Adaptive Learning in Complex Reproducing Kernel Hilbert Spaces employing Wirtinger's subgradients

Pantelis Bouboulis, *Member, IEEE,* Konstantinos Slavakis, *Member, IEEE,* and Sergios Theodoridis, *Fellow, IEEE*

Abstract—**This paper presents a wide framework for non-linear online supervised learning tasks in the context of complex valued signal processing. The (complex) input data are mapped into a complex Reproducing Kernel Hilbert Space (RKHS), where the learning phase is taking place. Both pure complex kernels and real kernels (via the complexification trick) can be employed. Moreover, any convex, continuous and not necessarily differentiable function can be used to measure the loss between the output of the specific system and the desired response. The only requirement is the subgradient of the adopted loss function to be available in an analytic form. In order to derive analytically the subgradients, the principles of the (recently developed) Wirtinger's Calculus in complex RKHS are exploited. Furthermore, both linear and widely linear (in RKHS) estimation filters are considered. To cope with the problem of increasing memory requirements, which is present in almost all online schemes in RKHS, the sparsification scheme, based on projection onto closed balls, has been adopted. We demonstrate the effectiveness of the proposed framework in a non-linear channel identification task, a non-linear channel equalization problem and a QPSK equalization scheme, using both circular and non circular synthetic signal sources.**

Index Terms—**Wirtinger's Calculus, Complex Kernels, Adaptive Kernel Learning, Projection, Subgradient, Widely Linear Estimation**

I. Introduction

Kernel based methods have been successfully applied in many classification, regression and estimation tasks in a variety of scientific domains ranging from pattern recognition, image and signal processing to biology and nuclear physics [1]–[24]. Their appeal lies mainly on the solid and efficient mathematical background which they rely upon: the theory of Reproducing Kernel Hilbert Spaces (RKHS) [25], [26]. The main advantage of mobilizing this powerful tool of RKHS is that it offers an elegant tactic to transform a nonlinear task (in a low dimensional space) into a linear one, that is performed in a high dimensional (possible infinite) space, and which can be solved by employing an easier "algebra". Usually, this process is described through the popular *kernel trick* [1], [2]:

> "Given an algorithm, which can be formulated in terms of dot (inner) 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 F (usually $F \subset \mathbb{R}^\nu$) into a higher dimensionality (possibly infinite) RKHS $\mathcal{H}$ 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 F. 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.

Undeniably, the flagship of the so called kernel methods is the popular *Support Vector Machines* paradigm [1]–[4]. This was developed by Vapnik and Chervonenkis in the sixties and in its original form was a linear classifier. However, with the incorporation of kernels it became a powerful nonlinear processing tool with excellent generalization properties, as it is substantiated by strong theoretical arguments in the context of the Statistical Learning Theory [3], and it has been verified in practice, e.g., [2].

Motivated mainly by the success of SVMs in classification problems, a large number of kernel based methods emerged in various domains. However, most of these methods relate to batch processing, where all necessary data are available beforehand. Over the last five years, significant efforts have been devoted to the development of online kernel methods for adaptive learning (e.g., adaptive filtering) [5]–[12], where the data arrive sequentially. However, all the aforementioned kernel methods (batch and online) were targeted for applications of real data sequences.

Complex-valued signals arise frequently in applications as diverse as communications, biomedicine, radar, etc. 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. Therefore, it is natural to wonder whether we should be able to apply the machinery of kernels to handle learning tasks in complex domains. However, although real RKHS have become quite popular and they have been used in a large number of applications, complex kernels (such as the complex Gaussian RBF kernel), while known to the mathematicians (especially those working on Reproducing Kernel Hilbert Spaces or Functional Analysis), have rather remained in obscurity in the Machine Learning

P. Bouboulis is with the Department of Informatics and Telecommunications, University of Athens, Greece, e-mail: pbouboulis@sch.gr.

S. Theodoridis is with the Department of Informatics and Telecommunications, University of Athens, Greece, and the Research Academic Computer Technology Institute, Patra, Greece. e-mail: stheodor@di.uoa.gr.

K. Slavakis is with the Department of Telecommunications Science and Technology, University of Peloponnese, Tripolis, Greece, email: slavakis@uop.gr.

and Signal Processing communities. Only recently, in [19], a unified framework for processing in complex RKHS has been developed. This can be achieved either by using popular real kernels (such as the Gaussian RBF), taking advantage of a technique called *complexification* of real RKHS, or by employing any pure complex kernel (such as the complex Gaussian RBF). In [19], this framework was applied to the complex Least Mean Squares (LMS) task and two realizations of the complex Kernel LMS (CKLMS) were developed.

In the more traditional setting, treating complex valued signals is often followed by (implicitly) assuming the *circularity* of the signal. Circularity is intimately related to the rotation in the geometric sense. A complex random variable Z is called circular, if for any angle ϕ, both Z and $Ze^{i\phi}$ (i.e., the rotation of Z by angle ϕ) follow the same probability distribution [27], [28]. Naturally, this assumption limits the area for applications, since many practical signals exhibit non-circular characteristics. Thus, following the ideas originated by Picinbono in [29], [30], on-going research is focusing on the *widely linear* filters (or *augmented* filters) in the complex domain (see, for example, [27], [28], [31]–[43]). The main characteristic of such filters is that they exploit simultaneously both the original signal as well as its conjugate analogue.

The present paper builds upon the rationale of [19] and extends the theory to the case of complex subgradients, to be used in the context of the powerful Adaptive Projected Subgradient Method (APSM) [44]–[46], both for linear and widely linear formulations. The APSM employs concepts of operators in Hilbert spaces, [47], in order to derive efficient generalizations of classical adaptive filtering concepts, [48], [49] and significantly facilitate the treatment of convexly constrained time-adaptive learning tasks, [50]. Thus, in this study, the APSM machinery is extended to the complex case, to provide a wide toolbox for adaptive learning in complex RKHS. In this context, any convex function (not necessarily differentiable) can be used as a measure of loss in the learning task. The only requirement is that the subgradients of the loss function must take an analytic form. To infuse robustness into the design, the *ϵ-insensitive* version of the corresponding chosen loss function is utilized, due to its attractive features, which are widely known in robust statistics, [1], [3], [51]. As this method employs subgradients in the minimization process, *Wirtinger's Calculus* is further extended and the notion of the Wirtinger's subgradients is introduced. To the best of our knowledge, this is the first time that this notion is developed in the respective literature, and its value goes beyond the current context of APSM and can be used in any optimization task, that involves subgradients in complex spaces.

The paper is organized as follows. In section II, the main properties of RKHS are presented and the differences between real and complex RKHS are highlighted. In section III, the main characteristics of the recently developed Wirtinger's calculus in complex RKHS are briefly sketched, before the notion of Wirtinger's subgradients for real functions of complex variables is introduced. Applying this newly developed tool, we compute the subgradients of the ϵ-insensitive versions of several popular loss functions (e.g., l_2, l_1, Huber). The complexification and the pure complex kernelization procedures are also described there. Section IV presents a detailed description of the proposed algorithmic scheme, i.e., the Complex Kernel Adaptive Projected Subgradient Method, for adaptive filtering problems. Finally, Section V provides experimental results in the context of (non-linear) channel identification and equalization tasks. Throughout the paper, we will denote the set of non negative integers, real and complex numbers by $\mathbb{N}, \mathbb{R}, \mathbb{C}$ respectively. For any integers $k_1 \leq k_2$, by $\overline{k_1, k_2}$ we denote the set $\{k_1, k_1+1, \dots, k_2\}$. The complex unit is denoted as $i = \sqrt{-1}$. Vector and matrix valued quantities appear in boldface symbols.

II. Reproducing Kernel Hilbert Spaces

In this section, we briefly describe the theory of Reproducing Kernel Hilbert Spaces, as this is the main mathematical tool employed in this study. Since we are interested in both real and complex kernels, we recall the basic facts on RKHS associated with a general field $\mathbb{F}$, which can be either $\mathbb{R}$ or $\mathbb{C}$. However, we highlight the basic differences between the two cases. The interested reader may dig deeper on this subject by referring to [52] (among others).

Given a function $\kappa : X \times X \to \mathbb{F}$ and $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N \in X$ (typically X is a subset of $\mathbb{R}^\nu$ or $\mathbb{C}^\nu$, $\nu > 0$), the matrix[1] $\boldsymbol{K} = (K_{n,m})^N$ with elements $K_{n,m} = \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m)$, for $n, m = 1, \dots, N$, is called the *Gram matrix* (or *kernel matrix*) of κ with respect to $x_1, \dots, x_N$. A Hermitian matrix $\boldsymbol{K} = (K_{n,m})^N$ satisfying

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

for all $c_n \in \mathbb{F}$, $n = 1, \dots, N$, where the notation * denotes the conjugate element $and \cdot^H$ the Hermitian matrix, is called *Positive Definite*. In matrix analysis literature, this is the definition of a positive semidefinite matrix. However, since this is a rather cumbersome term and the distinction between positive definite and positive semidefinite matrices is not important in this paper, we employ the term positive definite, as it was already defined. A function $\kappa : X \times X \to \mathbb{F}$, which for all $N \in \mathbb{N}$ and all $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N \in X$ 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*.

Next, consider a linear class $\mathcal{H}$ of complex 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 $\mathcal{H}$.
2) κ has the so called *reproducing property*, i.e.,

$$f(\boldsymbol{x}) = \langle f, \kappa(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}}, \text{ for all } f \in \mathcal{H}, \boldsymbol{x} \in X, \tag{1}$$

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

[1] The term $(K_{n,m})^N$ denotes a square $N \times N$ matrix.

It has been shown (see [25]) that to every positive definite kernel κ there corresponds one class of functions $\mathcal{H}$ with a uniquely determined inner product in it, forming a Hilbert space and admitting κ as a reproducing kernel. In fact, the kernel κ produces the entire space $\mathcal{H}$, i.e., $\mathcal{H} = \overline{\text{span}\{\kappa(\boldsymbol{x},\cdot)|\boldsymbol{x}\in X\}}$[2]. The map $\Phi : X \to \mathcal{H} : \Phi(\boldsymbol{x}) = \kappa(\cdot,\boldsymbol{x})$ is called the *feature map* of $\mathcal{H}$. Recall, that in the case of complex Hilbert spaces (i.e., $\mathbb{F}=\mathbb{C}$) the inner product is sesqui-linear (i.e., linear in one argument and antilinear 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, the condition $\kappa(\boldsymbol{x},\boldsymbol{y}) = \langle\kappa(\cdot,\boldsymbol{y}),\kappa(\cdot,\boldsymbol{x})\rangle_{\mathcal{H}}$ may be replaced by $\kappa(\boldsymbol{x},\boldsymbol{y}) = \langle\kappa(\cdot,\boldsymbol{x}),\kappa(\cdot,\boldsymbol{y})\rangle_{\mathcal{H}}$. However, since in the complex case the inner product is Hermitian, the aforementioned condition is equivalent to $\kappa(\boldsymbol{x},\boldsymbol{y}) = (\langle\kappa(\cdot,\boldsymbol{x}),\kappa(\cdot,\boldsymbol{y})\rangle_{\mathcal{H}})^*$.

Although, the underlying theory has been developed by the mathematicians for general complex reproducing kernels and their associated RKHSs, it is the case of the real kernels that has been considered, mainly, by the Machine Learning and Signal Processing communities. Some of the most widely used kernels are the *Gaussian RBF*, i.e.,

$$\kappa_{\sigma,\mathbb{R}^d}(\boldsymbol{x},\boldsymbol{y}) := \exp\left(-\frac{\sum_{k=1}^d (x_k-y_k)^2}{\sigma^2}\right), \tag{2}$$

defined for $\boldsymbol{x},\boldsymbol{y}\in\mathbb{R}^d$, where σ is a free positive parameter and the *polynomial kernel*: $\kappa_d(\boldsymbol{x},\boldsymbol{y}) := \left(1+\boldsymbol{x}^T\boldsymbol{y}\right)^d$, for $d\in\mathbb{N}$, where $\cdot^T$ stands for the transpose matrix. Many more kernels emerging from various aspects of mathematics (ranging form splines and wavelets to fractals) can be found in the related literature [1], [2], [4], [53].

