\Phi(\boldsymbol{y})$ is between 0^o and 90^o degrees. We conclude this section with the following two important formulas, which hold for the case of the RKHS induced by the Gaussian kernel. For the norm of $f\in\mathcal{H}$, one can prove that:

$$\|f\|_{\mathcal{H}}^2 = \int_X\sum_n \frac{\sigma^{2n}}{n!2^n}(O^n f(\boldsymbol{x}))^2 d\boldsymbol{x}, \tag{1.13}$$

with $O^{2n}=\Delta^n$ and $O^{2n+1}=\nabla\Delta^n$, Δ being the Laplacian and ∇ the gradient operator. The implication of this is that a regularization term of the form $\|f\|_{\mathcal{H}}^2$ (which is usually adopted in practice) "penalizes" the

derivatives of the minimizer. This results to a very smooth solution of the regularized risk minimization problem. Finally, the Fourier transform of the Gaussian kernel κ_σ is given by

$$F[k](\omega) = |\sigma| \exp\left(-\frac{\sigma^2\omega^2}{2}\right). \tag{1.14}$$

1.8 The Complex case

It has already been mentioned in section 1.2, that the general theory of RKHS was developed by the mathematicians for general complex Hilbert spaces. In this context, a positive definite matrix is defined as a Hermitian matrix $\boldsymbol{K} = (K_{i,j})^N$ satisfying

$$\boldsymbol{c}^H \cdot \boldsymbol{K} \cdot \boldsymbol{c} = \sum_{i=1,j=1}^{N,N} c_i^* c_j K_{i,j} \geq 0,$$

for all $c_i \in \mathbb{C}$, $i = 1, \ldots, N$, where the notation * denotes the conjugate element and $\cdot^H$ the conjugate transpose matrix. The general theory considers linear classes of complex functions under the field of complex numbers, i.e., the scalar product $c \cdot f$ is defined with complex numbers ($c \in \mathbb{C}$) where the multiplication is the standard complex one. The definition of a complex RKHS is identical to one given in the real case. A complex Hilbert space $\mathcal{H}$ will be called a RKHS, if the following two important properties hold:

1. For every $\boldsymbol{x} \in X$, $\kappa(\cdot, \boldsymbol{x})$ belongs to $\mathcal{H}$.
2. κ has the so called *reproducing property*, i.e.,

   $$f(\boldsymbol{x}) = \langle f, \kappa(\cdot, \boldsymbol{x})\rangle_{\mathcal{H}}, \text{ for all } f \in \mathcal{H},$$

   in particular $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \langle \kappa(\cdot, \boldsymbol{y}), \kappa(\cdot, \boldsymbol{x})\rangle_{\mathcal{H}}$.

The main difference with the real case lies in the definition of the complex inner product, where the linearity and the symmetry properties do not hold. Recall that in the case of complex Hilbert spaces the inner product is sesqui-linear (i.e., linear in one argument and anti-linear in the other) and Hermitian:

$$\begin{aligned} \langle af + bg, h\rangle_{\mathcal{H}} &= a\langle f, h\rangle_{\mathcal{H}} + b\langle g, h\rangle_{\mathcal{H}}, \\ \langle f, ag + bh\rangle_{\mathcal{H}} &= a^*\langle f, g\rangle_{\mathcal{H}} + b^*\langle f, h\rangle_{\mathcal{H}}, \\ \langle f, g\rangle_{\mathcal{H}}^* &= \langle g, f\rangle_{\mathcal{H}}, \end{aligned}$$

for all $f, g, h \in \mathcal{H}$, and $a, b \in \mathbb{C}$. In the real case, we established the symmetry condition $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \langle \kappa(\cdot, \boldsymbol{y}), \kappa(\cdot, \boldsymbol{x})\rangle_{\mathcal{H}} = \kappa(\boldsymbol{x}, \boldsymbol{y}) = \langle \kappa(\cdot, \boldsymbol{x}), \kappa(\cdot, \boldsymbol{y})\rangle_{\mathcal{H}}$. However, since in the complex case the inner product is Hermitian, the aforementioned condition is equivalent to $\kappa(\boldsymbol{x}, \boldsymbol{y}) = (\langle \kappa(\cdot, \boldsymbol{x}), \kappa(\cdot, \boldsymbol{y})\rangle_{\mathcal{H}})^*$.

As a consequence, almost all theorems that have been established in sections 1.2, **??**, 1.5 and 1.7, for the real case are actually special cases of more general ones, that involve complex functions and numbers (excluding the ones that explicitly need real numbers - e.g. Schoenberg's theorem, Proposition 1.7.2, some of the properties mentioned in section **??**, e.t.c.). There are, however, certain differences (due to the complex inner product) that must be stressed out. For example, the expansion of the kernel in terms of an orthonormal basis of $\mathcal{H}$, which is given in Theorem 1.5.2 becomes $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \sum_{s \in S} e_s(\boldsymbol{y})^* e_s(\boldsymbol{x})$, the kernels in the Proposition 1.5.4 regarding the conformal transformations become $\kappa(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x}) f(\boldsymbol{y})^*$ and $\kappa_2(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x})\kappa(\boldsymbol{x}, \boldsymbol{y}) f(\boldsymbol{y})^*$, e.t.c.

Complex reproducing kernels, that have been extensively studied by the mathematicians, are, among others, the *Szego kernels*, i.e, $\kappa(z, w) = \frac{1}{1-w^* z}$, for Hardy spaces on the unit disk, and the Bergman kernels, i.e., $\kappa(z, w) = \frac{1}{(1-w^* z)^2}$, for Bergman spaces on the unit disk, where $|z|, |w| < 1$ [31]. Another complex kernel of great importance is the *complex Gaussian kernel*:

$$\kappa_{\sigma, \mathbb{C}^d}(\boldsymbol{z}, \boldsymbol{w}) := \exp\left(-\frac{\sum_{i=1}^d (z_i - w_i^*)^2}{\sigma^2}\right), \tag{1.15}$$

defined on $\mathbb{C}^d \times \mathbb{C}^d$, where $\boldsymbol{z}, \boldsymbol{w} \in \mathbb{C}^d$, z_i denotes the i-th component of the complex vector $\boldsymbol{z} \in \mathbb{C}^d$ and exp is the extended exponential function in the complex domain. It can be shown that $\kappa_{\sigma,\mathbb{C}^d}$ is a complex valued kernel with parameter σ. Its restriction $\kappa_\sigma := \left(\kappa_{\sigma,\mathbb{C}^d}\right)_{|\mathbb{R}^d \times \mathbb{R}^d}$ is the well known *real Gaussian kernel*. An explicit description of the RKHSs of these kernels, together with some important properties can be found in [42].

1.9 Differentiation in Hilbert spaces

1.9.1 Fréchet's Differentiation

In the following sections we will develop cost functions defined on RKHS, that are suitable for minimization tasks related with adaptive filtering problems. As most minimization procedures involve computation of gradients or subgradients, we devote this section to study differentiation on Hilbert spaces. The notion of Fréchet's Differentiability, which generalizes differentiability to general Hilbert spaces, lies at the core of this analysis.

Definition 1.9.1. *(Fréchet's Differential)*
Let $\mathcal{H}$ be a Hilbert space on a field $\mathbb{F}$ (typically $\mathbb{R}$ or $\mathbb{C}$), $T : \mathcal{H} \to \mathbb{F}$ an operator and $f \in \mathcal{H}$. The operator T is said to be Fréchet differentiable at f, if there exists a $\theta \in \mathcal{H}$ such that

$$\lim_{\|h\|_{\mathcal{H}} \to 0} \frac{T(f+h) - T(f) - \langle h, \theta \rangle_{\mathcal{H}}}{\|h\|_{\mathcal{H}}} = 0, \tag{1.16}$$

where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ is the dot product of the Hilbert space $\mathcal{H}$ and $\| \cdot \|_{\mathcal{H}} = \sqrt{\langle \cdot, \cdot \rangle_{\mathcal{H}}}$ is the induced norm.

The element $\theta \in \mathcal{H}$ is called the gradient of the operator at f, and is usually denoted as $\nabla T(f)$. This relates to the standard gradient operator known by Calculus in Euclidean spaces. The Fréchet's Differential is also known as Strong Differential. There is also a weaker definition of Differentiability, named Gâteaux's Differential (or Weak Differential), which is a generalization of the directional derivative. The Gâteaux differential $dT(f, \psi) \in \mathbb{F}$ of T at $f \in \mathcal{H}$ in the direction $\psi \in \mathcal{H}$ is defined as

$$dT(f, \psi) = \lim_{\epsilon \to 0} \frac{T(f + \epsilon\psi) - T(f_0)}{\epsilon}. \tag{1.17}$$

In the following, whenever we are referring to a derivative or a gradient we will mean the one produced by Fréchet's notion of differentiability. The interested reader is addressed to [14, 23, 3, 18, 34, 44], (amongst others) for a more detailed discussion on the subject. The well known properties of the derivative of a real valued function of one variable, which are known from elementary Calculus, apply to the Fréchet's derivatives as well. Below we summarize some of these properties. For the first three we consider the operators $T_1, T_2 : \mathcal{H} \to \mathbb{F}$ differentiable at $f \in \mathcal{H}$ and $\lambda \in F$:

1. **Sum.** $\nabla(T_1 + T_2)(f) = \nabla T_1(f) + \nabla T_2(f)$.
2. **Scalar Product.** $\nabla(\lambda T_1)(f) = \lambda \nabla T_1(f)$.
3. **Product Rule.** $\nabla(T_1 \cdot T_2)(f) = T_2(f) \cdot \nabla T_1(f) + T_1(f) \cdot \nabla T_2(f)$.
4. **Chain Rule.** Consider $T_1 : \mathcal{H} \to \mathbb{F}$ differentiable at $f \in \mathcal{H}$ and $T_2 : \mathbb{F} \to \mathbb{F}$ differentiable at $y = T_1(f) \in \mathbb{F}$, then $\nabla(T_2 \circ T_1)(f) = T_2'(T_1(f)) \cdot \nabla T_1(f)$.

The following simple examples demonstrate the differentiation procedure in arbitrary spaces.

Example 1.9.1. *Consider the real Hilbert space $\mathcal{H}$, with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$, and $T : H \to \mathbb{R} : T(f) = \langle f, \psi \rangle_{\mathcal{H}}$, where $\psi \in \mathcal{H}$ fixed. We can easily show (using Fréchet's definition) that T is differentiable at any $f \in \mathcal{H}$ and that $\nabla T(f) = \psi$.*

Example 1.9.2. *Consider the real Hilbert space $\mathcal{H}$, with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$, and $T : \mathcal{H} \to \mathbb{R} : T(f) = \langle f, f \rangle_{\mathcal{H}}$. We can easily show (using Fréchet's definition) that T is differentiable at any $f \in \mathcal{H}$ and that $\nabla T(f) = 2f$.*

For real valued convex functions defined on Hilbert spaces the gradient at f satisfies the well known first order condition:

$$T(g) \geq T(f) + \langle g - f, \nabla T(f) \rangle.$$

for all $h \in \mathcal{H}$. This condition has a simple geometric meaning when T is finite at f: it says that the graph of the affine function $Q(g) = T(f) + \langle g - f, \nabla T(f) \rangle$ is a non-vertical supporting hyperplane to the convex set epi T (epi T denotes the epigraph of T, i.e. the set $\{(f, y) : f \in \mathcal{H}, y \in \mathbb{R} : T(f) \leq y\}$) at $(f, T(f))$. This is one of the reasons why the notion of gradient is so important in optimization problems. If T is not differentiable at f, we can still construct such a hyperplane using the subgradient.

Definition 1.9.2. *Let $T : \mathcal{H} \to \overline{\mathbb{R}}$ be a convex function defined on a Hilbert space $\mathcal{H}$. A vector $\theta \in \mathcal{H}$ is said to be the subgradient of T at f, if*

$$T(g) \geq T(f) + \langle g - f, \theta \rangle_{\mathcal{H}}.$$

The set of all subgradients of T at f is called the subdifferential of T at f and is denoted by $\partial T(f)$. If $f \in (\mathrm{dom}\, T)^0$ then $\partial T(f)$ is a non-empty bounded set (Theorem 23.4 of [34]), where dom T *denotes the effective domain of the function T (i.e. the set $\{x \in H : T(x) < +\infty\}$) and $(\mathrm{dom}\, T)^0$ the interior of the set* dom T.

The notion of Fréchet's differentiability may be extended to include also partial derivatives. Consider the operator $T : \mathcal{H}^\mu \to \mathbb{F}$ defined on the Hilbert space $\mathcal{H}^\mu$ with corresponding inner product:

$$\langle \boldsymbol{f}, \boldsymbol{g} \rangle_{\mathcal{H}^\mu} = \sum_{\iota=1}^{\mu} \langle f_\iota, g_\iota \rangle_{\mathcal{H}},$$

where $\boldsymbol{f} = (f_1, f_2, \dots f_\mu)$, $\boldsymbol{g} = (g_1, g_2, \dots g_\mu)$. T is said to be *Fréchet differentiable* at $\boldsymbol{f}$ with respect to f_ι, iff there exists a $\theta \in \mathcal{H}$, such that

$$\lim_{\|h\|_{\mathcal{H}} \to 0} \frac{T(\boldsymbol{f}_0 + [h]_\iota) - T(\boldsymbol{f}_0) - \langle [h]_\iota, \theta \rangle_{\mathcal{H}}}{\|h\|_{\mathcal{H}}} = 0, \tag{1.18}$$

where $[h]_\iota = (0, 0, \dots, 0, h, 0, \dots, 0)$, is the element of H^μ with zero entries everywhere, except at place ι. The element θ is called the gradient of T at $\boldsymbol{f}$ with respect to f_ι and it is denoted by $\theta = \nabla_\iota T(\boldsymbol{f})$.

1.9.2 Wirtinger's Calculus

Complex-valued signals arise frequently in applications as diverse as communications, biomedicine, radar, e.t.c. The complex domain not only provides a convenient and elegant representation for these signals, but also a natural way to preserve their characteristics and to handle transformations that need to be performed. In such tasks, one needs to handle operators defined in complex spaces and exploit an alternative path to differentiation, as the complex derivative is somewhat strict. For example, in minimization tasks the involved costs functions, although real valued, will have complex arguments. Such operators are, by definition, non-holomorphic, as all non-constant real valued functions do not obey the Cauchy-Riemann conditions. As a consequence, the complex derivative cannot be exploited. The alternative path is to restate the problem in a real space of double dimensionality (as $\mathbb{C}^\nu$ can be identified with $\mathbb{R}^{2\nu}$ by a 1-1 correspondence) and exploit the standard real derivatives. However, this approach is, usually, tedious and can lead to cumbersome expressions, while the advantages of complex representation are lost. An alternative, equivalent, approach is to use the notions of Wirtinger's calculus.

Wirtinger's calculus [46] has, recently, attracted attention in the signal processing community mainly in the context of complex adaptive filtering [1, 24, 19, 8, 10, 9, 30], as a means to compute, in an elegant way, gradients of real valued cost functions that are defined on complex domains ($\mathbb{C}^\nu$). Although, as we have already mentioned, these gradients may be derived equivalently in the traditional way, if one splits the complex variables to the real and imaginary parts and considers the corresponding partial derivatives, Wirtinger's toolbox usually requires much less algebra and involves simpler expressions. It is based on simple rules and principles, which bear a great resemblance to the rules of the standard complex derivative, and

it greatly simplifies the calculations of the respective derivatives; these are evaluated by treating z and z^* independently using traditional differentiation rules. In [8], the notion of Wirtinger's calculus was extended to general complex Hilbert spaces, exploiting the notion of the Fréchet's differentiability, providing the tool to compute the gradients that are needed to develop kernel-based algorithms for treating complex data.

In the traditional setting, the pair of Wirtinger's gradients are defined as follows. Let $f : \mathbb{C} \to \mathbb{C}$ be a complex function defined on $\mathbb{C}$. Obviously, such a function may be regarded as either defined on $\mathbb{R}^2$ or $\mathbb{C}$ (i.e., $f(z) = f(x + iy) = f(x, y)$). Furthermore, it may be regarded as either a complex valued function, $f(x, y) = u(x, y) + iv(x, y)$ or as a vector valued function $f(x, y) = (u(x, y), v(x, y))$. We will say that f is *differentiable in the real sense*, if u and v are differentiable. It turns out that, when the complex structure is considered, the real derivatives may be described using an equivalent and more elegant formulation, which bears a surprising resemblance with the complex derivative. In fact, if the function f is *differentiable in the complex sense* (i.e. the complex derivative exists), the developed derivatives coincide with the complex ones. Although this methodology is known for some time in the German speaking countries and it has been applied to practical applications [11, 13], only recently has attracted the attention of the signal processing community, mostly in the context of works that followed Picinbono's paper on widely linear estimation filters [32].