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 $|z|,|w|<1$ [52]. Another important complex kernel, that has remained relatively unknown in the Machine Learning and Signal Processing communities, is the *complex Gaussian kernel*

$$\kappa_{\sigma,\mathbb{C}^d}(\boldsymbol{z},\boldsymbol{w}) := \exp\left(-\frac{\sum_{k=1}^d (z_k-w_k^*)^2}{\sigma^2}\right), \tag{3}$$

defined on $\mathbb{C}^d\times\mathbb{C}^d$, where $\boldsymbol{z},\boldsymbol{w}\in\mathbb{C}^d$, z_k denotes the k-th component of the complex vector $\boldsymbol{z}\in\mathbb{C}^d$ and $\exp$ is the extended exponential function in the complex domain. 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* (2). An explicit description of the RKHSs of these kernels, together with some important properties can be found in [54].

III. Working on complex RKHS

A. Wirtinger's Calculus on complex RKHS

Wirtinger's calculus [55] (or CR-Calculus) was brought into light recently [27]–[29], [31], [56], [57], as a means to compute, in an efficient and elegant way, gradients of real valued cost functions that are defined on complex domains ($\mathbb{C}^\nu$). Although 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 [19], the notion of Wirtinger's calculus was extended to general complex Hilbert spaces, providing the tool to compute the gradients that are needed to develop kernel-based algorithms for treating complex data. This extension mainly uses the notion of the *Fréchet differentiability*, which is a path to generalize differentiability to general Hilbert spaces. In this section, however, we give a brief description and thus we do not get into much details about Fréchet differentiability. The interested reader may find more on the subject in [19], [58].

We begin our discussion with some basic definitions. Let $X\subseteq\mathbb{R}^\nu$. Define $X^2\equiv X\times X\subseteq\mathbb{R}^{2\nu}$ and $\mathbb{X} = \{\boldsymbol{z} = \boldsymbol{x}+i\boldsymbol{y}|\boldsymbol{x},\boldsymbol{y}\in X\}\subseteq\mathbb{C}^\nu$, which is equipped with a complex product structure. Let $\mathcal{H}$ be a real RKHS associated with a real kernel κ defined on $X^2\times X^2$ and let $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ be its corresponding inner product. Note that, under the mapping $(\boldsymbol{x},\boldsymbol{y})\to\boldsymbol{z}=\boldsymbol{x}+i\boldsymbol{y}$, X^2 is isomorphic to $\mathbb{X}$ (under the same mapping, $\mathbb{R}^2$ is isomorphic to $\mathbb{C}$). Thus, every real valued $f\in\mathcal{H}$ can be regarded as a function defined on either X^2 or $\mathbb{X}$, i.e., $f(\boldsymbol{z}) = f(\boldsymbol{x}+i\boldsymbol{y}) = f(\boldsymbol{x},\boldsymbol{y})$, for $\boldsymbol{z}=\boldsymbol{x}+i\boldsymbol{y}$. Next, we define $\mathcal{H}^2 = \mathcal{H}\times\mathcal{H}$. It is easy to verify that $\mathcal{H}^2$ is also a Hilbert Space with inner product

$$\langle\boldsymbol{f},\boldsymbol{g}\rangle_{\mathcal{H}^2} = \langle f_1,g_1\rangle_{\mathcal{H}} + \langle f_2,g_2\rangle_{\mathcal{H}}, \tag{4}$$

for $\boldsymbol{f}=(f_1,f_2)$, $\boldsymbol{g}=(g_1,g_2)$. Our objective is to enrich $\mathcal{H}^2$ with a complex structure. To this end, we define the RKHS $\mathbb{H} = \{\boldsymbol{f}=f_1+if_2|f_1,f_2\in\mathcal{H}\}$ equipped with the complex inner product:

$$\langle\boldsymbol{f},\boldsymbol{g}\rangle_{\mathbb{H}} = \langle f_1,g_1\rangle_{\mathcal{H}} + \langle f_2,g_2\rangle_{\mathcal{H}} + i\left(\langle f_2,g_1\rangle_{\mathcal{H}} - \langle f_1,g_2\rangle_{\mathcal{H}}\right).$$

Similarly to the case of X^2 and $\mathbb{X}$, under the mapping $(f_1,f_2)\to\boldsymbol{f}=f_1+if_2$, $\mathcal{H}^2$ becomes isomorphic to $\mathbb{H}$.

Consider the function $\boldsymbol{T}: A\subseteq\mathbb{H}\to\mathbb{C}$, $\boldsymbol{T}(\boldsymbol{f}) = T_r(\boldsymbol{f}) + iT_i(\boldsymbol{f})$ defined on an open subset A of $\mathbb{H}$, where $u_{\boldsymbol{f}},v_{\boldsymbol{f}}\in\mathcal{H}$, $\boldsymbol{f}=u_{\boldsymbol{f}}+iv_{\boldsymbol{f}}$ and T_r,T_i are real valued functions defined on $\mathcal{H}^2$. Due to the isomorphisms $\mathbb{R}^2\simeq\mathbb{C}$ and $\mathcal{H}^2\simeq\mathbb{H}$, we may equivalently write:

$$\boldsymbol{T}(\boldsymbol{f}) = \left(T_r(u_{\boldsymbol{f}},v_{\boldsymbol{f}}),T_i(u_{\boldsymbol{f}},v_{\boldsymbol{f}})\right)^T, \tag{5}$$

or

$$\boldsymbol{T}(\boldsymbol{f}) = \boldsymbol{T}(u_{\boldsymbol{f}}+iv_{\boldsymbol{f}}) = T_r(u_{\boldsymbol{f}},v_{\boldsymbol{f}}) + iT_i(u_{\boldsymbol{f}},v_{\boldsymbol{f}}). \tag{6}$$

[2] The overbar denotes the closure of the set.

Based on the isomorphism between $\mathcal{H}^2$ and $\mathbb{H}$, two types of differentiability may be considered. In the first case, if we apply the notion of the Fréchet differentiability on $\boldsymbol{T}:\mathcal{H}^2\to\mathbb{R}^2$ (5), we may define the derivative of $\boldsymbol{T}$ at $\boldsymbol{c}=(\boldsymbol{u_c},\boldsymbol{v_c})$ and the respective gradient, i.e., $\nabla\boldsymbol{T}(\boldsymbol{c})$, as well as the partial derivatives of $\boldsymbol{T}$ at $\boldsymbol{c}$, $\nabla_{\boldsymbol{u}}\boldsymbol{T}(\boldsymbol{c})$ and $\nabla_{\boldsymbol{v}}\boldsymbol{T}(\boldsymbol{c})$. If $\nabla\boldsymbol{T}(\boldsymbol{c})$ exists, we will say that $\boldsymbol{T}$ is *Fréchet differentiable in the real sense*. On the other hand, applying the notion of Fréchet differentiability on $\boldsymbol{T}:\mathbb{H}\to\mathbb{C}$ (6), we may define the complex derivative of $\boldsymbol{T}$ at $\boldsymbol{c}=\boldsymbol{u_c}+i\boldsymbol{v_c}$, i.e., $d\boldsymbol{T}(\boldsymbol{c})$. In this case, if $d\boldsymbol{T}(\boldsymbol{c})$ exists, we will say that $\boldsymbol{T}$ is *Fréchet differentiable in the complex sense*. However, the notion of complex differentiability is rather strict and it excludes the case of real valued functions T (these, as it can easily be shown, do not obey the Cauchy-Riemann conditions), which are present in all optimization tasks (taking the form of cost functions). The goal in this work is to employ Wirtinger's calculus as an alternative tool for computing derivatives. Wirtinger's calculus, although based on the Fréchet differentiability in the real sense, exploits the complex algebra to make the computations easier and the derived formulas more compact [19], [58].

Definition 1. We define the *Fréchet Wirtinger's gradient* (or *W-gradient* for short) of $\boldsymbol{T}$ at $\boldsymbol{c}$ as

$$\begin{aligned}\nabla_{\boldsymbol{f}}\boldsymbol{T}(\boldsymbol{c})&=\frac{1}{2}\left(\nabla_u\boldsymbol{T}(\boldsymbol{c})-i\nabla_v\boldsymbol{T}(\boldsymbol{c})\right) \qquad (7)\\ &=\frac{1}{2}\left(\nabla_uT_r(\boldsymbol{c})+\nabla_vT_i(\boldsymbol{c})\right)+\frac{i}{2}\left(\nabla_uT_i(\boldsymbol{c})-\nabla_vT_r(\boldsymbol{c})\right),\end{aligned}$$

and the *Fréchet conjugate Wirtinger's gradient* (or *CW-gradient* for short) of $\boldsymbol{T}$ at $\boldsymbol{c}$ as

$$\begin{aligned}\nabla_{\boldsymbol{f}^*}\boldsymbol{T}(\boldsymbol{c})&=\frac{1}{2}\left(\nabla_u\boldsymbol{T}(\boldsymbol{c})+i\nabla_v\boldsymbol{T}(\boldsymbol{c})\right) \qquad (8)\\ &=\frac{1}{2}\left(\nabla_uT_r(\boldsymbol{c})-\nabla_vT_i(\boldsymbol{c})\right)+\frac{i}{2}\left(\nabla_uT_i(\boldsymbol{c})+\nabla_vT_r(\boldsymbol{c})\right).\end{aligned}$$

Remark 1. The rationale that underlies these particular definitions becomes apparent if one considers the Taylor expansion formula of $\boldsymbol{T}$. In [19], it is shown that

$$\begin{aligned}\boldsymbol{T}(\boldsymbol{c}+\boldsymbol{h})=&\boldsymbol{T}(\boldsymbol{c})+\frac{1}{2}\left\langle\boldsymbol{h},\left(\nabla_u\boldsymbol{T}(\boldsymbol{c})-i\nabla_v\boldsymbol{T}(\boldsymbol{c})\right)^*\right\rangle_{\mathbb{H}} \qquad (9)\\ &+\frac{1}{2}\left\langle\boldsymbol{h}^*,\left(\nabla_u\boldsymbol{T}(\boldsymbol{c})+i\nabla_v\boldsymbol{T}(\boldsymbol{c})\right)^*\right\rangle_{\mathbb{H}}+o(\|\boldsymbol{h}\|_{\mathbb{H}}).\end{aligned}$$

The main rules of the generalized calculus can be found in [19]. In view of these properties, one might easily compute the W and CW gradients of any complex function $\boldsymbol{T}$, which is written in terms of $\boldsymbol{f}$ and $\boldsymbol{f}^*$, following the following simple tricks:

- *To compute the W-derivative of a function $\boldsymbol{T}$, which is expressed in terms of $\boldsymbol{f}$ and $\boldsymbol{f}^*$, apply the differentiation rules considering $\boldsymbol{f}^*$ as a constant.*
- *To compute the CW-derivative of a function $\boldsymbol{T}$, which is expressed in terms of $\boldsymbol{f}$ and $\boldsymbol{f}^*$, apply the differentiation rules considering $\boldsymbol{f}$ as a constant.*

B. Wirtinger's Subgradients

Since subgradients of operators, which are defined on Hilbert spaces, play a crucial role in several parts of this paper, it is important to present their formal definition. For real valued convex functions, defined on real Hilbert spaces, the gradient at x_0 satisfies the well known first order condition:

$$T(z)\geq T(x_0)+\langle\nabla T(x_0),z-x_0\rangle,$$

for all z. This condition has a simple geometric meaning when T is finite at x_0: it says that the graph of the affine function $h(z)=T(x_0)+\langle\nabla T(x_0),z-x_0\rangle$ is a non-vertical supporting hyperplane to the convex set $\operatorname{epi}T$[3] at $(x_0,T(x_0))$. In other words, (a) $h(z)$ defines an osculant hyperplane of the graph of T at $(x_0,T(x_0))$ and (b) all the points of the graph of T lie at the same side of the hyperplane. Moreover, it is well known that, in optimization tasks, the gradient direction guarantees a path towards the optimal point. If T is not Fréchet differentiable at x, we can still construct such an osculant hyperplane (and a corresponding path towards the optimal point) using the subgradient.

Definition 2. Let $T:\mathcal{H}\to\mathbb{R}$ be a convex function defined on a real Hilbert space $(H,\langle\cdot,\cdot\rangle_H)$. A vector $\boldsymbol{x}^\partial\in\mathcal{H}$ is said to be a *subgradient* of T at $\boldsymbol{x}_0$ if

$$T(z)\geq T(\boldsymbol{x}_0)+\langle\boldsymbol{x}^\partial,z-\boldsymbol{x}_0\rangle_{\mathcal{H}}. \qquad (10)$$

The set of all subgradients of T at $\boldsymbol{x_0}$ is called the *subdifferential* of T at $\boldsymbol{x}_0$ and is denoted by $\partial T(\boldsymbol{x}_0)$.

The notion of the subgradient is a generalization of the classical differential of a function T, at $\boldsymbol{x}_0$, and it has proved itself an indispensable tool for modern optimization tasks, which involve objective functions that are not differentiable [47], [59]. In the case of real-valued objective functions, the use of the subgradient has shown a very rich potential for demanding adaptive learning optimization tasks, e.g., [12], [50], [60] and has also been popularized in the context of ℓ_1 norms in the framework of compressive sensing.

It is clear that definition 2 cannot be applied for complex valued functions, as it involves inequalities. Nevertheless, our objective is to introduce a Wirtinger-like subgradient for the special case of a real function $T:\mathbb{H}\to\mathbb{R}$. Suppose that $\nabla^sT(\boldsymbol{c})=\left(\nabla_u^sT(\boldsymbol{c}),\nabla_v^sT(\boldsymbol{c})\right)^T$ is a subgradient of T at $\boldsymbol{c}=(c_u,c_v)^T$, if we consider that T is defined on $\mathcal{H}^2$ instead of $\mathbb{H}$. Then the following inequalities hold

$$\begin{aligned}T(\boldsymbol{c}+\boldsymbol{h})\geq&T(\boldsymbol{c})+\langle\boldsymbol{h},\nabla^sT(\boldsymbol{c})\rangle_{\mathcal{H}^2}\\ \geq&T(\boldsymbol{c})+\left\langle\begin{pmatrix}h_u\\h_v\end{pmatrix},\begin{pmatrix}\nabla_u^sT(\boldsymbol{c})\\\nabla_v^sT(\boldsymbol{c})\end{pmatrix}\right\rangle_{\mathcal{H}^2}\\ \geq&T(\boldsymbol{c})+\langle h_u,\nabla_u^sT(\boldsymbol{c})\rangle_{\mathcal{H}}+\langle h_v,\nabla_v^sT(\boldsymbol{c})\rangle_{\mathcal{H}}. \qquad (11)\end{aligned}$$

This will be our kick off point for the derivation of the respective Wirtinger-like subgradients.

Definition 3. To be inline with the definition of Wirtinger gradients, we define the *Wirtinger subgradient* of $T:\mathbb{H}\to\mathbb{R}$

[3] $\operatorname{epi}T$ denotes the epigraph of T, i.e. the set $\{(x,y):x\in H,y\in\mathbb{R}:T(x)\leq y\}$.

at $\boldsymbol{c} = c_u + i \cdot c_v$ as

$$\nabla^s_{\boldsymbol{f}} T(\boldsymbol{c}) = \frac{1}{2}\left(\nabla^s_u T(\boldsymbol{c}) - i \cdot \nabla^s_v T(\boldsymbol{c})\right), \tag{12}$$

and the *conjugate Wirtinger subgradient* of T at $\boldsymbol{c}$ as

$$\nabla^s_{\boldsymbol{f}^*} T(\boldsymbol{c}) = \frac{1}{2}\left(\nabla^s_u T(\boldsymbol{c}) + i \cdot \nabla^s_v T(\boldsymbol{c})\right), \tag{13}$$

for any ordinary subgradient $\nabla^s T(\boldsymbol{c})$ of T at $\boldsymbol{c} = (c_u, c_v)^T$. The set of all conjugate Wirtinger's subgradients of T at $\boldsymbol{c}$ is called the *Wirtinger subdifferential* of T at $\boldsymbol{c}$ and is denoted by $\partial_{\boldsymbol{f}^*} T(\boldsymbol{c})$.

Under the scope of the aforementioned definitions, as $h_u = \frac{\boldsymbol{h}+\boldsymbol{h}^*}{2}$ and $h_v = \frac{\boldsymbol{h}-\boldsymbol{h}^*}{2i}$, one can easily prove that

$$\begin{aligned}\langle \boldsymbol{h}, \nabla^s T(\boldsymbol{c})\rangle_{\mathcal{H}^2} =& \langle h_u, \nabla^s_u T(\boldsymbol{c})\rangle_{\mathcal{H}} + \langle h_v, \nabla^s_v T(\boldsymbol{c})\rangle_{\mathcal{H}} \\ =& \langle \boldsymbol{h}, (\nabla^s_{\boldsymbol{f}} T(\boldsymbol{c}))^*\rangle_{\mathbb{H}} + \langle \boldsymbol{h}^*, (\nabla^s_{\boldsymbol{f}^*} T(\boldsymbol{c}))^*\rangle_{\mathbb{H}} \\ =& 2\Re\left(\langle \boldsymbol{h}, (\nabla^s_{\boldsymbol{f}} T(\boldsymbol{c}))^*\rangle_{\mathbb{H}}\right).\end{aligned}$$

Therefore, from (11) we obtain

$$T(\boldsymbol{c}+\boldsymbol{h}) \geq T(\boldsymbol{c}) + \langle \boldsymbol{h}, (\nabla^s_{\boldsymbol{f}} T(\boldsymbol{c}))^*\rangle_{\mathbb{H}} + \langle \boldsymbol{h}^*, (\nabla^s_{\boldsymbol{f}^*} T(\boldsymbol{c}))^*\rangle_{\mathbb{H}}. \tag{14}$$

or equivalently

$$T(\boldsymbol{c}+\boldsymbol{h}) \geq T(\boldsymbol{c}) + 2\Re\left(\langle \boldsymbol{h}, (\nabla^s_{\boldsymbol{f}} T(\boldsymbol{c}))^*\rangle_{\mathbb{H}}\right). \tag{15}$$

Relation (14) can be thought of as a type of Wirtinger subgradient inequality (i.e., an inequality similar to (10) in definition 2). At this point, we should note, that although we defined two types of Wirtinger subgradients, in order to be consistent with the definition of Wirtinger gradients, only one will be used (the $\nabla^s_{\boldsymbol{f}^*} T(\boldsymbol{c})$) in the subsequent sections, as the second subgradient, $\nabla^s_{\boldsymbol{f}} T(\boldsymbol{c})$, is $\nabla^s_{\boldsymbol{f}^*} T(\boldsymbol{c})$'s conjugate. The following Lemma is an extension of the one presented in [61], Theorem 25.6:

Lemma 1. *Let $l : \mathbb{C}^\nu \to \mathbb{R}$ be a convex continuous function. Also, let $\mathcal{X}(\boldsymbol{c})$ be the set of all limit points of sequences of the form $(\nabla_{\boldsymbol{f}^*} l(\boldsymbol{z}_n))_{n\in\mathbb{N}}$, where $(\boldsymbol{z}_n)_{n\in\mathbb{N}}$ is a sequence of points at which l is Fréchet differentiable in the real sense and $lim_{n\to\infty}\boldsymbol{z}_n = \boldsymbol{c}$. Then, the Wirtinger subdifferential of l at $\boldsymbol{c}$ is given by $\partial_{\boldsymbol{f}^*} l(\boldsymbol{c}) = \overline{\mathrm{conv}}(\mathcal{X}(\boldsymbol{c}))$, where* conv *denotes the convex hull of a set, and the overline symbol stands for the closure of a set.*

Proof: Observe that the conjugate Wirtinger subgradient is given by $\nabla^s_{\boldsymbol{f}^*} l(\boldsymbol{c}) = \frac{1}{2}\left(\nabla^s_u l(\boldsymbol{c}) + i \cdot \nabla^s_v l(\boldsymbol{c})\right)$, while the Fréchet subgradient obtained if we consider $l(\boldsymbol{c})$ defined on $\mathbb{R}^{2\nu}$ is $\nabla^s l(\boldsymbol{x}, \boldsymbol{y}) = (\nabla^s_u l(\boldsymbol{x}, \boldsymbol{y}), \nabla^s_v l(\boldsymbol{x}, \boldsymbol{y}))^T$. Similar results hold for the Wirtinger gradient and the Fréchet gradient if l is differentiable at $\boldsymbol{c}$. This implies an 1-1 correspondence between the conjugate Wirtinger subgradient $\nabla^s_{\boldsymbol{f}^*} l(\boldsymbol{c})$ and the Fréchet subgradient $\frac{1}{2}\nabla^s l(\boldsymbol{x}, \boldsymbol{y})$. A similar correspondence exists between the conjugate Wirtinger gradient $\nabla_{\boldsymbol{f}^*} l(\boldsymbol{c})$ and the Fréchet gradient $\frac{1}{2}\nabla l(\boldsymbol{x}, \boldsymbol{y})$. Hence, applying Theorem 25.6 of [61] gives the result. ■

Remark 2. We emphasize that the definition of the Wirtinger subgradient is a general one and it can be employed whenever a function T is not Fréchet differentiable in the real sense. In this paper, we exploit its use to derive the necessary subgradients in the context of the Adaptive Projected Subgradient Method.

C. Mapping data to complex RKHS

This paper considers the case of supervised (regression) learning, i.e., the scenario where a sequence $\boldsymbol{z}_n \in \mathbb{C}^\nu$, of complex input data, and a sequence $d_n \in \mathbb{C}$, of complex desired responses, are available to the designer. No assumption is made on the stochastic processes hidden behind the sequence $(\boldsymbol{z}_n, d_n)_{n\geq 0}$. Moreover, the stochastic properties of both $(\boldsymbol{z}_n)_{n\geq 0}$ and $(d_n)_{n\geq 0}$ are susceptible to change as n grows larger. In the supervised learning scenario, one typically estimates the output as $\hat{d}_n = \boldsymbol{D}_n(\boldsymbol{z}(n))$, where $\boldsymbol{D}_n$ are time-varying sequences of complex functions, so that the "disagreement" between d_n and $\hat{d}_n$, measured via a user-defined loss function, i.e., $l(d_n - \hat{d}_n)$, obtains some small value. The choice of the space, where $\boldsymbol{D}_n$ lives, determines the accuracy of the estimation. For example, in a typical CLMS task, $\boldsymbol{D}_n$ are $\mathbb{C}$-linear functions, i.e., $\boldsymbol{D}_n(\boldsymbol{z}) = \boldsymbol{w}_n^H \boldsymbol{z}$, for some $\boldsymbol{w}_n \in \mathbb{C}^\nu$, while in a typical widely linear LMS (WL-LMS) task they take the form of $\mathbb{R}$-linear functions, i.e., $\boldsymbol{D}_n(\boldsymbol{z}) = \boldsymbol{w}^H \boldsymbol{z} + \boldsymbol{v}^H \boldsymbol{z}^*$, for some $\boldsymbol{w}_n, \boldsymbol{v}_n \in \mathbb{C}^\nu$ [62]. In the machinery presented in this paper, $\boldsymbol{D}_n$ are non-linear functions implicitly defined by the specific choice of the complex RKHS, where the input complex data $\boldsymbol{z}_n$ are mapped.

We can perform such transformations either by employing pure complex kernels, or via traditional real kernels, as it has been substantiated in [19]. For the first case, which we will henceforth call *pure complex kernelization* procedure, the choice of a complex kernel $\kappa_{\mathbb{C}}$ implicitly defines the complex RKHS $\mathbb{H}$, where the data are mapped. The transformation from input space $\mathbb{C}^\nu$ to the complex RKHS takes place via the feature map $\boldsymbol{\Phi}(\boldsymbol{z}) = \kappa_{\mathbb{C}}(\cdot, \boldsymbol{z})$ of $\mathbb{H}$. Thus, the input/output training sequence $(\boldsymbol{z}_n, d_n)$ is transformed to $(\boldsymbol{\Phi}(\boldsymbol{z}_n), d_n)$. The learning phase is taking place in the transformed data.