The *Wirtinger's derivative* (or *W-derivative* for short) of f at a point c is defined as follows

$$\frac{\partial f}{\partial z}(c) = \frac{1}{2}\left(\frac{\partial f}{\partial x}(c) - i\frac{\partial f}{\partial y}(c)\right) = \frac{1}{2}\left(\frac{\partial u}{\partial x}(c) + \frac{\partial v}{\partial y}(c)\right) + \frac{i}{2}\left(\frac{\partial v}{\partial x}(c) - \frac{\partial u}{\partial y}(c)\right). \tag{1.19}$$

The *conjugate Wirtinger's derivative* (or *CW-derivative* for short) of f at c is defined by:

$$\frac{\partial f}{\partial z^*}(c) = \frac{1}{2}\left(\frac{\partial f}{\partial x}(c) + i\frac{\partial f}{\partial y}(c)\right) = \frac{1}{2}\left(\frac{\partial u}{\partial x}(c) - \frac{\partial v}{\partial y}(c)\right) + \frac{i}{2}\left(\frac{\partial v}{\partial x}(c) + \frac{\partial u}{\partial y}(c)\right). \tag{1.20}$$

The usual properties of the standard derivative can be proven for this case also (see [17, 7, 1, 19]). In a nutshell, one might easily compute the W and CW derivatives of any complex function f, which is written in terms of z and z^*, following the following simple tricks:

- *To compute the W-derivative of a function f, which is expressed in terms of z and z^*, apply the usual differentiation rules considering z^* as a constant.*
- *To compute the CW-derivative of a function f, which is expressed in terms of z and z^*, apply the usual differentiation rules considering z as a constant.*

Following a similar rationale, we can define Wirtinger-type gradients in general Hilbert spaces. Consider the complex Hilbert space $\mathbb{H} = \mathcal{H} + i\mathcal{H}$ and the function $\boldsymbol{T} : A \subseteq \mathbb{H} \to \mathbb{C}$, $\boldsymbol{T}(\boldsymbol{f}) = \boldsymbol{T}(u_{\boldsymbol{f}} + iv_{\boldsymbol{f}}) = T_r(u_{\boldsymbol{f}}, v_{\boldsymbol{f}}) + iT_i(u_{\boldsymbol{f}}, v_{\boldsymbol{f}})$, defined on an open subset A of $\mathbb{H}$, where $u_{\boldsymbol{f}}, v_{\boldsymbol{f}} \in \mathcal{H}$ and T_r, T_i are real valued functions defined on H^2. Any such function, $\boldsymbol{T}$, may be regarded as defined either on a subset of $\mathbb{H}$, or on a subset of $\mathcal{H}^2$. Moreover, $\boldsymbol{T}$ may be regarded either as a complex valued function, or as a vector valued function, which takes values in $\mathbb{R}^2$. Therefore, we may equivalently write:

$$\begin{aligned}\boldsymbol{T}(\boldsymbol{f}) &= \boldsymbol{T}(u_{\boldsymbol{f}} + iv_{\boldsymbol{f}}) = T_r(u_{\boldsymbol{f}}, v_{\boldsymbol{f}}) + iT_i(u_{\boldsymbol{f}}, v_{\boldsymbol{f}}),\\ \boldsymbol{T}(\boldsymbol{f}) &= (T_r(u_{\boldsymbol{f}}, v_{\boldsymbol{f}}), T_i(u_{\boldsymbol{f}}, v_{\boldsymbol{f}}))^T.\end{aligned}$$

In the following, we will often change the notation according to the specific problem and consider any element of $\boldsymbol{f} \in \mathbb{H}$ defined either as $\boldsymbol{f} = u_{\boldsymbol{f}} + iv_{\boldsymbol{f}} \in \mathbb{H}$, or as $\boldsymbol{f} = (u_{\boldsymbol{f}}, v_{\boldsymbol{f}}) \in \mathcal{H}^2$. In a similar manner, any complex number may be regarded as either an element of $\mathbb{C}$, or as an element of $\mathbb{R}^2$. We say that $\boldsymbol{T}$ is *complex Fréchet differentiable* at $\boldsymbol{c} \in \mathbb{H}$, if there exists $\boldsymbol{\theta} \in \mathbb{H}$ such that:

$$\lim_{\|\boldsymbol{h}\|_{\mathbb{H}} \to 0} \frac{\boldsymbol{T}(\boldsymbol{c} + \boldsymbol{h}) - \boldsymbol{T}(\boldsymbol{c}) - \langle \boldsymbol{h}, \boldsymbol{\theta} \rangle_{\mathbb{H}}}{\|\boldsymbol{h}\|_{\mathbb{H}}} = 0. \tag{1.21}$$

Then $\boldsymbol{\theta}$ is called the *complex gradient* of $\boldsymbol{T}$ at $\boldsymbol{c}$ and it is denoted as $\nabla \boldsymbol{T}(\boldsymbol{c})$. This definition, although similar with the typical Fréchet's derivative, exploits the complex structure of $\mathbb{H}$. More specifically, the complex inner product, that appears in the definition, forces a great deal of structure on $\boldsymbol{T}$. Similarly to the case of ordinary complex functions, it is this simple fact that gives birth to all the important strong properties of the complex derivative. We begin our study by exploring the relations between the complex Fréchet's derivative and the real Fréchet's derivatives. In the following, we will say that $\boldsymbol{T}$ is *Fréchet differentiable in the complex sense*, if the complex derivative exists, and that $\boldsymbol{T}$ is *Fréchet differentiable in the real sense*, if its real Fréchet's derivative exists (i.e., $\boldsymbol{T}$ is regarded as a vector valued operator $\boldsymbol{T} : \mathcal{H}^2 \to \mathcal{H}^2$). Equations (1.22) are the *Cauchy Riemann conditions* with respect to the Fréchet's notion of differentiability:

$$\nabla_u T_r(f_1, f_2) = \nabla_v T_i(f_1, f_2), \quad \nabla_v T_r(f_1, f_2) = -\nabla_u T_i(f_1, f_2). \tag{1.22}$$

Similar to the simple case of complex valued functions, they provide a necessary and sufficient condition, for a complex operator $\boldsymbol{T}$, that is defined on $\mathbb{H}$, to be differentiable in the complex sense, provided that $\boldsymbol{T}$ is differentiable in the real sense.

If the Fréchet's Cauchy Riemann conditions are not satisfied for an operator $\boldsymbol{T}$, then the complex Fréchet's derivative does not exist. In this case we may define Wirtinger-type gradients as follows. We define the *Fréchet-Wirtinger's gradient* (or *W-gradient* for short) of $\boldsymbol{T}$ at $\boldsymbol{c}$ as

$$\nabla_{\boldsymbol{f}} \boldsymbol{T}(\boldsymbol{c}) = \frac{1}{2} \left(\nabla_1 \boldsymbol{T}(\boldsymbol{c}) - i \nabla_2 \boldsymbol{T}(\boldsymbol{c}) \right) = \frac{1}{2} \left(\nabla_u T_r(\boldsymbol{c}) + \nabla_v T_i(\boldsymbol{c}) \right) + \frac{i}{2} \left(\nabla_u T_i(\boldsymbol{c}) - \nabla_v T_r(\boldsymbol{c}) \right). \tag{1.23}$$

Consequently, the *conjugate Fréchet-Wirtinger's gradient* (or *CW-gradient* for short) is defined by:

$$\nabla_{\boldsymbol{f}^*} \boldsymbol{T}(\boldsymbol{c}) = \frac{1}{2} \left(\nabla_1 \boldsymbol{T}(\boldsymbol{c}) + i \nabla_2 \boldsymbol{T}(\boldsymbol{c}) \right) = \frac{1}{2} \left(\nabla_u T_r(\boldsymbol{c}) - \nabla_v T_i(\boldsymbol{c}) \right) + \frac{i}{2} \left(\nabla_u T_i(\boldsymbol{c}) + \nabla_v T_r(\boldsymbol{c}) \right). \tag{1.24}$$

Note, that both the W-derivative and the CW-derivative exist, if $\boldsymbol{T}$ is Fréchet differentiable in the real sense. In view of these new definitions, we can derive a first-order Taylor expansion formula around $\boldsymbol{c}$:

$$\boldsymbol{T}(\boldsymbol{c} + \boldsymbol{h}) = \boldsymbol{T}(\boldsymbol{c}) + \langle \boldsymbol{h}, (\nabla_{\boldsymbol{f}} \boldsymbol{T}(\boldsymbol{c}))^* \rangle_{\mathbb{H}} + \left\langle \boldsymbol{h}^*, \left(\nabla_{\boldsymbol{f}^*} \boldsymbol{T}(\boldsymbol{c}) \right)^* \right\rangle_{\mathbb{H}} + o(\|\boldsymbol{h}\|_{\mathbb{H}}). \tag{1.25}$$

From these definitions, several properties can be derived:

1. If $\boldsymbol{T}(\boldsymbol{f})$ is $\boldsymbol{f}$-analytic at $\boldsymbol{c}$ (i.e., it has a Taylor series expansion with respect to $\boldsymbol{f}$ around $\boldsymbol{c}$), then its W-gradient at $\boldsymbol{c}$ degenerates to the standard complex gradient and its CW-gradient vanishes, i.e., $\nabla_{\boldsymbol{f}^*} \boldsymbol{T}(\boldsymbol{c}) = \boldsymbol{0}$.
2. If $\boldsymbol{T}(\boldsymbol{f})$ is $\boldsymbol{f}^*$-analytic at $\boldsymbol{c}$ (i.e., it has a Taylor series expansion with respect to $\boldsymbol{f}^*$ around $\boldsymbol{c}$), then $\nabla_{\boldsymbol{f}} \boldsymbol{T}(\boldsymbol{c}) = \boldsymbol{0}$.
3. $(\nabla_{\boldsymbol{f}} \boldsymbol{T}(\boldsymbol{c}))^* = \nabla_{\boldsymbol{f}^*} \boldsymbol{T}^*(\boldsymbol{c})$.
4. $\left(\nabla_{\boldsymbol{f}^*} \boldsymbol{T}(\boldsymbol{c}) \right)^* = \nabla_{\boldsymbol{f}} \boldsymbol{T}^*(\boldsymbol{c})$.
5. If $\boldsymbol{T}$ is real valued, then $(\nabla_{\boldsymbol{f}} \boldsymbol{T}(\boldsymbol{c}))^* = \nabla_{\boldsymbol{f}^*} \boldsymbol{T}(\boldsymbol{c})$.
6. The first order Taylor expansion around $\boldsymbol{f} \in \mathbb{H}$ is given by

$$\boldsymbol{T}(\boldsymbol{f} + \boldsymbol{h}) = \boldsymbol{T}(\boldsymbol{f}) + \langle \boldsymbol{h}, (\nabla_{\boldsymbol{f}} \boldsymbol{T}(\boldsymbol{f}))^* \rangle_{\mathbb{H}} + \langle \boldsymbol{h}^*, \left(\nabla_{\boldsymbol{f}^*} \boldsymbol{T}(\boldsymbol{f}) \right)^* \rangle_{\mathbb{H}}.$$

7. If $\boldsymbol{T}(\boldsymbol{f}) = \langle \boldsymbol{f}, \boldsymbol{w} \rangle_{\mathbb{H}}$, then $\nabla_{\boldsymbol{f}} \boldsymbol{T}(\boldsymbol{c}) = \boldsymbol{w}^*$, $\nabla_{\boldsymbol{f}^*} \boldsymbol{T}(\boldsymbol{c}) = \boldsymbol{0}$, for every $\boldsymbol{c}$.
8. If $\boldsymbol{T}(\boldsymbol{f}) = \langle \boldsymbol{w}, \boldsymbol{f} \rangle_{\mathbb{H}}$, then $\nabla_{\boldsymbol{f}} \boldsymbol{T}(\boldsymbol{c}) = \boldsymbol{0}$, $\nabla_{\boldsymbol{f}^*} \boldsymbol{T}(\boldsymbol{c}) = \boldsymbol{w}$, for every $\boldsymbol{c}$.
9. If $\boldsymbol{T}(\boldsymbol{f}) = \langle \boldsymbol{f}^*, \boldsymbol{w} \rangle_{\mathbb{H}}$, then $\nabla_{\boldsymbol{f}} \boldsymbol{T}(\boldsymbol{c}) = \boldsymbol{0}$, $\nabla_{\boldsymbol{f}^*} \boldsymbol{T}(\boldsymbol{c}) = \boldsymbol{w}^*$, for every $\boldsymbol{c}$.
10. If $\boldsymbol{T}(\boldsymbol{f}) = \langle \boldsymbol{w}, \boldsymbol{f}^* \rangle_{\mathbb{H}}$, then $\nabla_{\boldsymbol{f}} \boldsymbol{T}(\boldsymbol{c}) = \boldsymbol{w}$, $\nabla_{\boldsymbol{f}^*} \boldsymbol{T}(\boldsymbol{c}) = \boldsymbol{0}$, for every $\boldsymbol{c}$.

11. Linearity: If $\boldsymbol{T}, \boldsymbol{S} : \mathbb{H} \to \mathbb{C}$ are Fréchetdifferentiable in the real sense at $\boldsymbol{c} \in \mathbb{H}$ and $\alpha, \beta \in \mathbb{C}$, then

$$\nabla_{\boldsymbol{f}}(\alpha\boldsymbol{T} + \beta\boldsymbol{S})(\boldsymbol{c}) = \alpha\nabla_{\boldsymbol{f}}\boldsymbol{T}(\boldsymbol{c}) + \beta\nabla_{\boldsymbol{f}}\boldsymbol{S}(\boldsymbol{c})$$
$$\nabla_{\boldsymbol{f}^*}(\alpha\boldsymbol{T} + \beta\boldsymbol{S})(\boldsymbol{c}) = \alpha\nabla_{\boldsymbol{f}^*}\boldsymbol{T}(\boldsymbol{c}) + \beta\nabla_{\boldsymbol{f}^*}\boldsymbol{S}(\boldsymbol{c}).$$

12. Product Rule: If $\boldsymbol{T}, \boldsymbol{S} : \mathbb{H} \to \mathbb{C}$ are Fréchetdifferentiable in the real sense at $\boldsymbol{c} \in \mathbb{H}$, then:

$$\nabla_{\boldsymbol{f}}(\boldsymbol{T} \cdot \boldsymbol{S})(\boldsymbol{c}) = \nabla_{\boldsymbol{f}}\boldsymbol{T}(\boldsymbol{c})\boldsymbol{S}(\boldsymbol{c}) + \boldsymbol{T}(\boldsymbol{c})\nabla_{\boldsymbol{f}}\boldsymbol{S}(\boldsymbol{c}),$$
$$\nabla_{\boldsymbol{f}^*}(\boldsymbol{T} \cdot \boldsymbol{S})(\boldsymbol{c}) = \nabla_{\boldsymbol{f}^*}\boldsymbol{T}(\boldsymbol{c})\boldsymbol{S}(\boldsymbol{c}) + \boldsymbol{T}(\boldsymbol{c})\nabla_{\boldsymbol{f}^*}\boldsymbol{S}(\boldsymbol{c}).$$

13. Division Rule: If $\boldsymbol{T}, \boldsymbol{S} : \mathbb{H} \to \mathbb{C}$ are Fréchetdifferentiable in the real sense at $\boldsymbol{c} \in \mathbb{H}$ and $\boldsymbol{S}(\boldsymbol{c}) \neq 0$, then:

$$\nabla_{\boldsymbol{f}}\left(\frac{\boldsymbol{T}}{\boldsymbol{S}}\right)(\boldsymbol{c}) = \frac{\nabla_{\boldsymbol{f}}\boldsymbol{T}(\boldsymbol{c})\boldsymbol{S}(\boldsymbol{c}) - \boldsymbol{T}(\boldsymbol{c})\nabla_{\boldsymbol{f}}\boldsymbol{S}(\boldsymbol{c})}{\boldsymbol{S}^2(\boldsymbol{c})},$$
$$\nabla_{\boldsymbol{f}^*}\left(\frac{\boldsymbol{T}}{\boldsymbol{S}}\right)(\boldsymbol{c}) = \frac{\nabla_{\boldsymbol{f}^*}\boldsymbol{T}(\boldsymbol{c})\boldsymbol{S}(\boldsymbol{c}) - \boldsymbol{T}(\boldsymbol{c})\nabla_{\boldsymbol{f}^*}\boldsymbol{S}(\boldsymbol{c})}{\boldsymbol{S}^2(\boldsymbol{c})}.$$

14. Chain Rule: If $\boldsymbol{T} : \mathbb{H} \to \mathbb{C}$ is Fréchetdifferentiable at $\boldsymbol{c} \in \mathbb{H}$, $\boldsymbol{S} : \mathbb{C} \to \mathbb{C}$ is differentiable in the real sense at $\boldsymbol{T}(\boldsymbol{c}) \in \mathbb{C}$, then:

$$\nabla_{\boldsymbol{f}}\boldsymbol{S} \circ \boldsymbol{T}(\boldsymbol{c}) = \frac{\partial \boldsymbol{S}}{\partial z}(\boldsymbol{T}(c))\nabla_{\boldsymbol{f}}\boldsymbol{T}(\boldsymbol{c}) + \frac{\partial \boldsymbol{S}}{\partial z^*}(\boldsymbol{T}(\boldsymbol{c}))\nabla_{\boldsymbol{f}}(\boldsymbol{T}^*)(\boldsymbol{c}),$$
$$\nabla_{\boldsymbol{f}^*}\boldsymbol{S} \circ \boldsymbol{T}(\boldsymbol{c}) = \frac{\partial \boldsymbol{S}}{\partial z}(\boldsymbol{T}(\boldsymbol{c}))\nabla_{\boldsymbol{f}^*}\boldsymbol{T}(\boldsymbol{c}) + \frac{\partial \boldsymbol{S}}{\partial z^*}(\boldsymbol{T}(\boldsymbol{c}))\nabla_{\boldsymbol{f}^*}(\boldsymbol{T}^*)(\boldsymbol{c}).$$

A complete list of the derived properties, together with the proofs are given in [7, 8].

An important consequence of the previous properties is that if T is a real valued operator defined on $\mathbb{H}$, as $(\nabla_{\boldsymbol{f}}T(\boldsymbol{c}))^* = \nabla_{\boldsymbol{f}^*}T(\boldsymbol{c})$, its first order Taylor's expansion is given by:

$$\begin{aligned} T(\boldsymbol{f}+\boldsymbol{h}) &= T(\boldsymbol{f}) + \langle \boldsymbol{h}, (\nabla_{\boldsymbol{f}}\boldsymbol{T}(\boldsymbol{f}))^* \rangle_{\mathbb{H}} + \langle \boldsymbol{h}^*, \left(\nabla_{\boldsymbol{f}^*}T(\boldsymbol{f})\right)^* \rangle_{\mathbb{H}} \\ &= T(\boldsymbol{f}) + \langle \boldsymbol{h}, \nabla_{\boldsymbol{f}^*}\boldsymbol{T}(\boldsymbol{f}) \rangle_{\mathbb{H}} + \left(\langle \boldsymbol{h}, \nabla_{\boldsymbol{f}^*}T(\boldsymbol{f}) \rangle_{\mathbb{H}}\right)^* = T(\boldsymbol{f}) + 2 \cdot \Re\left[\langle \boldsymbol{h}, \nabla_{\boldsymbol{f}^*}T(\boldsymbol{f}) \rangle_{\mathbb{H}}\right]. \end{aligned}$$

However, in view of the Cauchy Riemann inequality we have:

$$\Re\left[\langle \boldsymbol{h}, \nabla_{\boldsymbol{f}^*}T(\boldsymbol{f}) \rangle_{\mathbb{H}}\right] \leq \left|\langle \boldsymbol{h}, \nabla_{\boldsymbol{f}^*}T(\boldsymbol{f}) \rangle_{\mathbb{H}}\right| \leq \|\boldsymbol{h}\|_{\mathbb{H}} \cdot \|\nabla_{\boldsymbol{f}^*}T(\boldsymbol{f})\|_{\mathbb{H}}.$$

The equality in the above relationship holds if $\boldsymbol{h} \uparrow\uparrow \nabla_{\boldsymbol{f}^*}T$ (where the notation $\uparrow\uparrow$ denotes that $\boldsymbol{h}$ and $\nabla_{\boldsymbol{f}^*}T$ have the same direction, i.e., there is a $\lambda > 0$, such that $\boldsymbol{h} = \lambda\nabla_{\boldsymbol{f}^*}T$). Hence, the direction of increase of T is $\nabla_{\boldsymbol{f}^*}T(\boldsymbol{f})$. Therefore, any gradient descent based algorithm minimizing $T(\boldsymbol{f})$ is based on the update scheme:

$$\boldsymbol{f}_n = \boldsymbol{f}_{n-1} - \mu \cdot \nabla_{\boldsymbol{f}^*}T(\boldsymbol{f}_{n-1}). \tag{1.26}$$

Assuming differentiability of T, a standard result from Calculus states that a necessary condition for a point $\boldsymbol{c}$ to be an optimum (in the sense that $T(\boldsymbol{f})$ is minimized or maximized) is that this point is a stationary point of T, i.e., the Fréchet's partial derivatives of T at $\boldsymbol{c}$ vanish. In the context of Wirtinger's calculus, we have the following obvious corresponding result.

Proposition 1.9.1. *If the function $T : A \subseteq \mathbb{H} \to \mathbb{R}$ is Fréchet differentiable at $\boldsymbol{c}$ in the real sense, then a necessary condition for a point $\boldsymbol{c}$ to be a local optimum (in the sense that $T(\boldsymbol{c})$ is minimized or maximized) is that either the W, or the CW gradient vanishes.*

Chapter 2

Least Squares Learning Algorithms

2.1 Least Mean Squares

Consider the sequence of examples $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n), \ldots\}$, where $\boldsymbol{x}_n \in X \subset \mathbb{R}^\nu$, and $y_n \in \mathbb{R}$ for $n \in \mathbb{N}$, and a parametric class of functions $\mathcal{F} = \{f_{\boldsymbol{\theta}} : X \to \mathbb{R}, \boldsymbol{\theta} \in \mathbb{R}^\mu\}$. The goal of a typical adaptive learning task is to infer an input-output relationship $f_{\boldsymbol{\theta}}$ from the parametric class of functions $\mathcal{F}$, based on the given data, so that to minimize a certain loss function, $\mathcal{L}(\boldsymbol{\theta})$, that measures the error between the actual output, y_n, and the estimated output, $\hat{y}_n = f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)$, at iteration n.

Least mean squares (LMS) algorithms are a popular class of adaptive learning systems that adopt the mean squares error (i.e., the mean squared difference between the desired and the actual signal) to produce the solution's coefficients $\boldsymbol{\theta}$. Its most typical form was invented in the 60's by professor Bernard Widrow and his Ph.D. student, Ted Hoff [45]. In the standard *least mean squares* (LMS) setting, one adopts the class of linear functions, i.e., $\mathcal{F} = \{f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^T\boldsymbol{x}, \boldsymbol{\theta} \in \mathbb{R}^\nu\}$ and employs the mean square error (MSE), $\mathcal{L}(\boldsymbol{\theta}) = E\left[|y_n - f_{\boldsymbol{\theta}}(\boldsymbol{x})|^2\right]$, as the loss function. To this end, the gradient descent rationale is employed and at each time instant, $n = 1, 2, \ldots, N$, the gradient of the mean square error, i.e., $\nabla\mathcal{L}(\boldsymbol{\theta}) = -2E[e_n\boldsymbol{x}_n]$, is estimated via its current measurement, i.e., $E[e_n\boldsymbol{x}_n] \approx e_n\boldsymbol{x}_n$, where $e_n = y_n - \boldsymbol{\theta}_{n-1}^T\boldsymbol{x}_n$ is the a-priori error at instance $n = 2, \ldots, N$. This leads to the step update equation

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + \mu e_n \boldsymbol{x}_n. \tag{2.1}$$

Assuming that $\boldsymbol{\theta}_0 = \mathbf{0}$ (i.e., the initial value of $\boldsymbol{\theta}$ is $\mathbf{0}$), a repeated application of (2.1) gives

$$\boldsymbol{\theta}_n = \mu \sum_{i=1}^{n} e_i \boldsymbol{x}_i, \tag{2.2}$$

while the predicted output of the learning system at iteration n becomes

$$\hat{y}_n = f_{\boldsymbol{\theta}_{n-1}}(\boldsymbol{x}_n) = \mu \sum_{i=1}^{n-1} e_i \boldsymbol{x}_i^T \boldsymbol{x}_n. \tag{2.3}$$

There are a number of convergence criteria for the LMS algorithm [16]. Among the most popular ones are:

- The *convergence of the mean* property, i.e. $E[\boldsymbol{\epsilon}_n] \to 0$, as $n \to \infty$, where $\boldsymbol{\epsilon}_n = \boldsymbol{\theta}_n - \boldsymbol{\theta}_0$ and $\boldsymbol{\theta}_0$ is the Wiener solution. Unfortunately, this is of little practical use, as a sequence of zero mean arbitrary random numbers converges in this sense.
- The *convergence in the mean* property, i.e. $E[\|\boldsymbol{\epsilon}_n\|] \to 0$, as $n \to \infty$.
- The *convergence in the mean square* property: If the true input-output relation is linear, i.e., $y_n = \boldsymbol{\theta}_*^T\boldsymbol{x}_n$ and $\boldsymbol{x}_n$ is a weak stationary process, then $J(n) = E[|e_n|^2] \to$ constant, as $n \to \infty$, provided that μ satisfies the condition $0 < \mu < \frac{2}{\lambda_{\max}}$, where $\lambda_{\max}$ is the largest eigenvalue of the of the correlation matrix $R = E[\boldsymbol{x}_n\boldsymbol{x}_n^T]$. In practice, one usually uses $0 < \mu < \frac{2}{\mathrm{tr}(R)}$.

- The *Misadjustment* property: The Misadjustment ratio, i.e. $\mathcal{M} = \frac{J(\infty)-J_{\min}}{J_{\min}}$, where $J(\infty) = \lim_n J(n) = \lim_n E[|e_n|^2]$ and $J_{\min}$ is the minimum mean squared error produced by the optimum Wiener filter, is equal to

$$\mathcal{M} = \sum_{j=1}^{M} \frac{\mu\lambda_i}{2-\mu\lambda_i},$$

  provided that the step update parameter μ satisfies $\sum_{i=1}^{M} \frac{2\lambda_i}{2-\mu\lambda_i} < 1$, where $\lambda_1, \ldots, \lambda_M$ are the eigenvalues of the correlation matrix R. This is a dimensionless quantity, providing a measure of closeness the LMS algorithm is to optimality in the mean square sense. The smallest $\mathcal{M}$ is (compared to 1), the more accurate the learning system is. Usually $\mathcal{M}$ is expressed as a percentage. A misadjustment of 10% is, ordinarily, considered to be satisfactory in practice.

LMS is sensitive to the scaling of the inputs $\boldsymbol{x}_n$. This makes it hard to chose a learning rate μ that guarantees the stability of the algorithm for various inputs. The Normalized LMS (NLMS) is a variant of the LMS designed to cope with this problem. It solves the problem by normalizing with the power of the input. Hence the step update equation becomes

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + \frac{\mu e_n}{|\boldsymbol{x}_n|^2}\boldsymbol{x}_n. \tag{2.4}$$

It has been proved that the optimal learning rate of the NLMS is $\mu = 1$. More on the LMS family of algorithms can be found in [16, 35].

2.2 The Kernel LMS

In kernel based algorithms, we estimate the output of the learning system by a non-linear function f of a specific RKHS, $\mathcal{H}$, which is associated to a positive definite kernel κ. The sequence of examples are transformed via the feature map $\phi : X \to \mathcal{H}$, $\phi(\boldsymbol{x}) = \kappa(\cdot, \boldsymbol{x})$ and the LMS rationale is employed on the transformed data

$$\{(\phi(\boldsymbol{x}_1), y_1), (\phi(\boldsymbol{x}_2), y_2), \ldots, (\phi(\boldsymbol{x}_n), y_n), \ldots\},$$

while the estimated output at iteration n takes the form $\hat{y}_n = \langle \phi(\boldsymbol{x}_n), f\rangle_{\mathcal{H}}$, for some $f \in \mathcal{H}$. Although this is a linear algorithm in the RKHS $\mathcal{H}$, it corresponds to a non-linear processing in the Euclidean input space X. To calculate the gradient of the respective loss function, $\mathcal{L}(f) = E\left[|y_n - \langle \phi(\boldsymbol{x}_n), f\rangle_{\mathcal{H}}|^2\right]$, the Fréchet's notion of differentiability is adopted, leading to the estimation: $\nabla\mathcal{L}(f) = -2E[e_n\phi(\boldsymbol{x}_n)]$, which according to the LMS rationale, at every time instant it is approximated by its current measurement, i.e., $-2e_n\phi(\boldsymbol{x}_n)$. Consequently, following a gradient descent logic, the step update equation of the KLMS is given by

$$f_n = f_{n-1} + \mu e_n \phi(\boldsymbol{x}_n). \tag{2.5}$$

Assuming that $f_0 = \mathbf{0}$ (i.e., the zero function), a repeated application of (2.5) gives

$$f_n = \mu \sum_{i=1}^{n} e_i \phi(\boldsymbol{x}_i), \tag{2.6}$$

and the output of the filter at iteration n becomes

$$\begin{aligned} \hat{y}_n = \langle \phi(\boldsymbol{x}_n), f_{n-1}\rangle_{\mathcal{H}} &= \mu \sum_{i=1}^{n-1} e_i \langle \phi(\boldsymbol{x}_n), \phi(\boldsymbol{x}_i)\rangle_{\mathcal{H}} \\ &= \mu \sum_{i=1}^{n-1} e_i \kappa(\boldsymbol{x}_i, \boldsymbol{x}_n). \end{aligned} \tag{2.7}$$

Equation (2.6) gives the solution of the KLMS at time n. This is an expansion (in the RKHS $\mathcal{H}$) in terms of the feature map function computed at each training point up to time n. Observe that, alternatively, one

could derive equation (2.7), simply by applying the kernel trick to (2.3). It is important to emphasize that, in contrast to the standard LMS, where the solution is a vector of fixed dimension, which is updated at each time instant, in KLMS the solution is a function of $\mathcal{H}$ and thus it cannot be represented by a machine. Nevertheless, as it is common in kernel-based methods, one needs not to compute the actual solution (2.6), but only the estimated output of the learning system, $\hat{y}_n$, which is given in terms of the kernel function centered at the points, $\boldsymbol{x}_i$, $i = 1, \ldots, n-1$, of the given sequence (equation (2.7)). Under this setting, the algorithm needs to store into memory two types of information: a) the centers, $\boldsymbol{x}_i$, of the expansion (2.6), which are stored into a dictionary $\mathcal{D}$ and b) the coefficients of (2.6), i.e., μe_i, which are represented by a growing vector $\boldsymbol{\theta}$. Observe, that (2.6) is inline with what we know by the Representer theorem given in section 1 (theorem 1.5.3). Algorithm 1 summarizes this procedure.

Algorithm 1 Kernel LMS (KLMS)

INPUT: $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n), \ldots$
Initialization: Set $\boldsymbol{\theta} = 0$, $\mathcal{D} = \{\}$. Select the step parameter μ and the parameters of the kernel.
for $n = 1, 2, \ldots$ **do**
if $n == 1$ **then**
$\hat{y}_n = 0$
else
Compute the learning system's output: $\hat{y}_n = \sum_{i=1}^{n-1} \theta_i \cdot \kappa(\boldsymbol{u}_i, \boldsymbol{x}_n)$.
end if
Compute the error: $e_n = y_n - \hat{y}_n$.
$\theta_n = \mu e_n$.
Add the new center $\boldsymbol{u}_n = \boldsymbol{x}_n$ to the list of centers, i.e., $\mathcal{D} = \mathcal{D} \cup \{\boldsymbol{u}_n\}$, $\boldsymbol{\theta} = (\boldsymbol{\theta}^T, \theta_n)^T$.
end for
OUTPUT: The n-dimensional vector $\boldsymbol{\theta}$ and the dictionary $\mathcal{D} = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n\}$. The solution is then given as $f = \sum_{i=1}^{n} \theta_i \kappa(\boldsymbol{u}_i, \cdot)$.

If in place of (2.5), we use its normalized version, i.e.,

$$f_n = f_{n-1} + \frac{\mu}{\kappa(\boldsymbol{x}_n, \boldsymbol{x}_n)} e_n \phi(\boldsymbol{x}_n), \tag{2.8}$$

the normalized KLMS counterpart (KNLMS) results. This comprises replacing in Algorithm 1 the step $\theta_n = \mu e_n$ by $\theta_n = \mu e_n / \gamma$, where $\gamma = \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n)$. The convergence and stability properties of KLMS is, still, a hot topic for research. One may consider that, as the KLMS is the LMS in a RKHS $\mathcal{H}$, the properties of the LMS are directly transferred to the KLMS [21]. However, we should note that the properties of the LMS have been proved for Euclidian spaces, while the RKHS that are usually used in practice are infinite dimensional Hilbert spaces. Finally, we should mention that, recently, a generalization of the KLMS, the so called complex kernel LMS, that works directly on complex RKHS has been developed [8, 10].

2.2.1 Sparsifying the solution

The main drawback of the KLMS algorithm is that a growing number of training points, $\boldsymbol{x}_n$, is involved in the estimation of the learning system's output (2.7). The set of these points can be thought of as a "dictionary", $\mathcal{D}$, which is stored into the memory, as in Algorithm 1. This is a typical situation that arises in any kernel based online learning scheme. As a consequence, increasing memory and computational resources are needed, as time evolves (the complexity of KLMS, as presented in section 2.2 is $O(n)$ at each iteration). Furthermore, it is impossible to use the KLMS in real world's applications, since the dictionary grows unbounded. In such a case, after a significant amount of iterations the expansion (2.7) will become so large that it will fill up the memory of the machine. In addition, it will require a huge amount of time to be computed, rendering the application useless. Hence, it is important to find a way to limit the size of this expansion. Several strategies have been proposed to cope with this problem. In such strategies, the dictionary of points is formed at the beginning of the algorithm and new points are inserted into it, only if they satisfy a specific rule, as in algorithm 2. Adopting such a strategy, the complexity of KLMS is reduced to $O(M_n)$, where M_n is the size of the dictionary at time instant n.

Algorithm 2 Kernel LMS (KLMS) with sparsification

INPUT: $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n), \ldots$
Initialization: Set $\boldsymbol{\theta} = 0$, $\mathcal{D} = \{\}$. Select the step parameter μ and the parameters of the kernel.

```
M = 0
for n = 1, 2, ... do
    if n == 1 then
        ŷ_n = 0
    else
        Compute the learning system's output: ŷ_n = Σ_{i=1}^{M} θ_i · κ(u_i, x_n).
    end if
    Compute the error: e_n = y_n − ŷ_n.
    θ_n = μ e_n.
    Check Sparsification Rules
    if Sparsification Rules are Satisfied then
        M = M + 1
        Add the new center u_M = x_n to the list of centers, i.e., D = D ∪ {u_n}, θ = (θ^T, θ_n)^T.
    end if
end for
```

OUTPUT: The M-dimensional vector $\boldsymbol{\theta}$ and the dictionary $\mathcal{D} = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_M\}$. The solution is then given as $f = \sum_{i=1}^{M} \theta_i \kappa(\boldsymbol{u}_i, \cdot)$.

For example, in the popular Plat's *novelty criterion* sparsification scheme [20], whenever a new data pair $(\boldsymbol{x}_n, y_n)$ is considered, a decision is immediately made of whether to add the new point, $\boldsymbol{x}_n$, to $\mathcal{D}$. The decision is reached following two simple rules:

- First, the distance of the new point, $\boldsymbol{x}_n$, from the current dictionary, $\mathcal{D}_{n-1}$, is evaluated:

$$\text{dist} = \min_{\boldsymbol{u}_k \in \mathcal{D}_{n-1}} \{\|\boldsymbol{x}_n - \boldsymbol{u}_k\|\}.$$

  If this distance is smaller than a given threshold δ_1 (i.e., the new input vector is close to a point, which is already in the existing dictionary), then the newly arrived point is not added to $\mathcal{D}_{n-1}$. Thus $\mathcal{D}_n = \mathcal{D}_{n-1}$.

- Otherwise, we compute the prediction error $e_n = y_n - \hat{d}_n$. If $|e_n|$ is smaller than a predefined threshold, δ_2, then the new point is discarded and we set $\mathcal{D}_n = \mathcal{D}_{n-1}$. Only if $|e_n| \geq \delta_2$, then $\boldsymbol{x}_n$ is inserted into $\mathcal{D}_{n-1}$, forming the new dictionary $\mathcal{D}_n = \mathcal{D}_{n-1} \bigcup \{\boldsymbol{x}_n\}$.

Note that whenever we insert a new point into $\mathcal{D}$, we must also insert the respective coefficient, $\theta_n = \mu e_n$, to the growing vector $\boldsymbol{\theta}$.

Another popular scheme is the so called *coherence based* sparsification strategy [33], where the point $\boldsymbol{x}_n$ is inserted into the dictionary, if its coherence is above a given threshold ϵ_0, i.e.

$$\max_{\boldsymbol{u}_i \in \mathcal{D}_n} \{|\kappa(\boldsymbol{x}_n, \boldsymbol{u}_i)|\} \leq \epsilon_0. \tag{2.9}$$

It has been proved that the dimension of the dictionary determined under rule (2.9) remains finite, as n goes to infinity [33].

A more sophisticated strategy is the so called *surprise criterion* [21], which employs ideas from information theoretic learning. The *surprise* of a new data pair $(\boldsymbol{x}_n, y_n)$ with respect to a learning system $\mathcal{T}$ is defined as the negative log likelihood of $(\boldsymbol{x}_n, y_n)$, given the learning system's hypothesis on the data distribution, i.e.,

$$S_{\mathcal{T}}(\boldsymbol{x}_n, y_n) = -\ln p((\boldsymbol{x}_n, y_n)|\mathcal{T}).$$

According to this measure, one may classify the new data pair into the following three categories:

- *Abnormal*: $S_{\mathcal{T}}(\boldsymbol{x}_n, y_n) > \delta_1$,

- *Learnable*: $\delta_1 \geq S_{\mathcal{T}}(\boldsymbol{x}_n, y_n) \geq \delta_2$,

- *Reduntant*: $S_{\mathcal{T}}(\boldsymbol{x}_n, y_n) < \delta_2$,

where δ_1, δ_2 are problem related parameters. In surprise based sparsification, if the new data pair is classified as learnable, it is inserted into the dictionary. For the case of the KLMS with Gaussian inputs, the surprise measure is

$$S_n = \frac{1}{2}\ln(r_n) + \frac{e_n^2}{2r_n}, \tag{2.10}$$

where

$$r_n = \lambda + \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n) - \max_{\boldsymbol{u}_i \in \mathcal{D}_n} \left\{ \frac{\kappa^2(\boldsymbol{x}_n, \boldsymbol{u}_i)}{\kappa(\boldsymbol{u}_i, \boldsymbol{u}_i)} \right\}$$

and λ is a user defined regularization parameter.