As an alternative, we can employ popular well-established real kernels defined on $\mathbb{R}^{2\nu}$, using the *complexification* procedure [19]. In this case, the induced RKHS $\mathcal{H}$ is a real one. However, we can define $\mathcal{H}^2$ and enrich it with a complex structure, i.e., construct the complex RKHS $\mathbb{H}$, as it is described in details in section III-A. Note that in this case, while $\mathbb{H}$ is a complex RKHS, its "generating" kernel is a real one, i.e., $\kappa_{\mathbb{R}}$. We map the input data to $\mathbb{H}$ using the following simple rule:

$$\hat{\boldsymbol{\Phi}}(\boldsymbol{z}) = \hat{\boldsymbol{\Phi}}(\boldsymbol{x} + i\boldsymbol{y}) = \hat{\boldsymbol{\Phi}}(\boldsymbol{x}, \boldsymbol{y}) = \Phi(\boldsymbol{x}, \boldsymbol{y}) + i\Phi(\boldsymbol{x}, \boldsymbol{y}), \tag{16}$$

where Φ is the feature map of the real reproducing kernel κ, i.e., $\Phi(\boldsymbol{x}, \boldsymbol{y}) = \kappa_{\mathbb{R}}(\cdot, (\boldsymbol{x}, \boldsymbol{y}))$. It must be emphasized, that $\hat{\boldsymbol{\Phi}}$ is not the feature map associated with the complex RKHS $\mathbb{H}$. Note that we cannot choose Φ to map the input data to $\mathbb{H}$, if we want to exploit the complex structure of $\mathbb{H}$, as Φ doesn't have an imaginary component.

D. Linear and Widely Linear estimation in complex RKHS

Following the mapping of the input training data to the complex RKHS, a linear, or a widely linear estimation function is employed. Thus, after the pure complex kernelization procedure, we adopt the time adaptive $\mathbb{R}$-linear (in RKHS) estimation function $\boldsymbol{D}_n(\boldsymbol{w},\boldsymbol{v}) = \langle \boldsymbol{\Phi}(\boldsymbol{z}_n), \boldsymbol{w}\rangle_{\mathbb{H}} + \langle \boldsymbol{\Phi}(\boldsymbol{z}_n)^*, \boldsymbol{v}\rangle_{\mathbb{H}}$. This is inline with the widely-linear estimation rationale [29], [30], where both the original data and their conjugate analogue are taken into account. The objective of the proposed machinery is to estimate, in an adaptive manner and at each time instance, the values of $\boldsymbol{w}$ and $\boldsymbol{v}$ so that the "disagreement" between d_n and $\boldsymbol{D}_n(\boldsymbol{w},\boldsymbol{v})$, measured via a user-defined loss function, is minimized. However, note that both $\boldsymbol{w}$, $\boldsymbol{v}$ live in the complex RKHS $\mathbb{H}$.

On the other hand, for the complexification technique, the $\mathbb{C}$ linear estimation function $\boldsymbol{D}_n(\boldsymbol{w}) = \left\langle \hat{\boldsymbol{\Phi}}(\boldsymbol{z}_n), \boldsymbol{w}\right\rangle_{\mathbb{H}}$ is employed, as this procedure implicitly adds a conjugate component to the adopted model.

E. Selecting the Loss Functions

The strategy for constructing loss functions for the online learning problem, which is employed in this work, contains three steps.

1) Choose any convex continuous loss function $l : \mathbb{C} \to \mathbb{R}$. The function l needs not be differentiable. The only requirement is for its Wirtinger subgradients to be known in analytic form.
2) Form an ϵ-insensitive version of l following the rule:
$$l_\epsilon(z) := \max\{0, l(z) - \epsilon\}, \text{ for all } z \in \mathbb{C}, \tag{17}$$
where $\epsilon \geq 0$ takes a predetermined value.
3) Given a linear or widely linear kernel-based estimation function $D_n(\boldsymbol{w},\boldsymbol{v})$ and a given pair of training data $(\boldsymbol{z}_n, d_n) \in \mathbb{C}^\nu \times \mathbb{C}$, define the loss function $\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) : \mathbb{H} \to \mathbb{R}$, as follows:
$$\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = l_\epsilon\left(y_n - D_n(\boldsymbol{w},\boldsymbol{v})\right), \tag{18}$$
for all $\boldsymbol{w},\boldsymbol{v} \in \mathbb{H}$.

A few comments are in order on the reason behind the introduction of the ϵ-insensitive version of l. The function l_ϵ aims to robustify the online learning task against inaccuracies, noise, and outliers. Given a learning task, the designer chooses, usually, a convex loss function, l, whose minimizers are envisaged as the solutions to the learning task at hand. However, it is often the case that the choice of l does not fit accurately the underlying model and noise profile, due to various measurement inaccuracies and the presence of outliers. To tackle such an unpleasant situation, we adopt here the strategy of enlarging the set of minimizers of l, without changing radically the shape of l, since we would like to adhere to our original intuition on the choice of l. This is achieved by the introduction of l_ϵ. To see this, in a more rigorous way, assume that $\epsilon \geq 0$ and l are chosen such that $\min_{z\in\mathbb{C}} l(z) \leq \epsilon$. Notice that such an assumption is not tight, since, in most cases, the loss l is chosen to be a nonnegative function, with $l(0) = 0$. Then, it is easy to verify that (i) $\arg\min_{z\in\mathbb{C}} l_\epsilon(z) = \{z \in \mathbb{C} : l(z) \leq \epsilon\}$, and (ii) $\arg\min_{z\in\mathbb{C}} l(z) \subset \arg\min_{z\in\mathbb{C}} l_\epsilon(z)$. The ϵ-insensitive rationale is also in agreement to the robust statistics' framework [51].

To complete the presentation, we compute the Wirtinger subgradients of $\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v})$ for some popular loss functions l. First comes a popular example.

Lemma 2 (Quadratic ϵ-insensitive loss function)**.** *Choose the l_2 norm, $l_2(z) = |z|^2$, as the function l in (18). Then the Wirtinger's subdifferential of $\mathcal{L}_{\epsilon,n}$ is given by*

$$\partial_{w^*}\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = \begin{cases} \{-e_n^*\boldsymbol{\Phi}(\boldsymbol{z}_n)\}, & \textit{if } |e_n|^2 > \epsilon \\ \{\boldsymbol{0}\}, & \textit{if } |e_n|^2 < \epsilon \\ \mathrm{conv}\{\boldsymbol{0}, -e_n^*\boldsymbol{\Phi}(\boldsymbol{z}_n)\}, & \textit{if } |e_n|^2 = \epsilon \end{cases} \tag{19}$$

and

$$\partial_{v^*}\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = \begin{cases} \{-e_n^*\boldsymbol{\Phi}^*(\boldsymbol{z}_n)\}, & \textit{if } |e_n|^2 > \epsilon \\ \{\boldsymbol{0}\}, & \textit{if } |e_n|^2 < \epsilon \ , \\ \mathrm{conv}\{\boldsymbol{0}, -e_n^*\boldsymbol{\Phi}^*(\boldsymbol{z}_n)\}, & \textit{if } |e_n|^2 = \epsilon \end{cases} \tag{20}$$

where $e_n = d_n - D_n(\boldsymbol{w},\boldsymbol{v})$ and $\boldsymbol{\Phi}$ is the function used to map the input data to $\mathbb{H}$.

Proof: If we choose the l_2 norm as the function l in (18), then $\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = \max\{0, |y_n - D_n(\boldsymbol{w},\boldsymbol{v})|^2 - \epsilon\}$. Let $e_n(\boldsymbol{w},\boldsymbol{v}) = y_n - D_n(\boldsymbol{w},\boldsymbol{v})$ measure the error between the filter output d_n and the estimation function (in many cases $e_n(\boldsymbol{w},\boldsymbol{v})$ is simple denoted as e_n to save space). We compute the subdifferential case by case.

1) Consider the case of a $(\boldsymbol{w},\boldsymbol{v})$ such that $|e_n(\boldsymbol{w},\boldsymbol{v})|^2 > \epsilon$. Then $\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = |d_n - D_n(\boldsymbol{w},\boldsymbol{v})|^2 - \epsilon = l_2 \circ e_n(\boldsymbol{w},\boldsymbol{v}) - \epsilon$. It can be easily verified that, as $l_2(z) = |z|^2 = z^*z$, its conjugate Wirtinger's gradient is $\nabla_{z^*} l_2(z) = z$. Furthermore, applying rules (4) and (5) of the generalized Wirtinger's calculus [19] we obtain $\nabla_{\boldsymbol{w}^*} e_n(\boldsymbol{w},\boldsymbol{v}) = -\boldsymbol{\Phi}(\boldsymbol{z}_n)e_n^*(\boldsymbol{w},\boldsymbol{v})$ and $\nabla_{\boldsymbol{v}^*} e_n(\boldsymbol{w},\boldsymbol{v}) = -\boldsymbol{\Phi}^*(\boldsymbol{z}_n)e_n^*(\boldsymbol{w},\boldsymbol{v})$. The result follows from the chain rule property of Wirtinger's calculus.
2) Next, consider the case of a $(\boldsymbol{w},\boldsymbol{v})$ such that $|e_n(\boldsymbol{w},\boldsymbol{v})|^2 < \epsilon$. Then $\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = \boldsymbol{0}$ and the result is obvious.
3) If a $(\boldsymbol{w},\boldsymbol{v})$ is given, such that $|e_n(\boldsymbol{w},\boldsymbol{v})|^2 = \epsilon$, then $\mathcal{L}_{\epsilon,n}$ is not differentiable at $(\boldsymbol{w},\boldsymbol{v})$. The result follows from Lemma 1, as for any such point we can find a sequence of points $\{(\boldsymbol{w}_m,\boldsymbol{v}_m)\}_{m\in\mathbb{N}}$ converging to $(\boldsymbol{w},\boldsymbol{v})$, where $(\boldsymbol{w}_m,\boldsymbol{v}_m)$ is chosen so that $|e_n(\boldsymbol{w}_m,\boldsymbol{v}_m)| \neq \epsilon$ (i.e., $\mathcal{L}_{\epsilon,n}$ is differentiable at $(\boldsymbol{w}_m,\boldsymbol{v}_m)$ for all $m \in \mathbb{N}$).

■

It is, by now, well-documented, [63], that the l_2 norm is not the best choice for a loss function in environments where the noise is non-Gaussian. In order to build a general scheme, which can accommodate any kind of noise and outlier profiles, the present section gives freedom to the designer to choose any convex objective function l. To support this approach, we provide a couple of examples, which depart from the classical l_2 norm strategy. The next examples are motivated by the recently overwhelming popularity of the l_1 norm as a robustness and sparsity-promoting loss function, [63].

Lemma 3 (l_1 ϵ-insensitive complex loss function)**.** *Choose the complex l_1 norm, $l_1(z) = |z|$, as the function l in (18). Then*

$\partial_{w^*}\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = \begin{cases} \left\{-\left(\frac{\operatorname{sign}(\Re(e_n))}{2}+\frac{\operatorname{sign}(\Im(e_n))}{2i}\right)\boldsymbol{\Phi}(\boldsymbol{z}_n)\right\}, & \text{if } \lvert\Re(e_n)\rvert+\lvert\Im(e_n)\rvert>\epsilon \\ \{\boldsymbol{0}\}, & \text{if } \lvert\Re(e_n)\rvert+\lvert\Im(e_n)\rvert<\epsilon \\ \operatorname{conv}\left\{\boldsymbol{0},-\left(\frac{\operatorname{sign}(\Re(e_n))}{2}+\frac{\operatorname{sign}(\Im(e_n))}{2i}\right)\boldsymbol{\Phi}(\boldsymbol{z}_n)\right\}, & \text{if } \lvert\Re(e_n)\rvert+\lvert\Im(e_n)\rvert=\epsilon \end{cases}$
$\partial_{v^*}\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = \begin{cases} \left\{-\left(\frac{\operatorname{sign}(\Re(e_n))}{2}+\frac{\operatorname{sign}(\Im(e_n))}{2i}\right)\boldsymbol{\Phi}^*(\boldsymbol{z}_n)\right\}, & \text{if } \lvert\Re(e_n)\rvert+\lvert\Im(e_n)\rvert>\epsilon \\ \{\boldsymbol{0}\}, & \text{if } \lvert\Re(e_n)\rvert+\lvert\Im(e_n)\rvert<\epsilon \\ \operatorname{conv}\left\{\boldsymbol{0},-\left(\frac{\operatorname{sign}(\Re(e_n))}{2}+\frac{\operatorname{sign}(\Im(e_n))}{2i}\right)\boldsymbol{\Phi}^*(\boldsymbol{z}_n)\right\}, & \text{if } \lvert\Re(e_n)\rvert+\lvert\Im(e_n)\rvert=\epsilon \end{cases}$

TABLE I

THE SUBDIFFERENTIAL OF THE l_1 "REAL" LOSS FUNCTION, WHERE $e_n = d_n - D_n(\boldsymbol{w},\boldsymbol{v})$ AND $\boldsymbol{\Phi}$ IS THE FUNCTION USED TO MAP THE INPUT DATA TO $\mathbb{H}$.

$\partial_{w^*}\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = \begin{cases} \{\boldsymbol{0}\}, & \text{if } 0\le\lvert e_n\rvert<\epsilon \\ \operatorname{conv}\left\{\boldsymbol{0},-\frac{1}{2}e_n^*\boldsymbol{\Phi}(\boldsymbol{z}_n)\right\} & \text{if } \lvert e_n\rvert=\epsilon \\ \left\{-\frac{1}{2}e_n^*\boldsymbol{\Phi}(\boldsymbol{z}_n)\right\}, & \text{if } \epsilon<\lvert e_n\rvert<\sigma \\ \operatorname{conv}\left\{-\frac{\sigma e_n^*}{2\lvert e_n\rvert}\boldsymbol{\Phi}(\boldsymbol{z}_n),-\frac{1}{2}e_n^*\boldsymbol{\Phi}(\boldsymbol{z}_n)\right\}, & \text{if } \lvert e_n\rvert=\sigma \\ \left\{-\frac{\sigma e_n^*}{2\lvert e_n\rvert}\boldsymbol{\Phi}(\boldsymbol{z}_n)\right\}, & \text{if } \lvert e_n\rvert>\sigma \end{cases}$
$\partial_{v^*}\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = \begin{cases} \{\boldsymbol{0}\}, & \text{if } 0\le\lvert e_n\rvert<\epsilon \\ \operatorname{conv}\left\{\boldsymbol{0},-\frac{1}{2}e_n^*\boldsymbol{\Phi}^*(\boldsymbol{z}_n)\right\} & \text{if } \lvert e_n\rvert=\epsilon \\ \left\{-\frac{1}{2}e_n^*\boldsymbol{\Phi}^*(\boldsymbol{z}_n)\right\}, & \text{if } \epsilon<\lvert e_n\rvert<\sigma \\ \operatorname{conv}\left\{-\frac{\sigma e_n^*}{2\lvert e_n\rvert}\boldsymbol{\Phi}^*(\boldsymbol{z}_n),-\frac{1}{2}e_n^*\boldsymbol{\Phi}^*(\boldsymbol{z}_n)\right\}, & \text{if } \lvert e_n\rvert=\sigma \\ \left\{-\frac{\sigma e_n^*}{2\lvert e_n\rvert}\boldsymbol{\Phi}^*(\boldsymbol{z}_n)\right\}, & \text{if } \lvert e_n\rvert>\sigma \end{cases}$

TABLE II

THE SUBDIFFERENTIAL OF THE HUBER LOSS FUNCTION, WHERE $e_n = d_n - D_n(\boldsymbol{w},\boldsymbol{v})$ AND $\boldsymbol{\Phi}$ IS THE FUNCTION USED TO MAP THE INPUT DATA TO $\mathbb{H}$.

the Wirtinger's subdifferential of $\mathcal{L}_{\epsilon,n}$ is given by

$$\partial_{w^*}\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = \begin{cases} \left\{-\frac{e_n^*}{2\lvert e_n\rvert}\boldsymbol{\Phi}(\boldsymbol{z}_n)\right\}, & \textit{if } \lvert e_n\rvert>\epsilon \\ \{\boldsymbol{0}\}, & \textit{if } \lvert e_n\rvert<\epsilon \\ \operatorname{conv}\left\{\boldsymbol{0},-\frac{e_n^*}{2\lvert e_n\rvert}\boldsymbol{\Phi}(\boldsymbol{z}_n)\right\}, & \textit{if } \lvert e_n\rvert=\epsilon \end{cases} \tag{21}$$

and

$$\partial_{v^*}\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = \begin{cases} \left\{-\frac{e_n^*}{2\lvert e_n\rvert}\boldsymbol{\Phi}^*(\boldsymbol{z}_n)\right\}, & \textit{if } \lvert e_n\rvert>\epsilon \\ \{\boldsymbol{0}\}, & \textit{if } \lvert e_n\rvert<\epsilon \\ \operatorname{conv}\left\{\boldsymbol{0},-\frac{e_n^*}{2\lvert e_n\rvert}\boldsymbol{\Phi}^*(\boldsymbol{z}_n)\right\}, & \textit{if } \lvert e_n\rvert=\epsilon \end{cases} \tag{22}$$

where $e_n = d_n - D_n(\boldsymbol{w},\boldsymbol{v})$ and $\boldsymbol{\Phi}$ is the function used to map the input data to $\mathbb{H}$.

Proof: For the first case, observe that if a $(\boldsymbol{w},\boldsymbol{v})$ is given such that $\lvert e_n(\boldsymbol{w},\boldsymbol{v})\rvert>\epsilon$, then $\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = \lvert d_n - D_n(\boldsymbol{w},\boldsymbol{v})\rvert - \epsilon = l_1 \circ e_n(\boldsymbol{w},\boldsymbol{v}) - \epsilon$. As $l_1(z) = \lvert z\rvert = \sqrt{z^*z}$, its Wirtinger's gradients are $\nabla_z l_1(z) = \frac{1}{2\lvert z\rvert}z$ and $\nabla_{z^*} l_1(z) = \frac{1}{2\lvert z\rvert}z^*$. The result follows from the chain rule of the generalized Wirtinger's calculus and Lemma 1. For the other two cases, we work as in Lemma 2. ■

Lemma 4 (l_1 ϵ-insensitive "real" loss function)**.** *Choose the "real" l_1^r norm, $l_1^r(z) = \lvert\Re(z)\rvert+\lvert\Im(z)\rvert$, as the function l in (18). Then the Wirtinger's subdifferential of $\mathcal{L}_{\epsilon,n}$ is given in table I.*

Proof: For the first case, observe that if a $(\boldsymbol{w},\boldsymbol{v})$ is given such that $\lvert\Re(z)\rvert+\lvert\Im(z)\rvert>\epsilon$, then $\mathcal{L}_{\epsilon,n}(\boldsymbol{w},\boldsymbol{v}) = \lvert\Re(d_n - D_n(\boldsymbol{w},\boldsymbol{v}))\rvert + \lvert\Im(d_n - D_n(\boldsymbol{w},\boldsymbol{v}))\rvert - \epsilon = l_1^r \circ e_n(\boldsymbol{w},\boldsymbol{v}) - \epsilon$. As

$$\begin{aligned} l_1^r(z) =& \lvert\Re(z)\rvert+\lvert\Im(z)\rvert = \operatorname{sign}(\Re(z))\Re(z)+\operatorname{sign}(\Im(z))\Im(z) \\ =& \operatorname{sign}(\Re(z))\frac{z+z^*}{2}+\operatorname{sign}(\Im(z))\frac{z-z^*}{2i}, \end{aligned}$$

its Wirtinger's gradients are

$$\begin{aligned} \nabla_z l_1^r(z) &= \left(\frac{\operatorname{sign}(\Re(z))}{2}+\frac{\operatorname{sign}(\Im(z))}{2i}\right), \\ \nabla_{z^*} l_1^r(z) &= \left(\frac{\operatorname{sign}(\Re(z))}{2}-\frac{\operatorname{sign}(\Im(z))}{2i}\right). \end{aligned}$$

The result follows from the chain rule of the generalized Wirtinger's calculus and Lemma 1. For the other two cases, we work as in Lemma 2. ■

Lemma 5 (ϵ-insensitive Huber loss function)**.** *Choose the Huber loss,*

$$l_h(z) = \begin{cases} \frac{1}{2}\lvert z\rvert^2 & \textit{if } \lvert z\rvert<\sigma \\ \sigma(\lvert z\rvert-\frac{\sigma}{2}) & \textit{if } \lvert z\rvert\ge\sigma \end{cases},$$

as the function l in (18), for some $\sigma>\epsilon$. Then the Wirtinger's subdifferential of $\mathcal{L}_{\epsilon,n}$ is given in table II.

Proof: We work similarly to lemmas 2 and 3. ■

IV. Complex Kernel Adaptive Projected Subgradient Method (CKAPSM)

The algorithmic scheme, which will be developed in this section, is based on the *Adaptive Projected Subgradient Method* (APSM) [44]–[46], [50]. This has been motivated by projection-based adaptive algorithms, e.g., the Normalized LMS and the Affine Projection Algorithm (APA) [64]. The APSM has been successfully applied to a variety of online learning problems, [9], [12], [50] and has been very recently generalized to tackle constrained optimization tasks in general Hilbert spaces [46]. In order to speed up convergence, APSM concurrently processes multiple data points at every time instant. Given a user defined positive integer q, for every time instant n, APSM considers a sliding window on the time axis of size (at most) q: $\mathcal{J}_n := \overline{\max\{0, n-q+1\}, n}$. Each $k \in \mathcal{J}_n$ associates to the loss function $\mathcal{L}_{\epsilon,k}$, which, in turn, is determined by the k-th training data point (e.g. $(\boldsymbol{x}_k, y_k)$). The set $\mathcal{J}_n$ indicates the loss functions that are going to be concurrently processed at the time instant n. For a real data sequence $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, APSM then employs the update mechanism:

$$\boldsymbol{w}_{n+1} = \boldsymbol{w}_n - \mu_n \sum_{k\in\mathcal{I}_n} \omega_k^{(n)} \frac{\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n)}{\|\nabla^s \mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n)\|^2} \nabla^s \mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n), \tag{23}$$

where $\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n)$ is the loss function between y_k and the estimation function $D_k(\boldsymbol{w})$ (which is chosen in a manner similar to section III-E, i.e., $D_k(\boldsymbol{w}) = \langle \boldsymbol{\Phi}(\boldsymbol{x}_k), \boldsymbol{w}\rangle)$, $\nabla^s \mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n)$ is a subgradient of $\mathcal{L}_{\epsilon,k}$ at $\boldsymbol{w}_n$, μ_n is an extrapolation parameter, $\omega_k^{(n)}$ are weights chosen such that $\sum_{k\in\mathcal{I}_n}\omega_k^{(n)} = 1$ and $\mathcal{I}_n$ is an appropriately chosen index set[4] ($\mathcal{I}_n \subset \mathcal{J}_n$). The interested reader may dig deeper on this algorithmic scheme by referring to [12], [60]. In this section we develop a similar machinery for complex data sequences using the newly introduced notion of Wirtinger's subgradients.

A. The CKAPSM Algorithm

We develop the algorithm for a general widely linear estimation function $D_n(\boldsymbol{w}, \boldsymbol{v})$, as this have been defined in section III-D. For a $\mathbb{C}$-linear estimation function $D_n(\boldsymbol{w})$, simply ignore the $\boldsymbol{v}_n$ term.