A main drawback of the aforementioned sparsification techniques is that if one data point is inserted in the dictionary, it remains in it indefinitely. This may effect the tracking performance of the algorithm in time varying environments. Another technique, that one can adopt to impose sparsity on the solution of the KLMS, that, in addition, changes the coefficients of the solution's expansion, is the *quantization* of the training data in the input space, as in the Quantized KLMS [12]. While each data point $\boldsymbol{x}_n$ arrives sequentially, the algorithm checks if it is a new point or a redundant point. If its distance from the current dictionary $\mathcal{D}_n$ is greater than or equal to the quantization size δ (i.e., $\boldsymbol{x}_n$ cannot be quantized to a point already contained in $\mathcal{D}_n$) then $\boldsymbol{x}_n$ is classified as a new point and it is inserted into the dictionary $\mathcal{D}_n = \mathcal{D}_{n-1} \cup \{\boldsymbol{u}_n\}$. Otherwise, $\boldsymbol{x}_n$ is classified as a "redundant" point and the algorithm uses this information to update the coefficient of the closest center, say $\boldsymbol{u}_{l_0} \in \mathcal{D}_n$. Algorithm 3 summarizes this procedure.

Algorithm 3 Quantized Kernel LMS (QKLMS)

INPUT: $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n), \ldots$
Initialization: Set $\boldsymbol{\theta} = 0$, $\mathcal{D} = \{\}$. Select the step parameter μ the parameters of the kernel and the quantization size δ.

```
M = 0
for n = 1 : N do
    if n == 1 then
        ŷ_n = 0
    else
        Compute the learning system's output: ŷ_n = Σ_{i=1}^{M} θ_i · κ(u_i, x_n).
    end if
    Compute the error: e_n = y_n − ŷ_n.
    θ_n = μ e_n.
    Compute the distance between x_n and D: dist = min_{u_i ∈ D} ||x_n − u_i|| = ||x_n − u_{l_0}||, for an l_0 ∈ {1, ... M}.
    if dist > δ then
        M = M + 1
        Add the new center u_M = x_n to the list of centers, i.e., D = D ∪ {u_n}, θ = (θ^T, θ_n)^T.
    else
        Keep the codebook unchanged and update the coefficient θ_{l_0}, i.e., θ_{l_0} = θ_{l_0} + μ e_n.
    end if
end for
```

OUTPUT: The M-dimensional vector $\boldsymbol{\theta}$ and the dictionary $\mathcal{D} = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_M\}$. The solution is then given as $f = \sum_{i=1}^{M} \theta_i \kappa(\cdot, \boldsymbol{u}_i)$.

An alternative method to impose sparsity in an adaptive setting has been considered in [41, 40, 9]. This is based on regularization and it has the additional benefit of introducing an exponential forgetting mechanism of past data. This mechanism is very natural to the projection-based philosophy of these algorithms, and sparsification only requires an extra projection on a convex ball.

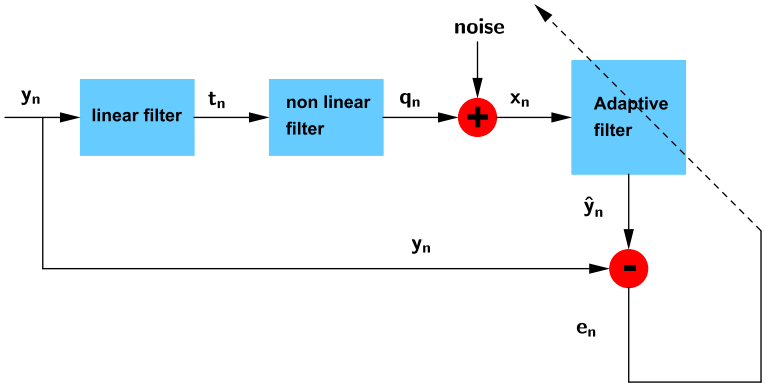


Figure 2.1: The equalization task.

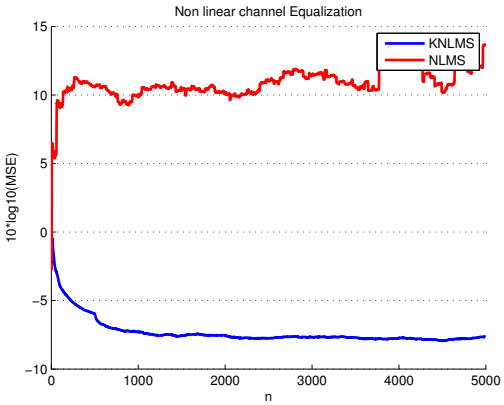


Figure 2.2: Learning curves of normalized LMS ($\mu = 1$) and the normalized Gaussian Kernel LMS ($\mu = 1/2$, $\sigma = 5$) for the equalization problem (filter length $K = 5$, delay $D = 2$).

2.2.2 Simulation Results

To demonstrate the performance of the KLMS, we consider a typical nonlinear channel equalization task (see figure 2.1) similar to the one presented in [38, 15]. The non-linear channel consists of a linear filter:

$$t_n = -0.8 \cdot y_n + 0.7 \cdot y_{n-1}$$

and a memoryless nonlinearity

$$q_n = t_n + 0.25 \cdot t_n^2 + 0.11 \cdot t_n^3. \tag{2.11}$$

The signal is then corrupted by additive white Gaussian noise and then it is observed as x_n. The level of the noise was set equal to 15dB. The aim of a channel equalization task is to design an inverse filter, which acts on the output of the channel, x_n, and reproduces the original input signal, y_n, as close as possible. To this end, we apply the normalized KLMS algorithm to the set of samples

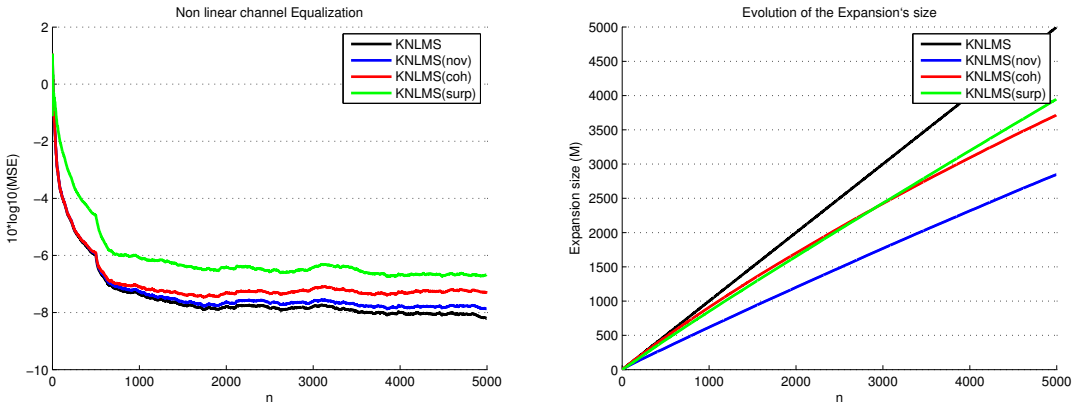
$$((x_n, x_{n-1}, \ldots, x_{n-K+1}), y_{n-D}), \tag{2.12}$$

where $K > 0$ is the equalizer's length and D the equalization time delay, which is present to, almost, any equalization set up. In other words, the output of the equalizer at time n, provides the estimate of y_{n-D}. This is to account for the non minimum phase property of the linear filter that was used to model the channel [16].

Experiments were conducted on 50 sets of 5000 samples of the input signal (Gaussian random variable with zero mean and unit variance) comparing the standard LMS and the KLMS. Regarding the KLMS, we employed the Gaussian kernel (with $\sigma = 5$) and we adopted the novelty criterion sparsification strategy ($\delta_1 = 0.15$, $\delta_2 = 0.2$). The step update parameter, μ, was tuned for the best possible results (in terms of the steady-state error rate). Time delay D was also set for optimality. Figure 2.2 shows the learning curves of KLMS versus the standard LMS algorithm. The superiority of the KLMS is evident and it was expected, since the standard LMS cannot cope with the nonlinearity.

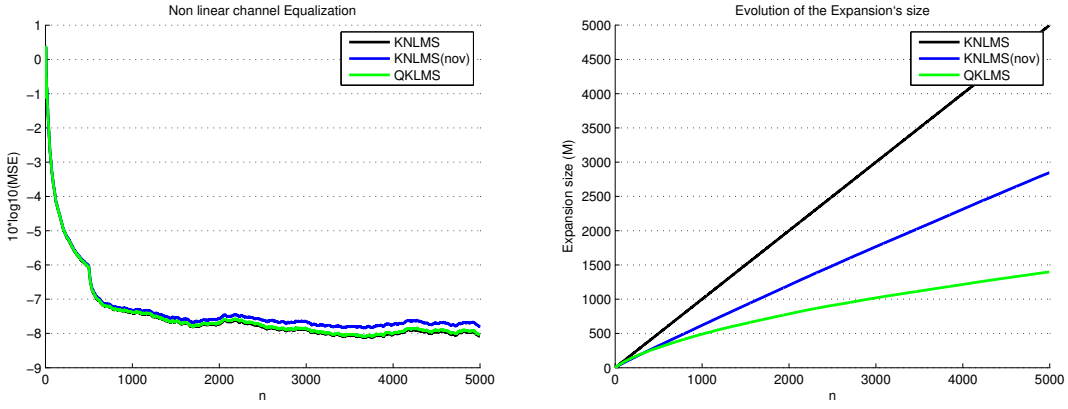
Figures 2.3(a) and 2.3(b) compare the performance of the various sparsification strategies presented in section 2.2.1, considering the same experimental set up. The user defined parameters for the novelty detection approach were set to $\delta_1 = 0.15$, $\delta_2 = 0.2$, for the coherence-based scenario were set to $\epsilon_0 = 0.99$ and for the surprise-based sparsification were set to $\lambda = 0.01$, $\delta_1 = 2$, $\delta_2 = -2$. In this experiment, one observes that although both the surprise and the coherence based scenarios perform almost equally well in terms of sparsity, the latter gives a lower steady state MSE, whereas the novelty detection approach outperforms the other two both in terms of sparsity and the steady state MSE. Moreover, we should note that the coherence based and the surprise strategies are more computationally demanding compared to the novelty criterion, as they involve several evaluations of the kernel function. Figure 2.4 compares the learning curves and the expansion' size evolution of the QKLMS and the novelty detection KLMS at the same experimental set up. There it is shown that QKLMS, while offering a significant sparser solution, at the same time accomplishes a lower steady state MSE. In fact, the convergence behavior of QKLMS is almost identical to the KLMS without sparsification! Moreover, in terms of computational resources QKLMS is comparable to the novelty detection sparsification scenario.

Finally, figures 2.5 and 2.6 compare the performances of the aforementioned sparsification strategies in a set up where the channel undergoes a significant change. In this particular experiment, the input signal was fed to a channel consisting of the linear part $t_n = -0.8y_n + 0.7y_{n-1} - 0.6y_{n-2} + 0.1y_{n-3}$ and the nonlinear part $q_n = t_n + 0.08t_n^2$. The signal was then corrupted by additive white Gaussian noise and then observed as x_n. After the 2000-th time instant the channel changes so that $t_n = 0.6y_n - 0.9y_{n-1} + 0.5y_{n-2} - 0.4y_{n-3}$ and $q_n = t_n + 0.1t_n^2$. Experiments were conducted on 50 sets of 4000 samples of the input signal (Gaussian random variable with zero mean and unit variance). The QKLMS in this test gives excellent results, as it achieves the same steady state MSE as the unsparsified KLMS using only the 1/8th of the training centers. The three sparsification strategies (novelty, coherence, surprise) carry the information learned while the channel was in its original state throughout the learning phase. However, the two first (novelty and coherence) seem to be able to cope with the change of the channel, while the latter fails. The code for the experiments presented in this section can be found in `http://bouboulis.mysch.gr/kernels.html`.



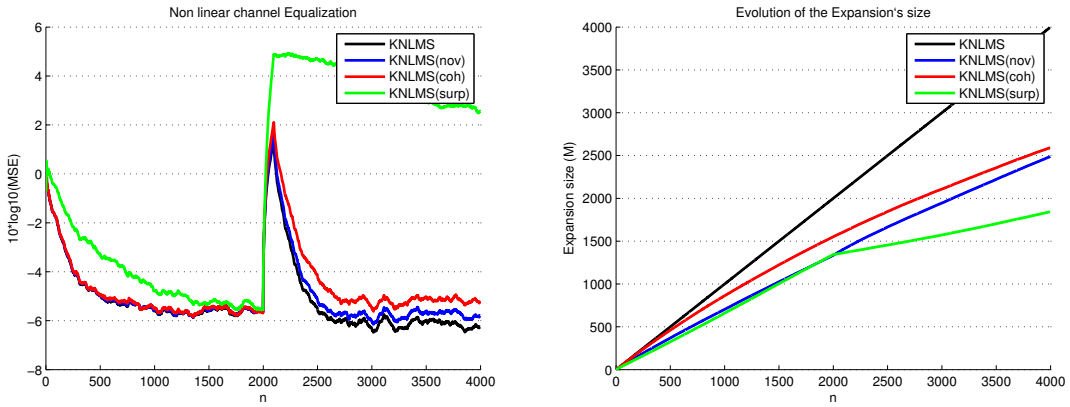
(a) (b)

Figure 2.3: (a) Learning curves of the normalized Kernel LMS (KNLMS) with $\mu = 1/2$, using the Gaussian kernel ($\sigma = 5$), under various sparsification strategies for the equalization problem (filter length $K = 5$, time delay $D = 2$). (b) Evolution of the expansion's size at each time instant, for each sparsification strategy. The user defined parameters for the novelty detection approach were set to $\delta_1 = 0.15$, $\delta_2 = 0.2$, for the coherence-based scenario were set to $\epsilon_0 = 0.99$ and for the surprise-based sparsification were set to $\lambda = 0.01$, $\delta_1 = 2$, $\delta_2 = -2$.



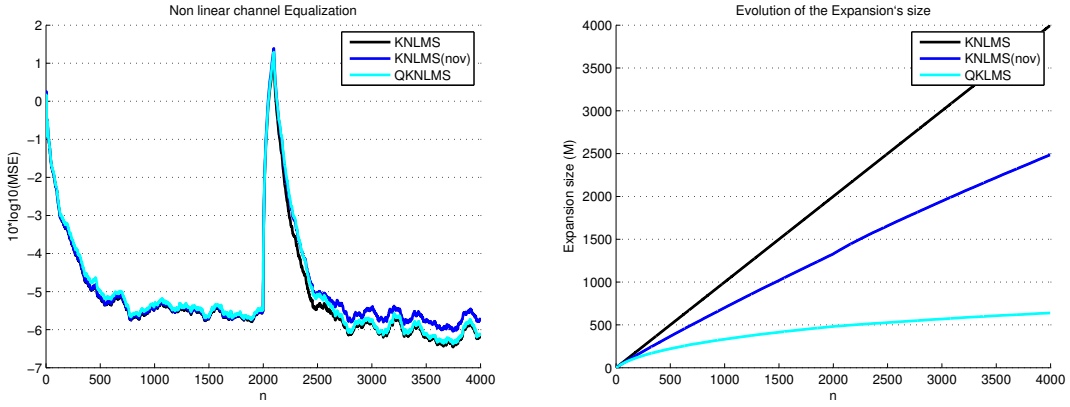
(a) (b)

Figure 2.4: (a) Learning curves of the normalized Kernel LMS (KNLMS) with $\mu = 1/2$, using the Gaussian kernel ($\sigma = 5$), the novelty-detection normalized KNLMS and the QKLMS for the equalization problem (filter length $K = 5$, time delay $D = 2$). (b) Evolution of the expansion's size at each time instant, for the KNLMS, novelty KNLMS, QKNLMS. The user defined parameters for the novelty detection approach were set to $\delta_1 = 0.15$, $\delta_2 = 0.2$. The quantization size (for the QKNLMS) was set $\delta = 1$.



(a) (b)

Figure 2.5: (a) Learning curves of the normalized Kernel LMS (KNLMS) with $\mu = 1/2$, using the Gaussian kernel ($\sigma = 5$), under various sparsification strategies for the two states equalization problem (filter length $K = 5$, time delay $D = 2$). (b) Evolution of the expansion's size at each time instant, for each sparsification strategy. The user defined parameters for the novelty detection approach were set to $\delta_1 = 0.15$, $\delta_2 = 0.2$, for the coherence-based scenario were set to $\epsilon_0 = 0.99$ and for the surprise-based sparsification were set to $\lambda = 0.01$, $\delta_1 = 2$, $\delta_2 = -2$.



(a) (b)

Figure 2.6: (a) Learning curves of the normalized Kernel LMS (KNLMS) with $\mu = 1/2$, using the Gaussian kernel ($\sigma = 5$), the novelty-detection normalized KNLMS and the QKLMS for the two-states equalization problem (filter length $K = 5$, time delay $D = 2$). (b) Evolution of the expansion's size at each time instant, for the KNLMS, novelty KNLMS, QKNLMS. The user defined parameters for the novelty detection approach were set to $\delta_1 = 0.15$, $\delta_2 = 0.2$. The quantization size (for the QKNLMS) was set $\delta = 1$.

2.3 Recursive Least Squares

Recall that we assumed the sequence of examples $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n), \ldots\}$, where $\boldsymbol{x}_n \in X \subset \mathbb{R}^\nu$, and $y_n \in \mathbb{R}$. In the *Recursive Least Squares* (RLS) algorithm, we adopt the parametric class of linear functions, i.e., $\mathcal{F} = \{f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^T\boldsymbol{x}, \boldsymbol{\theta} \in \mathbb{R}^\nu\}$, as the field in which we search for the optimal input-output relationship between $\boldsymbol{x}_n$ and y_n, while the loss function at iteration n takes the form:

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} |y_i - f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)|^2 + \lambda\|\boldsymbol{\theta}\|^2, \tag{2.13}$$

for some chosen $\lambda > 0$. Observe that the loss function consists of two parts. The first one measures the error between the estimated output, $f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$, and the actual output, y_i, at each time instant, i, up to n, while the second is a regularization term. The existence of the second term is crucial to the algorithm, as it prevents it from being ill-posed and guards against overfitting. At each time instant, the RLS finds the solution to the minimization problem:

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \mathcal{L}(\boldsymbol{\theta}) \tag{2.14}$$

The RLS algorithm, usually, exhibits an order of magnitude faster convergence compared to the LMS, at the cost of higher computational complexity.

The rationale of the RLS algorithm can be described in a significant more compact form, if one adopts a matrix representation. To this end, we define the $\nu \times n$ matrices $\boldsymbol{X}_n = (\boldsymbol{x}_1\ \boldsymbol{x}_2\ \ldots\ \boldsymbol{x}_n)$ and the vectors $\boldsymbol{y}_n = (y_1, y_2, \ldots, y_n)^T \in \mathbb{R}^n$, for $n = 1, 2, \ldots$. Then, the loss function (2.13) can be equivalently defined as:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}) &= \|\boldsymbol{y}_n - \boldsymbol{X}_n^T\boldsymbol{\theta}\|^2 + \lambda\|\boldsymbol{\theta}\|^2 = (\boldsymbol{y}_n - \boldsymbol{X}_n^T\boldsymbol{\theta})^T(\boldsymbol{y}_n - \boldsymbol{X}_n^T\boldsymbol{\theta}) + \lambda\|\boldsymbol{\theta}\|^2 \\ &= \|\boldsymbol{y}_n\|^2 - 2\boldsymbol{\theta}^T\boldsymbol{X}_n\boldsymbol{y}_n + \boldsymbol{\theta}^T\boldsymbol{X}_n\boldsymbol{X}_n^T\boldsymbol{\theta} + \lambda\boldsymbol{\theta}^T\boldsymbol{\theta}.\end{aligned}$$

As this is a strictly convex function, it has a unique minimum, which it is obtained at the point, $\boldsymbol{\theta}_n = \text{argmin}\{\mathcal{L}(\boldsymbol{\theta})\}$, where its gradient vanishes, i.e., $\nabla\mathcal{L}(\boldsymbol{\theta}_n) = 0$. Note, that the gradient of the loss function can be derived using standard differentiation rules:

$$\nabla\mathcal{L}(\boldsymbol{\theta}) = -2\boldsymbol{X}_n\boldsymbol{y}_n + 2\boldsymbol{X}_n\boldsymbol{X}_n^T\boldsymbol{\theta} + 2\boldsymbol{\theta}. \tag{2.15}$$

Equating the gradient to zero, we obtain the solution to the minimization problem (2.14):

$$\boldsymbol{\theta}_n = (\lambda\boldsymbol{I}_\nu + \boldsymbol{X}_n\boldsymbol{X}_n^T)^{-1}\boldsymbol{X}_n\boldsymbol{y}_n, \tag{2.16}$$

where $\boldsymbol{I}_\nu$ is the $\nu \times \nu$ identity matrix. Observe, that this solution includes the inversion of a matrix at each time step, which is usually, a computationally demanding procedure. To overcome this obstacle, the RLS algorithm computes the solution recursively, exploiting the matrix inversion lemma:

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}. \tag{2.17}$$

Let $\boldsymbol{P}_n = \lambda I_\nu + \boldsymbol{X}_n\boldsymbol{X}_n^T$. Then, we can easily deduce that

$$\begin{aligned}\boldsymbol{P}_n &= \lambda I_\nu + \boldsymbol{X}_n\boldsymbol{X}_n^T = \lambda I_\nu + \sum_{i=1}^{n}\boldsymbol{x}_i\boldsymbol{x}_i^T = \lambda I_\nu + \sum_{i=1}^{n-1}\boldsymbol{x}_i\boldsymbol{x}_i^T + \boldsymbol{x}_n\boldsymbol{x}_n^T \\ &= \boldsymbol{P}_{n-1} + \boldsymbol{x}_n\boldsymbol{x}_n^T.\end{aligned}$$

Thus, using the matrix inversion lemma (2.17) we take:

$$\boldsymbol{P}_n^{-1} = \left(\boldsymbol{P}_{n-1} + \boldsymbol{x}_n\boldsymbol{x}_n^T\right)^{-1} = \boldsymbol{P}_{n-1}^{-1} - \frac{\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n\boldsymbol{x}_n^T\boldsymbol{P}_{n-1}^{-1}}{1 + \boldsymbol{x}_n^T\boldsymbol{P}_n^{-1}\boldsymbol{x}_n}.$$

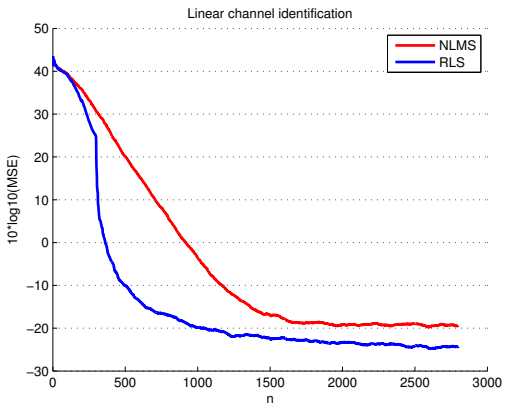


Figure 2.7: Learning curves of the normalized LMS ($\mu = 1$) and the RLS ($\lambda = 0.01$) for a filter identification problem (filter length $L = 200$).

Also, considering that $\boldsymbol{X}_n\boldsymbol{y}_n = \boldsymbol{X}_{n-1}\boldsymbol{y}_{n-1} + \boldsymbol{x}_n y_n$, the solution $\boldsymbol{\theta}_n$ becomes

$$\begin{aligned}
\boldsymbol{\theta}_n = \boldsymbol{P}_n\boldsymbol{X}_n\boldsymbol{y}_n &= \left(\boldsymbol{P}_{n-1}^{-1} - \frac{\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n\boldsymbol{x}_n^T\boldsymbol{P}_{n-1}^{-1}}{1+\boldsymbol{x}_n^T\boldsymbol{P}_n^{-1}\boldsymbol{x}_n}\right)(\boldsymbol{X}_{n-1}\boldsymbol{y}_{n-1} + \boldsymbol{x}_n d_n) \\
&= \boldsymbol{P}_{n-1}^{-1}\boldsymbol{X}_{n-1}\boldsymbol{y}_{n-1} + \boldsymbol{P}_{n-1}\boldsymbol{x}_n y_n - \frac{\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n\boldsymbol{x}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{P}_{n-1}\boldsymbol{X}_{n-1}\boldsymbol{y}_{n-1}}{1+\boldsymbol{x}_n^T\boldsymbol{P}_n^{-1}\boldsymbol{x}_n} - \frac{\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n\boldsymbol{x}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{P}_{n-1}\boldsymbol{x}_n y_n}{1+\boldsymbol{x}_n^T\boldsymbol{P}_n^{-1}\boldsymbol{x}_n} \\
&= \boldsymbol{\theta}_{n-1} - \frac{\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n\boldsymbol{x}_n^T\boldsymbol{\theta}_{n-1}}{1+\boldsymbol{x}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n} + \left(\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n y_n - \frac{\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n\boldsymbol{x}_n^T\boldsymbol{P}_{n-1}^{T}\boldsymbol{x}_n y_n}{1+\boldsymbol{x}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n}\right) \\
&= \boldsymbol{\theta}_{n-1} - \frac{\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n\boldsymbol{x}_n^T\boldsymbol{\theta}_{n-1}}{1+\boldsymbol{x}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n} + \frac{\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n y_n}{1+\boldsymbol{x}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n} \\
&= \boldsymbol{\theta}_{n-1} + \frac{\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n}{1+\boldsymbol{x}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n}\left(y_n - \boldsymbol{x}_n^T\boldsymbol{\theta}_{n-1}\right).
\end{aligned}$$

Thus, the solution $\boldsymbol{\theta}_n$ is computed recursively, as

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + \frac{\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n}{1+\boldsymbol{x}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n} e_n, \tag{2.18}$$

where $e_n = y_n - \boldsymbol{\theta}_{n-1}^T\boldsymbol{x}_n$ is the estimation error at iteration n.

As can be seen in figure 2.7, which considers a 200-taps filter identification problem (where the input signal is a Gaussian random variable with zero mean and unit variance), the RLS learning procedure typically exhibits significantly faster convergence and achieves a lower steady state error compared to the NLMS (in this example the steady state MSE achieved by the RLS is 25% lower than the one achieved by the NLMS). Such a performance is expected for the stationary environment case. The RLS is a Newton-type algorithm, while the LMS is a gradient descent type. Moreover, concerning the excess error for the LMS, this is due to the nondiminishing value of μ, which as we know increases the steady state error.

2.4 Exponentially Weighted RLS

The RLS learning rationale utilizes information of all past data at each iteration step. Although this tactic enables RLS to achieve better convergence speed, it may become a burden in time-varying environments. In such cases, it would be preferable to simply "forget" some of the past data, which correspond to an earlier

Algorithm 4 Recursive Least Squares (RLS)

INPUT: The data $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n), \ldots, \boldsymbol{x}_n \in \mathbb{R}^\nu$, $y_n \in \mathbb{R}$, and the regularization parameter λ.
Initialization: Set $\boldsymbol{P} = \lambda \boldsymbol{I}_\nu$, $\boldsymbol{\theta} = \boldsymbol{0}$.
for $n = 1, 2, \ldots$ **do**
Compute the learning system's output: $\hat{y}_n = \boldsymbol{\theta}^T \boldsymbol{x}_n$.
Compute the error: $e_n = y_n - \hat{y}_n$.
$r = 1 + \boldsymbol{x}_n^T \boldsymbol{P}^{-1} \boldsymbol{x}_n$
$\boldsymbol{k} = \frac{\boldsymbol{P}^{-1}\boldsymbol{x}_n}{r}$,
Update solution: $\boldsymbol{\theta} = \boldsymbol{\theta} + e_n \boldsymbol{k}$.
Update the inverse matrix: $\boldsymbol{P}^{-1} = \boldsymbol{P}^{-1} - \boldsymbol{k}\boldsymbol{k}^T r$.
end for
OUTPUT: The vector $\boldsymbol{\theta} \in \mathbb{R}^\nu$.

state of the learning task. We can incorporate such an approach to the RLS algorithm by introducing some weighting factors to the cost function (2.13), i.e.,

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} w(n,i)|y_i - f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)|^2 + \lambda w(n)\|\boldsymbol{\theta}\|^2, \tag{2.19}$$

where w is an appropriately chosen weighting function. A typical weighting scenario that is commonly used is the *exponential weighting factor* (or the *forgetting factor*), where we choose $w(n,i) = w^{n-i}$ and $w(n) = w^n$, for some $0 < w \leq 1$. Small values of w (close to 0) imply a strong forgetting mechanism, which makes the algorithm more sensitive to recent samples and might lead to poor performance. Hence, we usually select w such that it is close to 1. Obviously, for $w = 1$ we take the ordinary RLS described in 2.3. In the method of *Exponentially Weighted RLS* (EWRLS) we minimize the cost function

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} w^{n-i}|y_i - \boldsymbol{\theta}^T \boldsymbol{x}_i|^2 + \lambda w^n \|\boldsymbol{\theta}\|^2. \tag{2.20}$$

Using the matrix notation the aforementioned cost function becomes $\mathcal{L}(\boldsymbol{\theta}) = \left\| \boldsymbol{W}_n^{\frac{1}{2}} (\boldsymbol{y}_n - \boldsymbol{X}_n \boldsymbol{\theta}) \right\|^2 + w^n \lambda \boldsymbol{\theta}^T \boldsymbol{\theta}$, where $\boldsymbol{X}_n$ and $\boldsymbol{y}_n$ are defined as in section 2.3 and $\boldsymbol{W}_n$ is the diagonal matrix with elements the powers of w up to $n-1$, i.e.,

$$\boldsymbol{W}_n = \begin{pmatrix} w^{n-1} & 0 & 0 & \ldots & 0 \\ 0 & w^{n-2} & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & & 1 \end{pmatrix}.$$

The solution to the respective minimization problem is

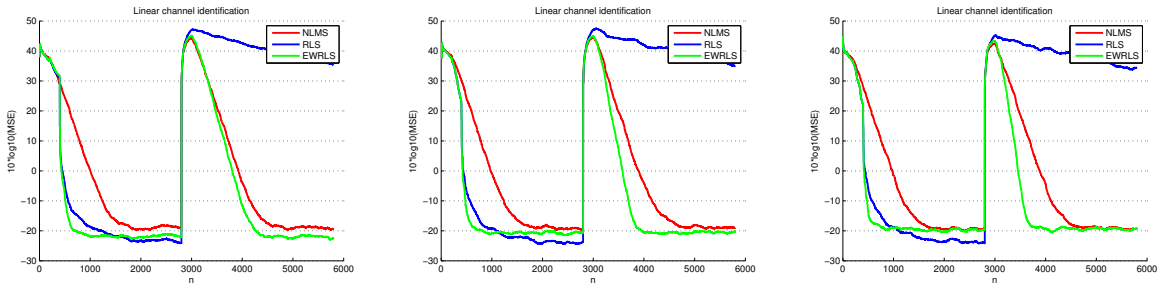
$$\boldsymbol{\theta}_n = (\boldsymbol{X}_n \boldsymbol{W}_n \boldsymbol{X}_n^T + w^n \lambda \boldsymbol{I}_\nu)^{-1} \boldsymbol{X}_n \boldsymbol{W}_n \boldsymbol{y}_n. \tag{2.21}$$

Following a similar rationale as in section 2.3, we have:

$$\begin{aligned} \boldsymbol{P}_n = (\boldsymbol{X}_n \boldsymbol{W}_n \boldsymbol{X}_n^T + w^n \lambda \boldsymbol{I}_\nu) &= \sum_{i=1}^{n} w^{n-i} \boldsymbol{x}_i \boldsymbol{x}_i^T + w^n \lambda \boldsymbol{I}_\nu \\ &= \boldsymbol{x}_n \boldsymbol{x}_n^T + \sum_{i=1}^{n-1} w^{n-i} \boldsymbol{x}_i \boldsymbol{x}_i^T + w^n \lambda \boldsymbol{I}_\nu \\ &= w \boldsymbol{P}_{n-1} + \boldsymbol{x}_n \boldsymbol{x}_n^T. \end{aligned}$$

Applying the matrix inversion lemma, we can produce a recursive relation for $\boldsymbol{P}_n^{-1}$:

$$\boldsymbol{P}_n^{-1} = w^{-1} \boldsymbol{P}_{n-1}^{-1} - \frac{w^{-2} \boldsymbol{P}_{n-1}^{-1} \boldsymbol{x}_n \boldsymbol{x}_n^T \boldsymbol{P}_{n-1}^{-1}}{1 + w^{-1} \boldsymbol{x}_n^T \boldsymbol{P}_{n-1}^{-1} \boldsymbol{x}_n},$$



(a) $w = 0.996$ (b) $w = 0.994$ (c) $w = 0.992$

Figure 2.8: Learning curves of the normalized LMS ($\mu = 1$), the RLS and EWRLS ($\lambda = 0.001$) for various values of the weighting factor w, for a time-varying filter identification problem (filter length $K = 200$).

and a respective recursive relation for the final solution:

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + \frac{w^{-1}\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n}{1 + w^{-1}\boldsymbol{x}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{x}_n} e_n.$$

Algorithm 5 describes the EWRLS machinery in details. We test the performance of the EWRLS in a time-varying channel identification problem. The signal (zero-mean unite-variance Gaussian random variable) is fed to a linear channel (length $K = 200$) and then corrupted by white Gaussian noise. After 2000 samples of data, the coefficients of the linear channel undergo a significant change. Figure 2.8 shows the performance of the EWRLS algorithm compared to the standard RLS and the NLMS for various values of the weighting coefficient w. While the standard RLS fails to adapt to the change of the environment, the EWRLS learns the new channel effectively, at a cost of a slightly larger steady state MSE due to the weighting of the error at each time step. Note that the tracking performance of both the LMS and the RLS algorithms depends on the input signal [16, 35].

Algorithm 5 Exponential Weighted Recursive Least Squares (RLS)

INPUT: The data $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n), \ldots, \boldsymbol{x}_n \in \mathbb{R}^\nu$, $y_n \in \mathbb{R}$, the weighting factor w and the regularization parameter λ.
Initialization: Set $\boldsymbol{P} = \lambda \boldsymbol{I}_\nu$, $\boldsymbol{\theta} = \boldsymbol{0}$.
for $n = 1, 2, \ldots$ **do**
Compute the learning system's output: $\hat{y}_n = \boldsymbol{\theta}^T \boldsymbol{x}_n$.
Compute the error: $e_n = y_n - \hat{y}_n$.
$r = 1 + w^{-1}\boldsymbol{x}_n^T \boldsymbol{P}^{-1}\boldsymbol{x}_n$
$\boldsymbol{k} = (rw)^{-1}\boldsymbol{P}^{-1}\boldsymbol{x}_n$,
Update solution: $\boldsymbol{\theta} = \boldsymbol{\theta} + e_n \boldsymbol{k}$.
Update the inverse matrix: $\boldsymbol{P}^{-1} = w^{-1}\boldsymbol{P}^{-1} - \boldsymbol{k}\boldsymbol{k}^T r$.
end for
OUTPUT: The vector $\boldsymbol{\theta} \in \mathbb{R}^\nu$.