1) Choose a non-negative $\epsilon \geq 0$ and a positive number q, which will stand for the number of loss functions that are concurrently processed at every time instant n. Furthermore, fix arbitrary $\boldsymbol{w}_0$ and $\boldsymbol{v}_0$ as a starting point for the algorithm (typically $\boldsymbol{w}_0 = \boldsymbol{v}_0 = \boldsymbol{0}$).
2) Given any time instant $n \in \mathbb{N}$, define the sliding window on the time axis, of size at most q: $\mathcal{J}_n := \overline{\max\{0, n-q+1\}, n}$. The user-defined parameter q determines the number of training points (and associated loss functions) that are concurrently processed at each time instant n.
3) Given the current estimates $\boldsymbol{w}_n$, $\boldsymbol{v}_n$, choose any Wirtinger subgradient $\nabla^s_{\boldsymbol{w}^*}\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n, \boldsymbol{v}_n) \in \partial_{\boldsymbol{w}^*}\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n, \boldsymbol{v}_n)$ and $\nabla^s_{\boldsymbol{v}^*}\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n, \boldsymbol{v}_n) \in \partial_{\boldsymbol{v}^*}\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n, \boldsymbol{v}_n)$. Thus, a collection of Wirtinger subgradients is formed:
$$\{\boldsymbol{W}_{\epsilon,k,n} = \nabla^s_{\boldsymbol{w}^*}\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n, \boldsymbol{v}_n)\}_{k\in\mathcal{J}_n} \text{ and }$$
$$\{\boldsymbol{V}_{\epsilon,k,n} = \nabla^s_{\boldsymbol{v}^*}\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n, \boldsymbol{v}_n)\}_{k\in\mathcal{J}_n}.$$
4) Define the active index set $\mathcal{I}_n := \{k \in \mathcal{J}_n : \nabla^s_{\boldsymbol{w}^*}\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n, \boldsymbol{v}_n) \neq \boldsymbol{0}, \text{ or } \nabla^s_{\boldsymbol{v}^*}\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n, \boldsymbol{v}_n) \neq \boldsymbol{0}\}$.
5) If $\mathcal{I}_n \neq \emptyset$, define a set of weights $\{\omega_k^{(n)}\}_{k\in\mathcal{I}_n} \subset (0, 1]$, such that $\sum_{k\in\mathcal{I}_n}\omega_k^{(n)} = 1$. Each parameter $\omega_k^{(n)}$ assigns a weight to the contribution of $\mathcal{L}_{\epsilon,k}$ to the following concurrent scheme. Typically, we set $\omega_k^{(n)} = 1/\operatorname{card}\mathcal{I}_n$, for all $k \in \mathcal{I}_n$ (card stands for the cardinality of a set).
6) Calculate the next estimate of $\boldsymbol{w}, \boldsymbol{v}$ using the following recurrent scheme:
$$\begin{aligned} \boldsymbol{w}_{n+1} &= \boldsymbol{w}_n - \mu_n \textstyle\sum_{k\in\mathcal{I}_n} \omega_k^{(n)} \frac{\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n, \boldsymbol{v}_n)}{2\mathcal{U}_{\epsilon,k,n}} \boldsymbol{W}_{\epsilon,k,n}, \\ \boldsymbol{v}_{n+1} &= \boldsymbol{v}_n - \mu_n \textstyle\sum_{k\in\mathcal{I}_n} \omega_k^{(n)} \frac{\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n, \boldsymbol{v}_n)}{2\mathcal{U}_{\epsilon,k,n}} \boldsymbol{V}_{\epsilon,k,n}. \end{aligned} \tag{24}$$
where $\mathcal{U}_{\epsilon,k,n} = \|\boldsymbol{W}_{\epsilon,k,n}\|^2 + \|\boldsymbol{V}_{\epsilon,k,n}\|^2$.

Equations (24) have been developed directly from the traditional recurrent scheme of the real case, i.e., relation (23), by substituting the real subgradients with the newly introduced Wirtinger subgradients via the rationale developed in the proof of Lemma 1. Loosely speaking, one can replace the real (partial) subgradient $\nabla^s_{\boldsymbol{w}}\mathcal{L}_{\epsilon,k}(\boldsymbol{w}, \boldsymbol{v})$, that is obtained if $\mathcal{L}_{\epsilon,k}$ is considered as a function defined on $\mathcal{H}^2 \times \mathcal{H}^2$, with $2\nabla^s_{\boldsymbol{w}^*}\mathcal{L}_{\epsilon,k}(\boldsymbol{w}, \boldsymbol{v})$. In the case where $\mathcal{I}_n = \emptyset$, the summation term over $\emptyset$ will be set equal to 0. The extrapolation parameter μ_n lies within the interval $(0, 2\mathcal{M}_n,)$, where $\mathcal{M}_n$ is given in (26). Notice that, due to convexity of $\|\cdot\|^2$, it is easy to verify that $\mathcal{M}_n \geq 1$. For larger values of the user-defined parameter q, $\mathcal{M}_n$ typically grows far from 1. We typically choose μ_n as

$$\mu_n = \begin{cases} 2\mathcal{M}_n - 0.05 & \text{if } \mathcal{M}_n \leq 2 \\ \min(\mathcal{M}_n, \mu_0) & \text{otherwise,} \end{cases} \tag{25}$$

where μ_0 is a user defined parameter (typically between 1 and 4).

Building upon the aforementioned algorithmic scheme, two realizations of the *Complex Kernel Adaptive Projected Subgradient Method* (CKAPSM) have been developed. The first one, which is denoted as CKAPSM, adopts the complexification trick to map the data to a complex RKHS using any real kernel. Moreover, the $\mathbb{C}$-linear function $D_n(\boldsymbol{w}) = \langle \hat{\boldsymbol{\Phi}}(\boldsymbol{z}_n), \boldsymbol{w}\rangle_{\mathbb{H}}$ is employed to estimate the filter's output. The second algorithm, which is denoted as *Augmented Complex Kernel Adaptive Projected Subgradient Method* (ACKAPSM), adopts the pure complex kernelization trick to map the data to a complex RKHS using the complex gaussian kernel. In the latter case, to estimate the filter's output, the widely linear (augmented) function $D_n(\boldsymbol{w}, \boldsymbol{v}) = \langle \boldsymbol{\Phi}(\boldsymbol{z}_n), \boldsymbol{w}\rangle_{\mathbb{H}} + \langle \boldsymbol{\Phi}^*(\boldsymbol{z}_n), \boldsymbol{v}\rangle_{\mathbb{H}}$ is used.

B. Sparsification

Any typical kernel-based adaptive filtering algorithm, suffers from increasing memory and computational requirements, as a growing number of training points is involved in the solution. This is verified by the celebrated Representer theorem

[4] It will be defined later in section IV-A.

$$\mathcal{M}_n = \begin{cases} \dfrac{\sum_{k\in\mathcal{I}_n} \omega_k^{(n)} \frac{\mathcal{L}_{\epsilon,k}^2(\boldsymbol{w}_n,\boldsymbol{v}_n)}{4\mathcal{U}_{\epsilon,k,n}}}{\left\|\sum_{k\in\mathcal{I}_n} \omega_k^{(n)} \frac{\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n,\boldsymbol{v}_n)}{2\mathcal{U}_{\epsilon,k,n}} \boldsymbol{W}_{\epsilon,k,n}\right\|^2 + \left\|\sum_{k\in\mathcal{I}_n} \omega_k^{(n)} \frac{\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n,\boldsymbol{v}_n)}{2\mathcal{U}_{\epsilon,k,n}} \boldsymbol{V}_{\epsilon,k,n}\right\|^2}, \\ \quad \text{if } \sum_{k\in\mathcal{I}_n} \omega_k^{(n)} \frac{\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n,\boldsymbol{v}_n)}{2\mathcal{U}_{\epsilon,k,n}} \boldsymbol{W}_{\epsilon,k,n} \neq 0, \\ \quad \text{or } \sum_{k\in\mathcal{I}_n} \omega_k^{(n)} \frac{\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n,\boldsymbol{v}_n)}{2\mathcal{U}_{\epsilon,k,n}} \boldsymbol{V}_{\epsilon,k,n} \neq 0, \\ 1, \text{otherwise.} \end{cases} \tag{26}$$

[65], which states that the solution of such a task lies in the finite dimensional subspace of the RKHS, which is spanned by the mapped training (real) input data points, i.e.,

$$\boldsymbol{w}_n = \sum_{k=0}^{n-1} a_k \boldsymbol{\Phi}(\boldsymbol{x}_k).$$

In our case, where complex input training data are considered, this is equivalent with

$$\boldsymbol{w}_n = \sum_{k=0}^{n} a_k \boldsymbol{\Phi}(\boldsymbol{z}_k), \boldsymbol{v}_n = \sum_{k=0}^{n} a_k \boldsymbol{\Phi}^*(\boldsymbol{z}_k), \tag{27}$$

if the widely linear estimation rationale is adopted, as it can be easily verified by the gradients of the loss functions considered in section III-E (where $\boldsymbol{\Phi}$ is the function used to map the input data to $\mathbb{H}$, for $k = 0, \dots, n$, and $a_k \in \mathbb{C}$).

In this paper, to cope with this problem, we focus on the *projection onto closed l_2 balls* rationale, introduced in [12], [60]. In this context, we choose a positive parameter Δ and impose the l_2 closed ball on $\mathbb{H}$ with center $\mathbf{0}$ and radius Δ, i.e., $B[0, \Delta]$ on the optimization scheme. That is, we replace the recurrent step of the algorithm with

$$\boldsymbol{w}_{n+1} = P_{B[0,\Delta]}\left(\boldsymbol{w}_n - \mu_n \sum_{k\in\mathcal{I}_n} \omega_k^{(n)} \frac{\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n,\boldsymbol{v}_n)}{2\mathcal{U}_{\epsilon,k,n}} \boldsymbol{W}_{\epsilon,k,n}\right),$$
$$\boldsymbol{v}_{n+1} = P_{B[0,\Delta]}\left(\boldsymbol{v}_n - \mu_n \sum_{k\in\mathcal{I}_n} \omega_k^{(n)} \frac{\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n,\boldsymbol{v}_n)}{2\mathcal{U}_{\epsilon,k,n}} \boldsymbol{V}_{\epsilon,k,n}\right),$$

where $P_{B[0,\Delta]}$ is the metric projection mapping onto the closed ball $B[0, \Delta]$, which is given by

$$P_{B[0,\Delta]}(\boldsymbol{f}) = \begin{cases} \boldsymbol{f}, & \text{if } \|\boldsymbol{f}\| \le \Delta, \\ \frac{\Delta}{\|\boldsymbol{f}\|}\boldsymbol{f}, & \text{if } \|\boldsymbol{f}\| > \Delta. \end{cases}$$

Let us, now, turn our attention on the weights update stage (i.e., equations (24)) and discuss on how they are practically implemented on a machine, as both $\boldsymbol{w}$ and $\boldsymbol{v}$ are elements of an infinite dimensional RKHS. After the receipt of the $n-$th sample, both $\boldsymbol{w}$ and $\boldsymbol{v}$ have a finite representation in terms of $\boldsymbol{\Phi}(\boldsymbol{z}_k)$ and $\boldsymbol{\Phi}^*(\boldsymbol{z}_k)$ respectively, for $k = 0, \dots, n-1$, (see (27)). Thus, one needs to store into the machine's memory only the n coefficients, $a_0, a_1, \dots, a_{n-1}$, of the expansion. Let $\mathcal{A}_{n-1} = \{a_0, a_1, \dots, a_{n-1}\}$ be the set of the coefficients that has been stored at iteration $n-1$. Next, as the n-th sample has been received, equations (24) update at most $q-1$ of the coefficients in $\mathcal{A}_{n-1}$ (the ones that are inside the active set) and, possibly, compute and store the coefficient a_n (if this is inside the active set, otherwise a_n is set to 0). In particular, for every $k \in \overline{\max\{0, n-q+1\}, n}$, which is inside the active set $\mathcal{I}_n$, we employ the update equation:

$$a_k^{new} = a_k^{old} - \mu_n \omega_k^{(n)} \frac{\mathcal{L}_{\epsilon,k}(\boldsymbol{w}_n, \boldsymbol{v}_n)}{2\mathcal{U}_{\epsilon,k,n}} C_k,$$

where $a_n^{(old)} = 0$ and C_k is the coefficient of $\Phi(z_k)$ in the respective gradients $\boldsymbol{W}_{\epsilon,k,n}$ and $\boldsymbol{V}_{\epsilon,k,n}$[5]. Consequently, the norms of $\boldsymbol{w}_{n+1}$ and $\boldsymbol{v}_{n+1}$ are computed[6] and if they are found larger than Δ, each one of the $n+1$ coefficients of $\mathcal{A}_n$ is shrunk by the factor $\frac{\Delta}{\|\boldsymbol{w}_{n+1}\|}$ and/or $\frac{\Delta}{\|\boldsymbol{v}_{n+1}\|}$ respectively. If, after multiple shrinks, some of the coefficients become really small (i.e., smaller than a predefined threshold ϵ_Δ), they are thrown out of the stored memory.