2.5 The Kernel RLS

Similar to the case of the KLMS, in the *Kernel RLS* (KRLS) we estimate the learning system's output by a function f defined on the RKHS, $\mathcal{H}$, which is associated to a positive definite kernel κ. The sequence of examples are transformed via the feature map $\phi : X \to \mathcal{H}$, $\phi(\boldsymbol{x}) = \kappa(\cdot, \boldsymbol{x})$ to generate the input data

$$\{(\phi(\boldsymbol{x}_1), y_1), (\phi(\boldsymbol{x}_2), y_2), \ldots, (\phi(\boldsymbol{x}_n), y_n), \ldots\}$$

and the employed loss function takes the form

$$\mathcal{L}(f) = \sum_{i=1}^{n} |y_i - \hat{y}_i|^2 + \lambda \|f\|_{\mathcal{H}}^2, \tag{2.22}$$

where the estimated output of the learning system at iteration n is considered as $\hat{y}_n = \langle \phi(\boldsymbol{x}_n), f \rangle_{\mathcal{H}}$, for some $f \in \mathcal{H}$. The representer theorem 1.5.3 ensures that the solution to the minimization task, $f_n = \text{argmin}_{f \in \mathcal{H}} \mathcal{L}(f)$, lies in the finite dimensional subspace of the span of the n particular kernels that are centered on the training points $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$. In other words:

$$f_n = \sum_{i=1}^{n} \theta_{n,i} \kappa(\cdot, \boldsymbol{x}_i) \tag{2.23}$$

and thus, exploiting the kernel trick, the estimated output at iteration n will be given by

$$\hat{y}_n = \langle \phi(\boldsymbol{x}_n), f_n \rangle_{\mathcal{H}} = \sum_{i=1}^{n} \theta_{n,i} \kappa(\boldsymbol{x}_i, \boldsymbol{x}_n). \tag{2.24}$$

As this expansion grows unbounded while time evolves, analogous to the case of KLMS, a sparsification mechanism, which keeps the most informative training points and discards the rest, is also needed. KRLS's main task is to estimate the (growing) vector $\boldsymbol{\theta}_n = (\theta_{n,1}, \ldots, \theta_{n,n})^T$ in a recursive manner. The first approach to the KRLS task was presented in [15] by Engel, Mannor and Meir as a part of Engel's Ph.D. thesis. Their method is built around a specific sparsification strategy, which is called *Approximate Linear Dependency* (ALD). To this end, consider a dictionary of centers $\mathcal{D}_{n-1} = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{M_{n-1}}\}$, of size M_{n-1}, that keeps some of the past training points. In this setting, we test whether the newly arrived data $\phi(\boldsymbol{x}_n)$ is approximately linearly dependent on the dictionary elements $\phi(\boldsymbol{u}_1), \ldots \phi(\boldsymbol{u}_{M_{n-1}})$. If not, we add it to the dictionary, otherwise $\phi(\boldsymbol{x}_n)$ is approximated by a linear combination of the dictionary's elements. The specific condition is whether

$$\delta_n = \min_{\boldsymbol{a}_n} \left\{ \left\| \sum_{m=1}^{M_{n-1}} a_{n,m} \phi(\boldsymbol{u}_m) - \phi(\boldsymbol{x}_n) \right\|^2 \right\} \leq \epsilon_0, \tag{2.25}$$

where ϵ_0 is a user defined parameter. Using matrix-notation this minimization problem can be equivalently recasted as:

$$\delta_n = \min_{\boldsymbol{a}_n} \left(\boldsymbol{a}_n^T \tilde{\boldsymbol{K}}_{n-1} \boldsymbol{a}_n - 2 \boldsymbol{a}_n^T \boldsymbol{\beta}_n + \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n) \right),$$

where $(\tilde{\boldsymbol{K}}_{n-1})_{i,j} = \kappa(\boldsymbol{u}_i, \boldsymbol{u}_j)$, for $i, j = 1, 2, \ldots, M_{n-1}$, and $\boldsymbol{\beta}_n = (\kappa(\boldsymbol{u}_1, \boldsymbol{x}_n), \kappa(\boldsymbol{u}_2, \boldsymbol{x}_n), \ldots, \kappa(\boldsymbol{u}_{M_{n-1}}, \boldsymbol{x}_n))^T$. Assuming that $\tilde{\boldsymbol{K}}_{n-1}$ is invertible, the solution to this minimization problem is

$$\boldsymbol{a}_n = \tilde{\boldsymbol{K}}_{n-1}^{-1} \boldsymbol{\beta}_n, \tag{2.26}$$

while the attained minimum value is $\delta_n = \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n) - \boldsymbol{\beta}_n^T \boldsymbol{a}_n$. In the case where $\delta_n > \epsilon_0$, $\boldsymbol{x}_n$ is added to the dictionary, which now contains $M_n = M_{n-1} + 1$ centers. If $\delta_n \leq \epsilon_0$, then $\boldsymbol{x}_n$ is not included to the dictionary, i.e., $\mathcal{D}_n = \mathcal{D}_{n-1}$, $M_n = M_{n-1}$, and we approximate $\phi(\boldsymbol{x}_n)$ as

$$\phi(\boldsymbol{x}_n) \approx \sum_{m=1}^{M_n} a_{n,m} \phi(\boldsymbol{u}_m). \tag{2.27}$$

Using (2.27), we can approximate the full kernel matrix $\boldsymbol{K}_n$ (i.e., $(\boldsymbol{K}_n)_{i,j} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, for $i, j = 1, \ldots, n$) as

$$\boldsymbol{K}_n \approx \boldsymbol{A}_n \cdot \tilde{\boldsymbol{K}}_n \cdot \boldsymbol{A}_n^T, \tag{2.28}$$

where $\boldsymbol{A}_n = (\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_n)^T$ is the $M_n \times M_n$ matrix that contains the coefficients of the linear combinations associated with each $\boldsymbol{u}_m$, $m = 1, \ldots, M_n$. This is due to the fact that at time index n each kernel evaluation is computed as

$$\begin{aligned} \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle_{\mathcal{H}} &= \left\langle \sum_{k=1}^{M_n} a_{i,k} \phi(\boldsymbol{u}_k), \sum_{l=1}^{M_n} a_{j,l} \phi(\boldsymbol{u}_l) \right\rangle_{\mathcal{H}} \\ &= \sum_{k=1}^{M_n} \sum_{l=1}^{M_n} a_{j,l} a_{i,k} \langle \phi(\boldsymbol{u}_k), \phi(\boldsymbol{u}_l) \rangle_{\mathcal{H}} = \sum_{k=1}^{M_n} \sum_{l=1}^{M_n} a_{j,l} a_{i,k} \kappa(\boldsymbol{u}_k, \boldsymbol{u}_l), \end{aligned}$$

for all $k, l = 1, \ldots, M_n$. Note that $a_{i,j} = 0$ for all $j > i$. This is the kick-off point of KRLS rationale.

In matrix notation the respective loss of the KRLS minimization task at iteration n is given by

$$\mathcal{L}(\boldsymbol{\theta}) = \|\boldsymbol{y}_n - \boldsymbol{K}_n\boldsymbol{\theta}\|^2.$$

As $\boldsymbol{K}_n \approx \boldsymbol{A}_n \cdot \tilde{\boldsymbol{K}}_n \cdot \boldsymbol{A}_n^T$, $\mathcal{L}(\boldsymbol{\theta})$ can be recasted as

$$\mathcal{L}(\boldsymbol{\theta}) \approx \|\boldsymbol{y}_n - \boldsymbol{A}_n\tilde{\boldsymbol{K}}_n\boldsymbol{A}_n^T\boldsymbol{\theta}\|^2 = \|\boldsymbol{y}_n - \boldsymbol{A}_n\tilde{\boldsymbol{K}}_n\tilde{\boldsymbol{\theta}}\|^2 = \mathcal{L}(\tilde{\boldsymbol{\theta}}),$$

where $\tilde{\boldsymbol{\theta}} = \boldsymbol{A}_n^T\boldsymbol{\theta}$ is the new parameter to be optimized. Observe that while the original vector $\boldsymbol{\theta}_n \in \mathbb{R}^n$ is a n-dimensional vector, the new vector, $\tilde{\boldsymbol{\theta}}_n \in \mathbb{R}^{M_n}$, is significantly smaller (depending on the sparsification strategy). The solution to the modified minimization task is

$$\begin{aligned}\tilde{\boldsymbol{\theta}}_n &= (\tilde{\boldsymbol{K}}_n^T\boldsymbol{A}_n^T\boldsymbol{A}_n\tilde{\boldsymbol{K}}_n)^{-1}\tilde{\boldsymbol{K}}_n^T\boldsymbol{A}_n^T\boldsymbol{y}_n \\ &= \tilde{\boldsymbol{K}}_n^{-1}(\boldsymbol{A}_n^T\boldsymbol{A}_n)^{-1}\tilde{\boldsymbol{K}}_n^{-T}\tilde{\boldsymbol{K}}_n^T\boldsymbol{A}_n^T\boldsymbol{y}_n \\ &= \boldsymbol{K}_n^{-1}\boldsymbol{P}_n^{-1}\boldsymbol{A}_n^T\boldsymbol{y}_n, \end{aligned} \tag{2.29}$$

where $\boldsymbol{P}_n = \boldsymbol{A}_n^T\boldsymbol{A}_n$, assuming that the inverse matrices exist. We distinguish the two different cases of the ALD sparsification and derive recursive relations for $\tilde{\boldsymbol{\theta}}_n$ accordingly.

CASE I. If $\delta_n < \epsilon_0$ then the data point arrived at iteration n, i.e., $(\boldsymbol{x}_n, y_n)$, is not added to the dictionary. Hence $\mathcal{D}_n = \mathcal{D}_{n-1}$, $M_n = M_{n-1}$, $\tilde{\boldsymbol{K}}_n = \tilde{\boldsymbol{K}}_{n-1}$ and the expansion of $\phi(\boldsymbol{x}_n)$ as a linear combination of the dictionary's elements is added to $\boldsymbol{A}_n$, i.e.,

$$\boldsymbol{A}_n = \begin{pmatrix}\boldsymbol{A}_{n-1} \\ \boldsymbol{a}_n^T\end{pmatrix}, \tag{2.30}$$

where $\boldsymbol{a}_n$ is given by (2.26). Therefore $\boldsymbol{P}_n = \boldsymbol{A}_n^T\boldsymbol{A}_n$ can be computed as

$$\begin{aligned}\boldsymbol{P}_n = \boldsymbol{A}_n^T\boldsymbol{A}_n &= (\boldsymbol{A}_{n-1}^T \quad \boldsymbol{a}_n)\begin{pmatrix}\boldsymbol{A}_{n-1} \\ \boldsymbol{a}_n^T\end{pmatrix} \\ &= \boldsymbol{A}_{n-1}^T\boldsymbol{A}_{n-1} + \boldsymbol{a}_n\boldsymbol{a}_n^T = \boldsymbol{P}_{n-1} + \boldsymbol{a}_n\boldsymbol{a}_n^T.\end{aligned}$$

Applying the matrix inversion lemma, we take the recursive relation

$$\boldsymbol{P}_n^{-1} = \boldsymbol{P}_{n-1}^{-1} - \frac{\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n\boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}}{1 + \boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n}. \tag{2.31}$$

Substituting in (2.29) and considering that $\boldsymbol{A}_n^T\boldsymbol{y}_n = \boldsymbol{A}_{n-1}^T\boldsymbol{y}_{n-1} + \boldsymbol{a}_n y_n$ we take

$$\begin{aligned}\tilde{\boldsymbol{\theta}}_n &= \tilde{\boldsymbol{K}}_n^{-1}\boldsymbol{P}_n^{-1}\boldsymbol{A}_n^T\boldsymbol{y}_n \\ &= \tilde{\boldsymbol{K}}_n^{-1}\left(\boldsymbol{P}_{n-1}^{-1} - \frac{\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n\boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}}{1 + \boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n}\right)(\boldsymbol{A}_{n-1}^T\boldsymbol{y}_{n-1} + \boldsymbol{a}_n y_n) \\ &= \tilde{\boldsymbol{\theta}}_{n-1} + \frac{\tilde{\boldsymbol{K}}_n^{-1}\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n y_n - \tilde{\boldsymbol{K}}_n^{-1}\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n\boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{A}_n^T\boldsymbol{y}_n}{1 + \boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n} \\ &= \tilde{\boldsymbol{\theta}}_{n-1} + \frac{\boldsymbol{K}_n^{-1}\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n\left(y_n - \boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{A}_{n-1}^T\boldsymbol{y}_{n-1}\right)}{1 + \boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n}.\end{aligned}$$

However, relation (2.29) implies that $\tilde{\boldsymbol{K}}_{n-1}\tilde{\boldsymbol{\theta}}_{n-1} = \boldsymbol{P}_{n-1}^{-1}\boldsymbol{A}_{n-1}^T\boldsymbol{y}_{n-1}$. Hence,

$$\begin{aligned}\tilde{\boldsymbol{\theta}}_n &= \tilde{\boldsymbol{\theta}}_{n-1} + \frac{\boldsymbol{K}_n^{-1}\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n\left(y_n - \boldsymbol{a}_n^T\tilde{\boldsymbol{K}}_{n-1}\tilde{\boldsymbol{\theta}}_{n-1}\right)}{1+\boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n}\\ &= \tilde{\boldsymbol{\theta}}_{n-1} + \frac{\boldsymbol{K}_n^{-1}\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n\left(y_n - \boldsymbol{\beta}_n^T\tilde{\boldsymbol{\theta}}_{n-1}\right)}{1+\boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n},\\ &= \tilde{\boldsymbol{\theta}}_{n-1} + \frac{\boldsymbol{K}_n^{-1}\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n e_n}{1+\boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{a}_n},\end{aligned}$$

where the last relations are generated by substituting $\boldsymbol{\beta}_n^T = \boldsymbol{a}_n^T\tilde{\boldsymbol{K}}_{n-1}$ (see relation (2.26)) and considering that the estimated output at time index n by the learning system is $\hat{y}_n = \boldsymbol{\beta}_n^T\tilde{\boldsymbol{\theta}}_n = \boldsymbol{a}_n^T\tilde{\boldsymbol{K}}_n\tilde{\boldsymbol{\theta}}_n$.

CASE II. If $\delta_n > \epsilon_0$, then the data point arrived at iteration n, i.e., $(\boldsymbol{x}_n, y_n)$, is added to the dictionary, i.e., $\mathcal{D}_n = \mathcal{D}_{n-1}\bigcup\{\boldsymbol{x}_n\}$, $M_n = M_{n-1}+1$ and the matrices $\boldsymbol{A}_n$ and $\tilde{\boldsymbol{K}}_n$ are constructed as follows:

$$\boldsymbol{A}_n = \begin{pmatrix}\boldsymbol{A}_{n-1} & \boldsymbol{0}\\ \boldsymbol{0}^T & 1\end{pmatrix}, \quad \tilde{\boldsymbol{K}}_n = \begin{pmatrix}\tilde{\boldsymbol{K}}_{n-1} & \boldsymbol{\beta}_n\\ \boldsymbol{\beta}_n^T & \kappa(\boldsymbol{x}_n,\boldsymbol{x}_n)\end{pmatrix}.$$

Moreover,

$$\boldsymbol{P}_n = \boldsymbol{A}_n^T\boldsymbol{A}_n = \begin{pmatrix}\boldsymbol{A}_{n-1}^T & \boldsymbol{0}^T\\ \boldsymbol{0} & 1\end{pmatrix}\begin{pmatrix}\boldsymbol{A}_{n-1} & \boldsymbol{0}\\ \boldsymbol{0}^T & 1\end{pmatrix} = \begin{pmatrix}\boldsymbol{A}_{n-1}^T\boldsymbol{A}_{n-1} & \boldsymbol{0}\\ \boldsymbol{0}^T & 1\end{pmatrix}.$$

Hence, applying the matrix inversion lemma to $\tilde{\boldsymbol{K}}_n$ and substituting $\boldsymbol{a}_n = \tilde{\boldsymbol{K}}_{n-1}^{-1}\boldsymbol{\beta}_n$, we have

$$\tilde{\boldsymbol{K}}_n^{-1} = \begin{pmatrix}\tilde{\boldsymbol{K}}_{n-1}^{-1} + \frac{1}{\delta_n}\tilde{\boldsymbol{K}}_{n-1}^{-1}\boldsymbol{\beta}_n\boldsymbol{\beta}_n^T\tilde{\boldsymbol{K}}_{n-1}^{-1} & -\frac{1}{\delta_n}\tilde{\boldsymbol{K}}_{n-1}^{-1}\boldsymbol{\beta}_n\\ -\frac{1}{\delta_n}\boldsymbol{\beta}_n^T\tilde{\boldsymbol{K}}_n^{-1} & \frac{1}{\delta_n}\end{pmatrix} = \frac{1}{\delta_n}\begin{pmatrix}\delta_n\tilde{\boldsymbol{K}}_{n-1}^{-1} + \boldsymbol{a}_n\boldsymbol{a}_n^T & -\boldsymbol{a}_n\\ -\boldsymbol{a}_n^T & 1\end{pmatrix}.$$

and

$$\boldsymbol{P}_n^{-1} = \begin{pmatrix}(\boldsymbol{A}_n^T\boldsymbol{A}_n)^{-1} & \boldsymbol{0}\\ \boldsymbol{0}^T & 1\end{pmatrix} = \begin{pmatrix}\boldsymbol{P}_{n-1}^{-1} & \boldsymbol{0}\\ \boldsymbol{0}^T & 1\end{pmatrix}.$$

Observe that while we replace $\tilde{\boldsymbol{K}}_{n-1}^{-1}\boldsymbol{\beta}_n$ by $\boldsymbol{a}_n$ using relation (2.26), the actual n-th row of $\boldsymbol{A}_n$ is $(0,0,\ldots,0,1)$. Substituting in (2.29) we take:

$$\begin{aligned}\tilde{\boldsymbol{\theta}}_n &= \frac{1}{\delta_n}\begin{pmatrix}\delta_n\tilde{\boldsymbol{K}}_{n-1}^{-1} + \boldsymbol{a}_n\boldsymbol{a}_n^T & -\boldsymbol{a}_n\\ -\boldsymbol{a}_n^T & 1\end{pmatrix}\begin{pmatrix}\boldsymbol{P}_{n-1}^{-1} & \boldsymbol{0}\\ \boldsymbol{0}^T & 1\end{pmatrix}\begin{pmatrix}\boldsymbol{A}_{n-1}^T & \boldsymbol{0}^T\\ \boldsymbol{0} & 1\end{pmatrix}\begin{pmatrix}\boldsymbol{y}_{n-1}\\ y_n\end{pmatrix}\\ &= \frac{1}{\delta_n}\begin{pmatrix}\delta_n\tilde{\boldsymbol{K}}_{n-1}^{-1}\boldsymbol{P}_{n-1}^{-1}\boldsymbol{A}_{n-1}^T\boldsymbol{y}_{n-1} + \boldsymbol{a}_n\boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{A}_{n-1}^T\boldsymbol{y}_{n-1} - \boldsymbol{a}_n y_n\\ -\boldsymbol{a}_n^T\boldsymbol{P}_{n-1}^{-1}\boldsymbol{A}_{n-1}^T\boldsymbol{y}_{n-1} + y_n\end{pmatrix}\\ &= \frac{1}{\delta_n}\begin{pmatrix}\delta_n\tilde{\boldsymbol{\theta}}_{n-1} + \boldsymbol{a}_n\boldsymbol{a}_n^T\tilde{\boldsymbol{K}}_{n-1}\tilde{\boldsymbol{\theta}}_{n-1} - \boldsymbol{a}_n y_n\\ -\boldsymbol{a}_n^T\tilde{\boldsymbol{K}}_{n-1}\tilde{\boldsymbol{\theta}}_{n-1} + y_n\end{pmatrix}\\ &= \frac{1}{\delta_n}\begin{pmatrix}\delta_n\tilde{\boldsymbol{\theta}}_{n-1} + \boldsymbol{a}_n\boldsymbol{\beta}_n^T\tilde{\boldsymbol{\theta}}_{n-1} - \boldsymbol{a}_n y_n\\ -\boldsymbol{\beta}_n^T\tilde{\boldsymbol{\theta}}_{n-1} + y_n\end{pmatrix}\\ &= \begin{pmatrix}\tilde{\boldsymbol{\theta}}_{n-1} - \frac{\boldsymbol{a}_n}{\delta_n}e_n\\ -\frac{e_n}{\delta_n}\end{pmatrix}.\end{aligned}$$

The procedure is summarized in algorithm 6. The KRLS algorithm has complexity $O(M_n)$, at time n, where M_n is the size of the dictionary. Note, that we developed the specific recursive strategy assuming the invertibility of $\tilde{\boldsymbol{K}}_n$ and $\boldsymbol{A}_n^T\boldsymbol{A}_n$ for each n. For the case of the RKHS induced by the Gaussian kernel, which is the most popular kernel for practical applications, we have established that $\tilde{\boldsymbol{K}}_n$ is indeed invertible (theorem 1.7.2). Furthermore, by construction, the matrix $\boldsymbol{A}_n$ has full-rank. Hence, $\boldsymbol{A}_n^T\boldsymbol{A}_n$ is strictly positive definite and thus invertible too.

Algorithm 6 Kernel Recursive Least Squares (E-KRLS)

INPUT: The data $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n), \ldots, \boldsymbol{x}_n \in \mathbb{R}^\nu$, $y_n \in \mathbb{R}$, the ALD parameter ϵ_0.
Initialization: Set $\tilde{\boldsymbol{K}}^{-1} = [1/\kappa(\boldsymbol{x}_1, \boldsymbol{x}_1)]$, $\boldsymbol{P}^{-1} = [1]$, $\tilde{\boldsymbol{\theta}} = [y_1/\kappa(\boldsymbol{x}_1, \boldsymbol{x}_1)]$, $M = 1$.
for $n = 2 : N$ **do**
  Compute $\boldsymbol{\beta} = (\kappa(\boldsymbol{u}_1, \boldsymbol{x}_n), \ldots, \kappa(\boldsymbol{u}_M, \boldsymbol{x}_n))^T$.
  Compute the learning system's output: $\hat{y}_n = \boldsymbol{\theta}^T \boldsymbol{\beta}$.
  Compute the error: $e_n = y_n - \hat{y}_n$.
  FOR ALD TEST:
  Compute $\boldsymbol{a} = \tilde{\boldsymbol{K}}^{-1}\boldsymbol{\beta}$.
  Compute $\delta = \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n) - \boldsymbol{\beta}^T \boldsymbol{a}$
  if $\delta > \epsilon_0$ **then**
    Include the new sample to the dictionary: $\mathcal{D} = \mathcal{D} \cup \{\boldsymbol{x}_n\}$, $M = M + 1$.
    Update $\tilde{\boldsymbol{K}}^{-1} = \frac{1}{\delta}\begin{pmatrix} \delta\tilde{\boldsymbol{K}}^{-1} + \boldsymbol{a}\boldsymbol{a}^T & -\boldsymbol{a} \\ -\boldsymbol{a} & 1 \end{pmatrix}$
    Update $\boldsymbol{P}^{-1} = \begin{pmatrix} \boldsymbol{P}^{-1} & \boldsymbol{0} \\ \boldsymbol{0}^T & 1 \end{pmatrix}$
    Update solution: $\tilde{\boldsymbol{\theta}} = \begin{pmatrix} \tilde{\boldsymbol{\theta}} - \frac{\boldsymbol{a}}{\delta} e_n \\ -\frac{e_n}{\delta} \end{pmatrix}$.
  else
    $\boldsymbol{q} = \frac{\boldsymbol{P}^{-1}\boldsymbol{a}}{1 + \boldsymbol{a}^T \boldsymbol{P}^{-1}\boldsymbol{a}}$.
    Update $\boldsymbol{P}^{-1} = \boldsymbol{P}^{-1} - \boldsymbol{q}\boldsymbol{a}^T\boldsymbol{P}^{-1}$.
    Update solution: $\tilde{\boldsymbol{\theta}} = \tilde{\boldsymbol{\theta}} + \boldsymbol{K}^{-1}\boldsymbol{q}e_n$.
  end if
end for

OUTPUT: The vector $\tilde{\boldsymbol{\theta}} \in \mathbb{R}^M$ and the dictionary $\mathcal{D} = \{\boldsymbol{u}_1, \ldots \boldsymbol{u}_M\}$.

2.5.1 An alternative KRLS algorithm

A different implementation of the RLS machinery using kernels is considered in [21]. There, the data points are transformed in the RKHS $\mathcal{H}$, via the feature map ϕ as usual, and an orthonormal basis of $\mathcal{H}$ is considered. Using this orthonormal basis, we can define a mapping $\psi : \mathcal{H} \to \ell_2$ such that each function $f \in \mathcal{H}$ is transformed to a sequence $\psi(f) \in \ell_2$ (or equivalently to a vector of, possible, infinite dimension) that is comprised from the coefficients of the expansion of f with respect to the orthonormal basis. Using this notation and elementary algebra, we can prove that

$$\langle f, h \rangle_{\mathcal{H}} = \psi(h)^T \psi(f), \tag{2.32}$$

where $\psi(f)$ and $\psi(h)$ are treated as vectors. A special case of the aforementioned relation is $\psi(\phi(\boldsymbol{y}))^T\psi(\phi(\boldsymbol{x})) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle_{\mathcal{H}} = \kappa(\boldsymbol{x}, \boldsymbol{y})$. In this context, the KRLS loss, i.e., relation (2.22), can be recasted as:

$$\mathcal{L}(\Theta) = \|\boldsymbol{y}_n - \boldsymbol{\Phi}_n^T \psi(\Theta)\|^2 + \lambda \|\psi(\Theta)\|^2, \tag{2.33}$$

where $\boldsymbol{\Phi}_n = (\psi(\phi(\boldsymbol{x}_1)), \ldots, \psi(\phi(\boldsymbol{x}_n)))$. The solution to the respective minimization problem at time index n is given by

$$\psi(\Theta_n) = \left(\boldsymbol{\Phi}_n \boldsymbol{\Phi}_n^T + \lambda \boldsymbol{I}_\infty\right)^{-1} \boldsymbol{\Phi}_n \boldsymbol{y}_n. \tag{2.34}$$

Unfortunately, equation (2.34) is not feasible as it employs infinite dimensional matrices. Nevertheless, Liu and Principe exploited an elegant, alternative way to overcome this problem. They observed that applying the matrix-inversion lemma leads to the following relations:

$$(\lambda \boldsymbol{I}_\infty + \boldsymbol{\Phi}_n \boldsymbol{\Phi}_n^T)^{-1} \boldsymbol{\Phi}_n = \frac{1}{\lambda}\boldsymbol{\Phi}_n - \frac{1}{\lambda^2}\boldsymbol{\Phi}_n \boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n - \frac{1}{\lambda}\boldsymbol{\Phi}_n \boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n \boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n$$
$$\boldsymbol{\Phi}_n (\lambda \boldsymbol{I}_n + \boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n)^{-1} = \frac{1}{\lambda}\boldsymbol{\Phi}_n - \frac{1}{\lambda^2}\boldsymbol{\Phi}_n \boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n - \frac{1}{\lambda}\boldsymbol{\Phi}_n \boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n \boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n$$

and hence they concluded that

$$(\lambda \boldsymbol{I}_\infty + \boldsymbol{\Phi}_n \boldsymbol{\Phi}_n^T)^{-1} \boldsymbol{\Phi}_n = \boldsymbol{\Phi}_n (\lambda \boldsymbol{I}_\infty + \boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n)^{-1}. \tag{2.35}$$

However, observe that the first relation is produced by an application of the matrix inversion lemma to infinite dimensional matrices (at the time that this manuscript was compiled, we weren't aware of whether there is a formal proof regarding the Woodbery's identity in infinite matrices). Thus, relation (2.34) is recasted as:

$$\psi(\Theta_n) = \boldsymbol{\Phi}_n \left(\boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n + \lambda \boldsymbol{I}_n\right)^{-1} \boldsymbol{y}_n. \tag{2.36}$$

The significance of the change from equation (2.35) to equation (2.36), is twofold. First, it is easy to show that $\boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n = \boldsymbol{K}_n$, i.e., the product $\boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n$ is identical to the kernel matrix associated with the past data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$. Moreover, as the representer theorem implies that $\psi(\Theta_n) = \boldsymbol{\Phi}_n \boldsymbol{\theta}_n$, instead of searching for an infinite dimensional vector $\psi(\Theta_n)$, we can search for the finite dimensional vector, $\boldsymbol{\theta}_n$, that satisfies:

$$\boldsymbol{\theta}_n = \left(\boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n + \lambda \boldsymbol{I}_n\right)^{-1} \boldsymbol{y}_n. \tag{2.37}$$

Let $\boldsymbol{P}_n = \lambda \boldsymbol{I}_n + \boldsymbol{\Phi}_n^T \boldsymbol{\Phi}_n$. Then applying simple algebra and using relation (2.32) we get:

$$\begin{aligned}
\boldsymbol{P}_n &= \lambda \begin{pmatrix} \boldsymbol{I}_{n-1} & \boldsymbol{0} \\ \boldsymbol{0}^T & 1 \end{pmatrix} + \begin{pmatrix} \psi(\phi(\boldsymbol{x}_1))^T \\ \psi(\phi(\boldsymbol{x}_2))^T \\ \vdots \\ \psi(\phi(\boldsymbol{x}_n))^T \end{pmatrix} \begin{pmatrix} \psi(\phi(\boldsymbol{x}_1)) & \psi(\phi(\boldsymbol{x}_2)) & \cdots & \psi(\phi(\boldsymbol{x}_n)) \end{pmatrix} \\
&= \lambda \begin{pmatrix} \boldsymbol{I}_{n-1} & \boldsymbol{0} \\ \boldsymbol{0}^T & 1 \end{pmatrix} + \begin{pmatrix} \psi(\phi(\boldsymbol{x}_1))^T \psi(\phi(\boldsymbol{x}_1)) & \psi(\phi(\boldsymbol{x}_1))^T \psi(\phi(\boldsymbol{x}_2)) & \cdots & \psi(\phi(\boldsymbol{x}_1))^T \psi(\phi(\boldsymbol{x}_n)) \\ \psi(\phi(\boldsymbol{x}_2))^T \psi(\phi(\boldsymbol{x}_1)) & \psi(\phi(\boldsymbol{x}_2))^T \psi(\phi(\boldsymbol{x}_2)) & \cdots & \psi(\phi(\boldsymbol{x}_2))^T \psi(\phi(\boldsymbol{x}_n)) \\ \vdots & \vdots & & \vdots \\ \psi(\phi(\boldsymbol{x}_n))^T \psi(\phi(\boldsymbol{x}_1)) & \psi(\phi(\boldsymbol{x}_n))^T \psi(\phi(\boldsymbol{x}_2)) & \cdots & \psi(\phi(\boldsymbol{x}_n))^T \psi(\phi(\boldsymbol{x}_n)) \end{pmatrix} \\
&= \lambda \begin{pmatrix} \boldsymbol{I}_{n-1} & \boldsymbol{0} \\ \boldsymbol{0}^T & 1 \end{pmatrix} + \begin{pmatrix} \boldsymbol{K}_{n-1} & \boldsymbol{\beta}_n \\ \boldsymbol{\beta}_n^T & \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n) \end{pmatrix} = \begin{pmatrix} \lambda \boldsymbol{I}_{n-1} + \boldsymbol{K}_{n-1} & \boldsymbol{\beta}_n \\ \boldsymbol{\beta}_n^T & \lambda + \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n) \end{pmatrix},
\end{aligned}$$

where $\boldsymbol{\beta}_n = (\kappa(\boldsymbol{x}_1, \boldsymbol{x}_n), \ldots, \kappa(\boldsymbol{x}_{n-1}, \boldsymbol{x}_n))^T$. Hence,

$$\boldsymbol{P}_n = \begin{pmatrix} \boldsymbol{P}_{n-1} & \boldsymbol{\beta}_n \\ \boldsymbol{\beta}_n^T & \lambda + \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n). \end{pmatrix} \tag{2.38}$$

At this stage we apply the matrix inversion lemma for block matrices, i.e.,

$$\begin{pmatrix} A & b \\ b^T & c \end{pmatrix}^{-1} = \begin{pmatrix} A^{-1} + \frac{1}{k} A^{-1} b b^T A^{-1} & -\frac{1}{k} A^{-1} b \\ -\frac{1}{k} b^T A^{-1} & \frac{1}{k} \end{pmatrix}, \tag{2.39}$$

to (2.38) to obtain a recursive relation for $\boldsymbol{P}_n^{-1}$, i.e.,

$$\boldsymbol{P}_n^{-1} = \begin{pmatrix} \boldsymbol{P}_{n-1}^{-1} + \frac{1}{r} \boldsymbol{P}_{n-1} \boldsymbol{\beta}_n \boldsymbol{\beta}_n^T \boldsymbol{P}_{n-1}^{-1} & -\frac{1}{r} \boldsymbol{P}_{n-1}^{-1} \boldsymbol{\beta}_n \\ -\frac{1}{r} \boldsymbol{\beta}_n^T \boldsymbol{P}_{n-1}^{-1} & \frac{1}{r} \end{pmatrix}, \tag{2.40}$$

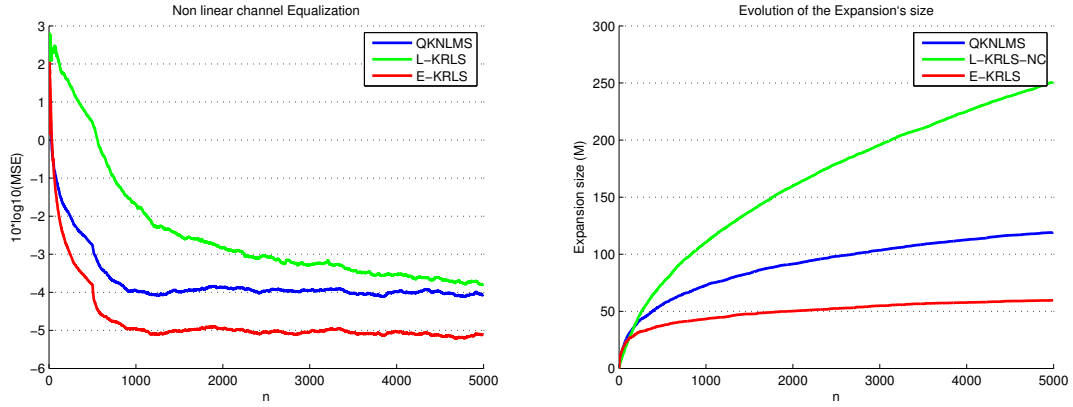
where $r = \lambda + \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n) - \gamma_n^T \boldsymbol{P}_{n-1}^{-1} \boldsymbol{\beta}_n$. Substituting in (2.37) we take the recursive relation for $\boldsymbol{\theta}_n$, i.e.,

$$\begin{aligned}
\boldsymbol{\theta}_n &= \begin{pmatrix} \boldsymbol{P}_{n-1}^{-1} + \frac{1}{r} \boldsymbol{P}_{n-1}^{-1} \boldsymbol{\beta}_n \boldsymbol{\beta}_n^T \boldsymbol{P}_{n-1}^{-1} & -\frac{1}{r} \boldsymbol{P}_{n-1}^{-1} \boldsymbol{\beta}_n \\ -\frac{1}{r} \boldsymbol{\beta}_n^T \boldsymbol{P}_{n-1}^{-1} & \frac{1}{r} \end{pmatrix} \begin{pmatrix} \boldsymbol{y}_{n-1} \\ y_n \end{pmatrix} \\
&= \begin{pmatrix} \boldsymbol{P}_{n-1}^{-1} \boldsymbol{y}_{n-1} + \frac{1}{r} \boldsymbol{P}_{n-1}^{-1} \boldsymbol{\beta}_n \boldsymbol{\beta}_n^T \boldsymbol{P}_{n-1}^{-1} \boldsymbol{y}_{n-1} - \frac{1}{r} \boldsymbol{P}_{n-1}^{-1} \boldsymbol{\beta}_n y_n \\ -\frac{1}{r} \boldsymbol{\beta}_n^T \boldsymbol{P}_{n-1}^{-1} \boldsymbol{y}_{n-1} + \frac{1}{r} y_n \end{pmatrix} \\
&= \begin{pmatrix} \boldsymbol{\theta}_{n-1} - \frac{1}{r} \boldsymbol{P}_{n-1}^{-1} \boldsymbol{\beta}_n (y_n - \boldsymbol{\beta}_n^T \boldsymbol{\theta}_{n-1}) \\ \frac{1}{r} (y_n - \boldsymbol{\beta}_n^T \boldsymbol{\theta}_{n-1}) \end{pmatrix} = \begin{pmatrix} \boldsymbol{\theta}_{n-1} - \frac{1}{r} \boldsymbol{P}_{n-1}^{-1} \boldsymbol{\beta}_n e_n \\ \frac{1}{r} e_n \end{pmatrix},
\end{aligned}$$

where the last relation emerges from the fact that $\hat{y}_n = \boldsymbol{\beta}_n^T \boldsymbol{\theta}_{n-1}$ and $e_n = y_n - \hat{y}_n$. Algorithm 7 summarizes this procedure.