V. Experiments

The performance of CKAPSM and ACKAPSM has been tested in the context of: (a) a non-linear channel identification task, (b) a non-linear channel equalization task and (c) an equalization task of a QPSK modulation scheme. In all the experiments, the parameters of the tested algorithmic schemes were tuned for the best performance (i.e., to achieve the smallest possible MSE). The code for the experiments can be found at http://users.sch.gr/pbouboulis/kernels.html.

A. Channel Identification

We consider the non-linear channel presented in [28], which consists of a linear filter:

$$t_n = \sum_{k=1}^{5} h_k \cdot s_{n-k+1},$$

where

$$h_k = 0.432\left(1 + \cos\frac{2\pi(k-3)}{5} - \left(1 + \cos\frac{2\pi(k-3)}{10}\right)i\right),$$

for $k = 1, \dots, 5$, and the nonlinear component $x_n = t_n + (0.15 - 0.1i)t_n^2$. At the receiver end of the channel, the signal

[5] For example, if the l_2 norm has been chosen, $C_k = -e_k^*$, as equation (19) suggests.

[6] The direct computation of the norm $\|\boldsymbol{w}_{n+1}\|$ is a computationally demanding step. However, as in the present context only q elements of the expansion of $\boldsymbol{w}$ are updated, we can compute the norm $\|\boldsymbol{w}_{n+1}\|$ using a recurrent scheme. For example, if $q = 1$, then $\boldsymbol{w}_{n+1} = \boldsymbol{w}_n + a_{n+1}\boldsymbol{\Phi}(\boldsymbol{z}_{n+1})$. Then $\|\boldsymbol{w}_{n+1}\|^2 = \langle \boldsymbol{w}_{n+1}, \boldsymbol{w}_{n+1}\rangle_{\mathbb{H}} = \|\boldsymbol{w}_n\|^2 + a_{n+1}\sum_{k=0}^{n} a_k^* \langle \boldsymbol{\Phi}(\boldsymbol{z}_k), \boldsymbol{\Phi}(\boldsymbol{z}_{n+1})\rangle_{\mathbb{H}} + a_{n+1}^* \sum_{k=0}^{n} a_k \langle \boldsymbol{\Phi}(\boldsymbol{z}_{n+1}), \boldsymbol{\Phi}(\boldsymbol{z}_k)\rangle_{\mathbb{H}} + \langle \boldsymbol{\Phi}(\boldsymbol{z}_{n+1}), \boldsymbol{\Phi}(\boldsymbol{z}_{n+1})\rangle_{\mathbb{H}}$.

is corrupted by white noise and then observed as r_n[7]. The input signal that was fed to the channel had the form

$$s_n = \left(\sqrt{1-\rho^2}X_n + i\rho Y_n\right), \tag{28}$$

where X_n and Y_n are zero-mean random variables. This input is circular for $\rho = \sqrt{2}/2$ and highly non-circular if ρ approaches 0 or 1 [28]. The aim of the channel identification task is to construct a non-linear filter that acts on the input s_n and reproduces the output x_n as close as possible. To this end, we apply CKAPSM and ACKAPSM to the set of samples

$$((s_n, s_{n-1}, \ldots, s_{n-L+1}), r_n),$$

where $L > 0$ is the filter length. To measure the closeness of fit between the original non-linear channel and the estimated filter, we compute the mean square error between the estimated filter's output, i.e., d_n, and x_n.

We tested CKAPSM and ACKAPSM using various input random variables (e.g., gaussian, uniform) as well as some popular noise models (e.g., gaussian, uniform, student, impulse) and different types of loss functions. Their performance is compared with the recently developed NCKLMS [19] and ANCKLMS [62], which have been found to perform significantly better [19] than other non-linear complex adaptive algorithmic schemes, such as Multi Layer Perceptrons (MLPs) [28] and Complex non-linear Gradient Descend (CNGD) [27]. In all of the performed tests (and especially in the non-circular case), CKAPSM and ACKAPSM considerably outperform the other two algorithms in terms of convergence speed and steady state mean square error. Figures 1, 3, 4, show the mean learning curves over 300 different sets of 10000 samples for each case.

In order to study the tracking performance of the proposed schemes in a time-adaptive setting, the case of a non-linear channel that undergoes a sudden significant change is considered in Figure 2. This is a typical scenario used in the context of adaptive filtering. After receiving sample $n = 5000$, the coefficients of the nonlinear filter become:

$$h_1 = 0.5 - 0.5i, h_2 = 0.1i - 0.2, h_3 = 0.6 - 0.3i,$$
$$h_4 = -0.5, h_5 = -0.8 + 1i,$$

and $x_n = t_n + (-0.1 + 0.08i)t_n^2$. Recall that, while CKLMS keeps the information of the first channel throughout the training phase, as the coefficients associated with the first filter remain in the associated expansion, CKAPSM is able to "forget" the information provided by the original channel via the shrinking process, which has been described in section IV-B. The novelty criterion sparsification mechanism was used for the NCKLMS and ANCKLMS algorithms with parameters $\delta_1 = 0.15$ and $\delta_2 = 0.2$. The radius of the closed ball for the CKAPSM and ACKAPSM sparsification technique was set to $\Delta = 10$.

The values of the parameters used in the algorithms are: $\sigma = 5$ (for both the real gaussian kernel and the complex gaussian kernel), $q = 5$ or $q = 20$ (this is shown in each figure), $\epsilon = 10^{-9}$ and $\mu_0 = 4$.

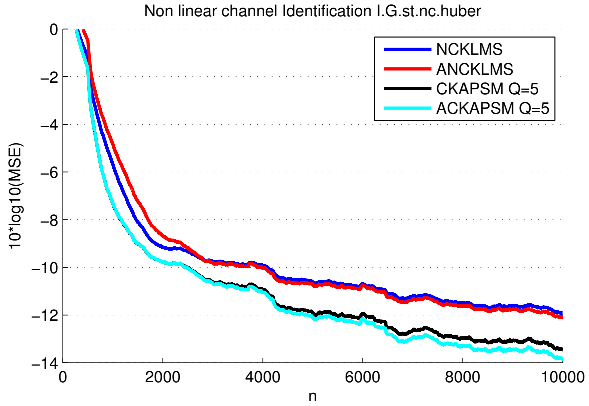


Fig. 3. Learning curves for NCKLMS ($\mu = 1$), ANCKLMS, ($\mu = 1/4$), CKAPSM and ACKAPSM (filter length $L = 5$) for the nonlinear channel identification problem with gaussian input and heavy-tailed student noise ($\nu = 3$)) at 20dB, for the non-circular input case ($\rho = 0.1$). In the realization of the CKAPSM and ACKAPSM the Huber loss function was employed.

The reason behind the improved performance of the APSM variants, over the Normalized LMS ones [19], [62], is due to the form of the iterations given in (23) and (24). In the NLMS framework, one pair of training data is processed per time instant n, while the APSM gives us the freedom to concurrently process a set of training data, indicated by $\mathcal{I}_n$, $\forall n$. To each data pair, that belongs to $\mathcal{I}_n$, a weight $\omega_k^{(n)}$ is assigned to quantify the significance of the specific pair of data in the concurrency scheme. Such a weighted contribution of a set of training data helps APSM to achieve, in most of the cases, lower error floors compared to NLMS techniques. Even further, due to the multiplicity of data that are utilized in parallel, an extrapolation parameter μ_n is defined, which can significantly speed up convergence, since it obtains values ≥ 2. Recall that in the NLMS framework, [19], [62], the associated extrapolation parameter is upper bounded by 2. For a more detailed discussion on the superior performance of the NLMS variants versus the MLPs [28] and the CNGD [27], the interested reader is referred to [19], [62].

B. Channel Equalization

The non-linear channel considered in this case consists of a linear filter:

$$t_n = (-0.9 + 0.8i) \cdot s_n + (0.6 - 0.7i) \cdot s_{n-1}$$

and a memoryless nonlinearity

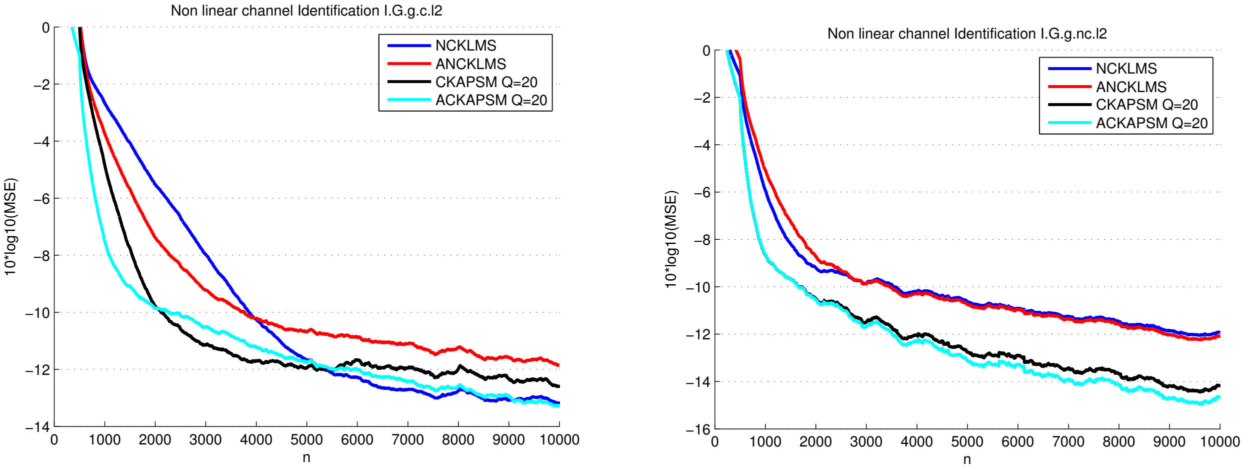
$$x_n = t_n + (0.1 + 0.15i) \cdot t_n^2 + (0.06 + 0.05i) \cdot t_n^3.$$

At the receiver end of the channel the signal is corrupted by white Gaussian noise and then observed as r_n. The input signal that was fed to the channels had the form

$$s_n = 0.5\left(\sqrt{1-\rho^2}X_n + i\rho Y_n\right), \tag{29}$$

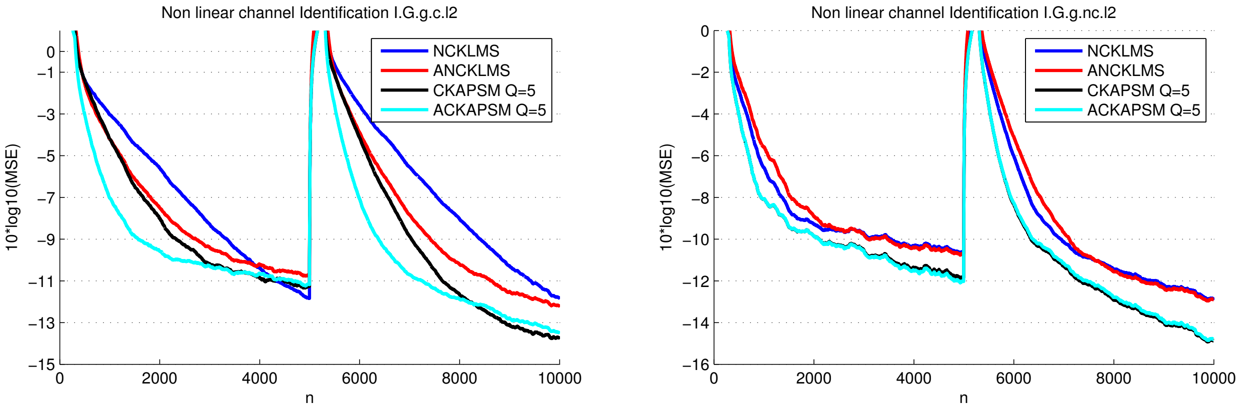
where $X(n)$ and $Y(n)$ are gaussian or uniform random variables. The level of the noise was set to $20dB$. The aim of a channel equalization task is to construct an inverse filter, which acts on the output r_n and reproduces the original input signal

[7] Hence, the input of the channel is s_n and the output r_n.