Algorithm 7 Liu-Principe's Kernel Recursive Least Squares (L-KRLS)

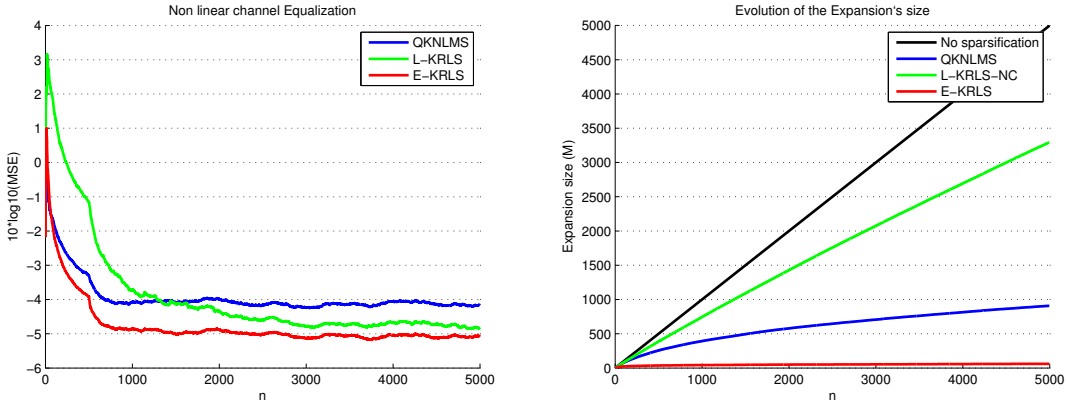
INPUT: The data $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n), \ldots, \boldsymbol{x}_n \in \mathbb{R}^\nu$, $y_n \in \mathbb{R}$.
Initialization: Set $\boldsymbol{P}^{-1} = (\lambda + \kappa(\boldsymbol{x}_1, \boldsymbol{x}_1))^{-1}$ and $\boldsymbol{\theta} = \boldsymbol{P}^{-1} y_1$.
for $n = 2, 3, \ldots$ **do**
Compute $\boldsymbol{\beta} = (\kappa(\boldsymbol{x}_1, \boldsymbol{x}_n), \kappa(\boldsymbol{x}_2, \boldsymbol{x}_n), \ldots, \kappa(\boldsymbol{x}_{n-1}\boldsymbol{x}_n))^T$.
Compute the learning system's output: $\hat{y}_n = \boldsymbol{\beta}^T \boldsymbol{\theta}$
Compute the error: $e_n = y_n - \hat{y}_n$.
Compute $r = \lambda + \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n) - \boldsymbol{\beta}^T \boldsymbol{P}^{-1} \boldsymbol{\beta}$.
compute $Q = \boldsymbol{P}^{-1}\boldsymbol{\beta}$.
Update solution: $\boldsymbol{\theta} = \begin{pmatrix} \boldsymbol{\theta} - \frac{1}{r}\boldsymbol{P}^{-1}\boldsymbol{\beta} e_n \\ \frac{1}{r} e_n \end{pmatrix}$.
Update $\boldsymbol{P}^{-1} = \begin{pmatrix} \boldsymbol{P}^{-1} + \frac{1}{r}\boldsymbol{P}\boldsymbol{\beta}\boldsymbol{\beta}^T\boldsymbol{P}^{-1} & -\frac{1}{r}\boldsymbol{P}^{-1}\boldsymbol{\beta} \\ -\frac{1}{r}\boldsymbol{\beta}^T\boldsymbol{P}^{-1} & \frac{1}{r} \end{pmatrix}$.
end for
OUTPUT: The vector $\tilde{\boldsymbol{\theta}} \in \mathbb{R}^n$.



(a) (b)

Figure 2.9: (a) Learning curves of the quantized KNLMS ($\mu = 1/2$), L-KRLS ($\lambda = 0.001$) and E-KRLS based on the ALD sparsification scenario ($\epsilon_0 = 0.1$) using the Gaussian kernel ($\sigma = 5$), for the equalization problem (2.41) (filter length $K = 5$, time delay $D = 2$). The quantization size was set to $\delta = 4$ and the parameters of the novelty criterion for the L-KRLS were set to $\delta_1 = 0.8$, $\delta_2 = 1.4$. (b) Evolution of the expansion's size at each time instant, for each algorithm.

This implementation of the KRLS has complexity in the order of $O(n^2)$, at iteration n. In addition, the network size increases linearly with the number of training data as in the KLMS. Thus, in order to make L-KRLS viable for real world applications a sparsification strategy must be adopted. Several sparsification strategies (such as the novelty and the coherence criteria) can be used readily without any modification to the algorithm. In [21], the authors propose, also, a sparsification scenario based on a simplified ALD criterion. However, the main drawback of this approach to a kernel-based RLS framework (with any of the aforementioned sparsification strategies) is that the training data, that do not qualify to enter the dictionary, are discarded without being used further to modify the solution. In contrast, E-KRLS uses all available information to update the solution at each time step. Hence, we expect that E-KRLS will exhibit faster convergence and lower steady state MSE than L-KRLS. Based on the rationale of the L-KRLS, the authors developed the Exponentially Weighted Kernel Recursive RLS [21] and the Extended Kernel Recursive Least Squares [22].



(a) (b)

Figure 2.10: (a) Learning curves of the quantized KNLMS ($\mu = 1/2$), L-KRLS ($\lambda = 0.001$) and E-KRLS based on the ALD sparsification scenario ($\epsilon_0 = 0.1$) using the Gaussian kernel ($\sigma = 5$), for the equalization problem (2.41) (filter length $K = 5$, time delay $D = 2$). The quantization size was set to $\delta = 1$ and the parameters of the novelty criterion for the L-KRLS were set to $\delta_1 = 0.15$, $\delta_2 = 0.2$. (b) Evolution of the expansion's size at each time instant, for each algorithm.

2.5.2 Simulation Results

In order to study the behavior of the KRLS, we consider some non-linear channel equalization set-ups similar to the ones used in 2.2.2. In the first experiment, the non-linear channel consists of a linear filter:

$$t_n = -0.8 \cdot y_n + 0.7 \cdot y_{n-1} - 0.6 \cdot y_{n-2} + 0.4 \cdot y_{n-3}$$

and a memoryless nonlinearity

$$q_n = t_n + 0.08 \cdot t_n^2. \tag{2.41}$$

The input signal that was fed to the channel followed the Gaussian distribution with zero mean and unite variance. At the receiver end of the channels, the signal is corrupted by white Gaussian noise and then observed as x_n. The level of the noise was set to 15dB. To solve the equalization task, we apply the KRLS-based learning systems to the set of samples $((x_n, x_{n-1}, \ldots, x_{n-M+1}), y_{n-D})$, where $K > 0$ is the equalizer's length and D the equalization time delay. Figure 2.9 shows the performance of L-KRLS and E-KRLS compared to KLMS over a set of 30 training sequences consisting of 5000 samples each. The Quantization sparsification strategy was adopted for the KLMS, as it demonstrates superior performance (see section 2.2.2). For the L-KRLS we adopted the Novelty-criterion. The sparsification parameters were set to $\epsilon_0 = 0.1$ for the E-KRLS, $\delta = 4$ for the quantization of KLMS and $\delta_1 = 0.8$, $\delta_2 = 1.4$ for the novelty criterion of L-KRLS. It is clear that E-KRLS significantly outperforms the other two algorithms (especially the L-KRLS) both in terms of convergence and sparsity. We should note that, as figure 2.10 demonstrates, the convergence rate of L-KRLS can be significantly enhanced at the cost of a significant larger expansion.

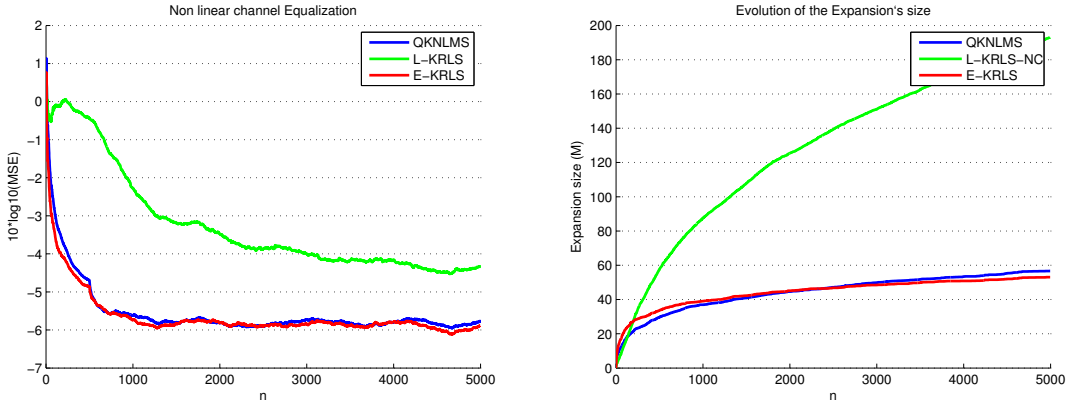
In the second experiment, the linear part of the channel is reduced, i.e.,

$$t_n = -0.8 \cdot y_n + 0.7 \cdot y_{n-1} \tag{2.42}$$

and the memoryless nonlinearity remains the same. Figure 2.11 shows the performance of the three algorithms over a set of 30 training sequences consisting of 5000 samples each. The parameters used for sparsification were set to $\epsilon_0 = 0.1$ for the E-KRLS, $\delta = 6$ for the quantization of KLMS and $\delta_1 = 0.8$, $\delta_2 = 1.4$ for the novelty criterion of L-KRLS.

Finally, in the third experiment, the linear part is identical to (2.42), while the non linear part is

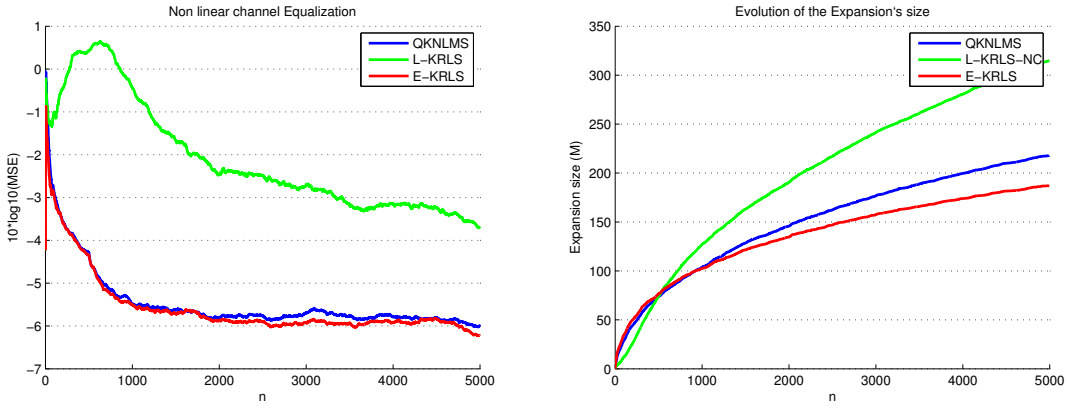
$$q_n = t_n + 0.25 \cdot t_n^2 + 0.11 \cdot t_n^3 \tag{2.43}$$



(a) (b)

Figure 2.11: (a) Learning curves of the quantized KNLMS ($\mu = 1/2$), L-KRLS ($\lambda = 0.001$) and E-KRLS based on the ALD sparsification scenario ($\epsilon_0 = 0.1$) using the Gaussian kernel ($\sigma = 5$), for the equalization problem (2.42) (filter length $K = 5$, time delay $D = 2$). The quantization size was set to $\delta = 6$ and the parameters of the novelty criterion for the L-KRLS were set to $\delta_1 = 0.8$, $\delta_2 = 1.4$. (b) Evolution of the expansion's size at each time instant, for each algorithm.

and the memoryless nonlinearity remains the same. Figure 2.12 shows the performance of the three algorithms over a set of 30 training sequences consisting of 5000 samples each, using the same parameters for the sparsification as in the second experiment. In general, E-KRLS shows the best performance compared to the other two, as it appears to achieve lower or the same steady state MSE as the QKLMS using somewhat sparser representations. However, there are cases where its performance is quite similar to the QKNLMS. The L-KRLS performs poorly compared to E-KRLS and QKNLMS, as it doesn't capture the valuable information contained in the training points that are discarded from the dictionary. The code for the experiments presented in this section can be found in `http://bouboulis.mysch.gr/kernels.html`.



(a) (b)

Figure 2.12: (a) Learning curves of the quantized KNLMS ($\mu = 1/2$), L-KRLS ($\lambda = 0.001$) and E-KRLS based on the ALD sparsification scenario ($\epsilon_0 = 0.1$) using the Gaussian kernel ($\sigma = 5$), for the equalization problem (2.43) (filter length $K = 5$, time delay $D = 2$). The quantization size was set to $\delta = 4$ and the parameters of the novelty criterion for the L-KRLS were set to $\delta_1 = 0.8$, $\delta_2 = 1.4$. (b) Evolution of the expansion's size at each time instant, for each algorithm.

Bibliography

[1] T. Adali and H. Li. *Adaptive signal processing: next generation solutions*, chapter Complex-valued Adaptive Signal Processing, pages 1–74. Wiley, NJ, 2010.

[2] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, May 1950.

[3] A. V. Balakrishnan. *Applied Functional Analysis*. Springer, 1976.

[4] S. Bergman. Ueber die entwicklung der harmonischen functionen der ebene und des raumes nach orthonalfunctionen. *Math. Ann*, 86:238–271, 1922.

[5] S. Bochner. Vorlesungen ueber fouriersche integrale, 1932.

[6] S. Bochner. Hilbert distances and positive definite functions. *Ann. of Math.*, 42:647–656, 1941.

[7] P. Bouboulis. Wirtinger's calculus in general Hilbert spaces. *Tech Report, University of Athens*, 2010.

[8] P. Bouboulis and S. Theodoridis. Extension of Wirtinger's calculus to reproducing kernel Hilbert spaces and the complex kernel LMS. *IEEE Transactions on Signal Processing*, 59(3):964–978, 2011.

[9] Pantelis Bouboulis, Kostantinos Slavakis, and Sergios Theodoridis. Adaptive learning in complex reproducing kernel hilbert spaces employing wirtingers subgradients. *IEEE Trans. Neural Networks and Learning Systems. DOI 10.1109/TNNLS.2011.2179810 (to appear)*, 2012.

[10] Pantelis Bouboulis, Sergios Theodoridis, and Michael Mavroforakis. The augmented complex kernel LMS. *IEEE Trans. Signal Process. (to appear)*, 2012.

[11] D. H. Brandwood. A complex gradient operator and its application in adaptive array theory. *IEE proc. H (Microwaves, optics and Antennas)*, 130(1):11–16, 1983.

[12] Badong Chen, Songlin Zhao, Pingping Zhu, and J.C. Principe. Quantized kernel least mean square algorithm. *Neural Networks and Learning Systems, IEEE Transactions on*, 23(1):22 –32, jan. 2012.

[13] A. Van de Bos. Complex gradient and hessian. *IEE proc. Visual image signal processing*, 141(6):380–382, 1994.

[14] L. Debnath and P. Mikusinski. *Introduction to Hilbert Spaces with applications,second edition.* Academic Press, 1999.

[15] Y. Engel, S. Mannor, and R. Meir. The kernel recursive least-squares algorithm. *IEEE Trans. Signal Process.*, 52(8):2275–2285, 2004.

[16] Simon Haykin. *Adaptive Filter Theory.* Prentice Hall, 2002.

[17] K. Kreutz-Delgado. The complex gradient operator and the $\mathbb{CR}$-calculus. `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.86.6515&rep=rep1&type=pdf`.

[18] S. Lang. *Real and Functional Analysis, third edition.* Springer, 1993.

[19] H. Li. *Complex-valued adaptive signal processing using Wirtinger calculus and its application to Independent Component Analysis*. PhD thesis, University of Maryland Baltimore County, 2008.

[20] W. Liu, P. Pokharel, and J. C. Principe. The kernel Least-Mean-Square algorithm. *IEEE Trans. Signal Process.*, 56(2):543–554, 2008.

[21] W. Liu, J. C. Principe, and S. Haykin. *Kernel Adaptive Filtering*. Wiley, 2010.

[22] Weifeng Liu, Il Park, Yiwen Wang, and J.C. Principe. Extended kernel recursive least squares algorithm. *Signal Processing, IEEE Transactions on*, 57(10):3801 –3814, oct. 2009.

[23] L. A. Liusternik and V. J. Sobolev. *Elements of Functional Analysis*. Frederick Ungar Publishing Co, 1961.

[24] D. Mandic and V.S.L Goh. *Complex Valued nonlinear Adaptive Filters*. Wiley, 2009.

[25] J. Mercer. Functions of positive and negative type and their connection with the theory of integral equation. *Philosophical Transactions of the Royal Society of London*, 209:415–446, 1909.

[26] J. Mercer. Sturm-Liuville series of normal functions in the theory of integral equations. *Philosophical Transactions of the Royal Society of London, Series A*, 211:111–198, 1911.

[27] E. H. Moore. On properly positive hermitian matrices. *Bull. Amer. Math. Soc.*, 23:59, 1916.

[28] E. H. Moore. General analysis, part i. *Memoirs of the American Philosophical Society*, 1935.

[29] E. H. Moore. General analysis, part ii. *Memoirs of the American Philosophical Society*, 1939.

[30] M. Novey and T. Adali. On extending the complex ICA algorithm to noncircular sources. *IEEE Transanctions on Signal Processing*, 56(5):2148–2154, 2008.

[31] V. I. Paulsen. An Introduction to the theory of Reproducing Kernel Hilbert Spaces. Notes, September 2009.

[32] B. Picinbono and P. Chevalier. Widely linear estimation with complex data. *IEEE Transactions on Signal Processing*, 43(8):2030–2033, 1995.

[33] C. Richard, J.C.M. Bermudez, and P. Honeine. Online prediction of time series data with kernels. *Signal Processing, IEEE Transactions on*, 57(3):1058 –1067, march 2009.

[34] T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[35] Ali. H. Sayed. *Adaptive Filters*. John Wiley & Sons, 2008.

[36] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.

[37] B. Scholkopf and A.J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, 2002.

[38] D.J. Sebald and J.A. Bucklew. Support vector machine techniques for nonlinear equalization. *Signal Processing, IEEE Transactions on*, 48(11):3217 –3226, nov 2000.

[39] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge University Press, 2004.

[40] K. Slavakis, P. Bouboulis, and S. Theodoridis. Adaptive multiregression in reproducing kernel Hilbert spaces: the multiaccess MIMO channel case. *IEEE Trans. Neural Networks*, 2011. (to appear).

[41] K. Slavakis, S. Theodoridis, and I. Yamada. Adaptive constrained learning in reproducing kernel Hilbert spaces: the robust beamforming case. *IEEE Transactions on Signal Processing*, 57(12):4744–4764, 2009.

[42] I. Steinwart, D. Hush, and C. Scovel. An explicit description of the Reproducing Kernel Hilbert spaces of Gaussian RBF kernels. *IEEE Trans. Info. Theory*, 52(10):4635–4643, 2006.

[43] S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Academic Press, 4^{th} edition, Nov. 2008.

[44] Jan Van Tiel. *Convex Analysis. An introductory text*. John Wiley & Sons Ltd, 1984.

[45] B. Widrow and M. E. Hoff. Adaptive switching circuits. *IRE WESCON Conv. Rec.*, (4):96104, 1960.

[46] W. Wirtinger. Zur formalen theorie der functionen von mehr complexen veranderlichen. *Mathematische Annalen*, 97:357–375, 1927.

[47] S. Zaremba. L' equation biharmonique et une classe remarquable de fonctions fondamentales harmoniques. *Bulletin international de l' Academie des sciences de Cracovie*, pages 147–196, 1907.

[48] S. Zaremba. Sur le calcul numerique des fonctions demandees dans le probleme de dirichlet et le probleme hydrodynamique. *Bulletin international de l' Academie des sciences de Cracovie*, 1908.