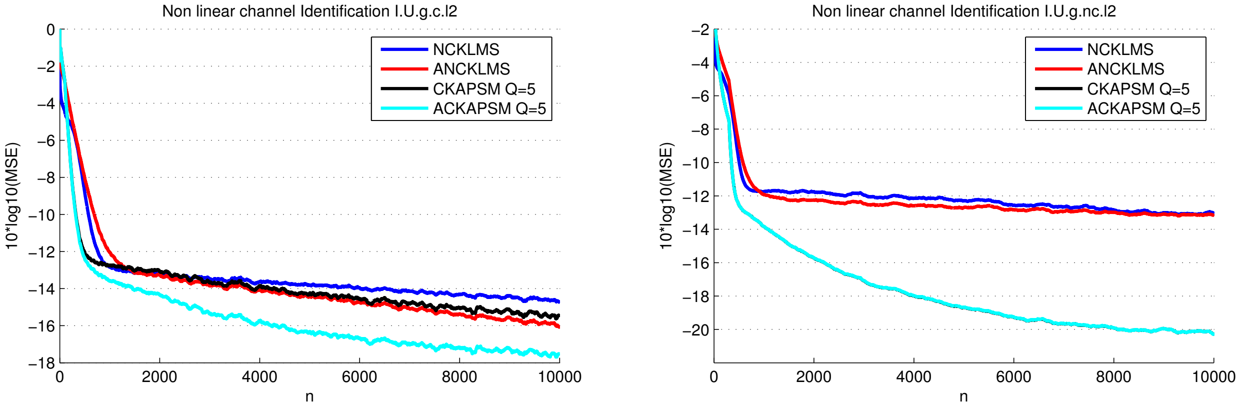
(a) (b)

Fig. 1. Learning curves for NCKLMS ($\mu = 1$), ANCKLMS, ($\mu = 1/4$), CKAPSM and ACKAPSM (filter length $L = 5$) for the nonlinear channel identification problem with gaussian input and gaussian noise at 20dB, for (a) the circular input case ($\rho = \sqrt{2}/2$) and (b) the non-circular input case ($\rho = 0.1$). In the realization of the CKAPSM and ACKAPSM the l_2 norm was employed.



(a) (b)

Fig. 2. Learning curves for NCKLMS ($\mu = 1$), ANCKLMS, ($\mu = 1/4$), CKAPSM and ACKAPSM (filter length $L = 5$) for the nonlinear two-channels identification problem with gaussian input and gaussian noise at 20dB, for (a) the circular input case ($\rho = \sqrt{2}/2$) and (b) the non-circular input case ($\rho = 0.1$). In the realization of the CKAPSM and ACKAPSM the l_2 norm was employed. After index $n = 5000$, both the linear and the non-linear component of the channel have been changed.



(a) (b)

Fig. 4. Learning curves for NCKLMS ($\mu = 1$), ANCKLMS, ($\mu = 1/4$), CKAPSM and ACKAPSM (filter length $L = 5$) for the nonlinear channel identification problem with uniform input and gaussian noise at 20dB, for (a) the circular input case ($\rho = \sqrt{2}/2$) and (b) the non-circular input case ($\rho = 0.1$). In the realization of the CKAPSM and ACKAPSM the l_2 loss function was employed.

as close as possible. To this end, we apply the algorithms to the set of samples

$$\left((r_{n+D}, r_{n+D-1}, \dots, r_{n+D-L+1}, s_n\right),$$

where $L > 0$ is the filter length and D the equalization time delay, which is present to, almost, any equalization set up. Experiments were conducted on 300 sets of 5000 samples of the input signal considering both the circular and the non-circular case. The results were compared to the NCKLMS and ANCKLMS, which have been shown to perform significantly better than other complex non-linear techniques such as MLPs and CNGD [19]. The values of the parameters used in the algorithms are: $\sigma = 5$ (for both the real gaussian kernel and the complex gaussian kernel), $q = 5$, $\epsilon = 10^{-8}$ and $\mu_0 = 4$. The sparsification mechanism adopted for this case was identical to the one employed in the channel identification paradigm. As it can be seen in figures 5, 6, CKAPSM and ACKAPSM converge more rapidly to the steady state mean square error, than NCKLMS and ANCKLMS (which have almost overlapping learning curves).

C. QPSK Equalization

In this case, we considered the non-linear channel which consists of the linear filter:

$$t_n = (-0.9 + 0.8i) \cdot s_n + (0.6 - 0.7i) \cdot s_{n-1}$$

and the memoryless nonlinearity

$$x_n = t_n + (0.1 + 0.15i) \cdot t_n^2.$$

At the receiver end of the channel the signal is corrupted by white Gaussian noise and then observed as r_n. The input signal that was fed to the channel consisted of the 4 QPSK symbols: $s_1 = 1 + i$, $s_2 = 1 - i$, $s_3 = -1 + i$ and $s_4 = -1 - i$. Both the circular and the non-circular input case were considered. For the first case, the 4 symbols are equiprobable, while in the later their probabilities for occurrence in the input sequence are $p_1 = 1/10$, $p_2 = 3/10$, $p_3 = 2/10$ and $p_4 = 4/10$, respectively (applications of non equiprobable symbol channels can be found in [66]). The objective in this task is to construct an inverse filter, which acts on the output r_n and reproduces the original input symbols as close as possible. Experiments were performed on 100 sets of 10000 input symbols. In the circular case, the NCKLMS and CKAPSM exhibit similar performance reaching a steady state mean SER of 0.0039 and 0.0034 respectively. For the non-circular case NCKLMs attained a steady state mean SER of 0.005, while the steady state mean SER of CKAPSM reached 0.0036 (i.e., a decrease of 28%). The values of the parameters used in the CKAPSM algorithm are: $\sigma = 5$, $q = 5$, $\epsilon = 10^{-8}$ and $\mu_0 = 4$. Figure 7 shows the SER versus SNR curves of those algorithms.

VI. Conclusions

A general tool for treating non-linear adaptive filtering problems of complex valued signal processing, on complex Reproducing Kernel Hilbert Spaces, has been developed. In this context, the complex input data are mapped into a complex RKHS, where the learning phase is taking place (based on the Adaptive Projected Subgradient Method), using both linear and widely linear estimation filters. The complex RKHS is implicitly defined through the choice of the kernel function. Both pure complex kernels (such as the complex gaussian one) as well as real kernels can be employed. Furthermore, any convex continuous function, whose subgradient is given in an analytic form, can be exploited to measure the loss between the output of the specific system and the desired response. To compute the subgradients of loss functions defined on complex RKHS, the notion of Wirtinger's subgradient has been introduced, and related subgradients have been derived for a number of popular cost functions. The effectiveness of the proposed framework has been demonstrated in several non-linear adaptive filtering tasks.

References

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

[2] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, 4th ed. Academic Press, Nov. 2008.

[3] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer Verlag, 1999.

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

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

[6] J. Kivinen, A. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2165–2176, 2004.

[7] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2275–2285, 2004.

[8] F. Pérez-Cruz and O. Bousquet, "Kernel methods and their potential use in signal processing," *IEEE Signal Processing Magazine*, pp. 57–65, May 2004.

[9] K. Slavakis, S. Theodoridis, and I. Yamada, "On line kernel-based classification using adaptive projection algorithms," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 2781–2796, 2008.

[10] J. Xu, A. Paiva, I. Park, and J. Principe, "A reproducing kernel Hilbert space framework for information theoretic learning," *IEEE Transactions on Signal Processing*, vol. 56, no. 12, pp. 5891–5902, 2008.

[11] W. Liu, P. Pokharel, and J. C. Principe, "The kernel Least-Mean-Square algorithm," *IEEE Trans. Signal Process.*, vol. 56, no. 2, pp. 543–554, 2008.

[12] K. Slavakis, S. Theodoridis, and I. Yamada, "Adaptive constrained learning in reproducing kernel Hilbert spaces: the robust beamforming case," *IEEE Transactions on Signal Processing*, vol. 57, no. 12, pp. 4744–4764, 2009.

[13] F. Dufrenois, J. Colliez, and D. Hamad, "Bounded influence support vector regression for robust single-model estimation," *IEEE Trans. Neural Networks*, vol. 20, no. 11, pp. 1689–1706, 2009.

[14] D. Musicant and A. Feinberg, "Active set support vector regression," *IEEE Trans. Neural Networks*, vol. 15, no. 2, pp. 268–275, 2004.

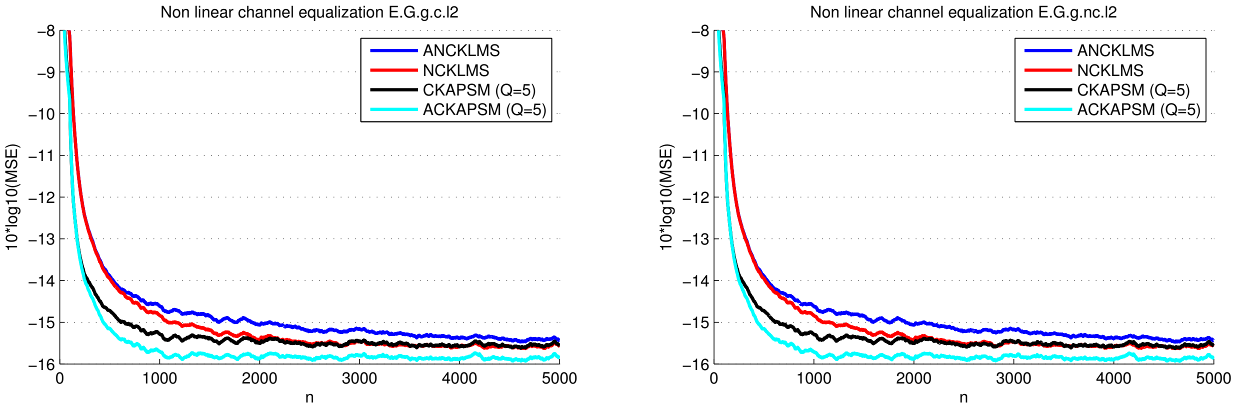
[15] M. Mavroforakis, M. Sdralis, and S. Theodoridis, "A geometric nearest point algorithm for the efficient solution of the SVM classification task," *IEEE Transactions on Neural Networks*, vol. 18, no. 5, pp. 1545–1549, 2007.

[16] S. Theodoridis and M. Mavroforakis, "Reduced convex hulls: A geometric approach to Support Vector Machines," *IEEE Signal Processing Magazine*, vol. 24, no. 3, pp. 119–122, May 2007.

[17] D. Tzikas, A. Likas, and N. Galatsanos, "Sparse bayesian modeling with adaptive kernel learning," *IEEE Trans. Neural Networks*, vol. 20, no. 6, pp. 926–937, 2009.

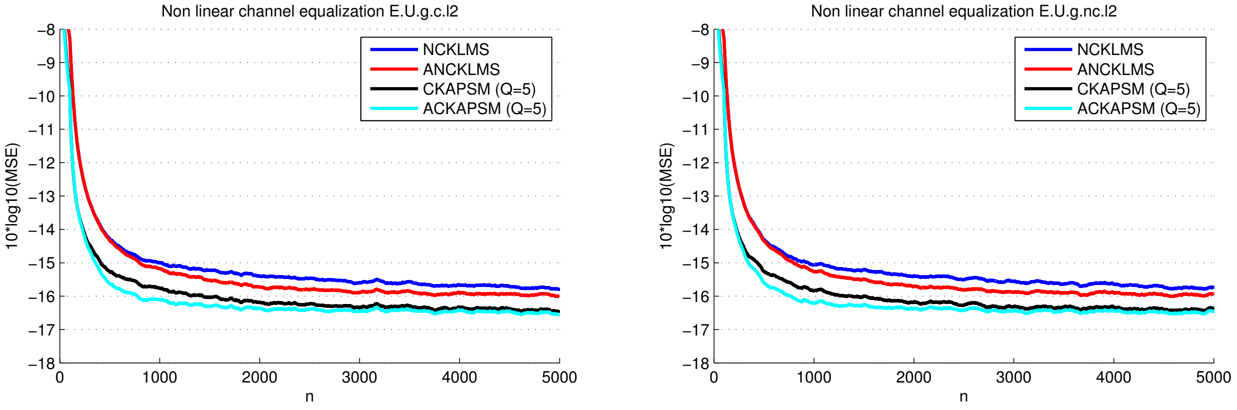
[18] P. Bouboulis, K. Slavakis, and S. Theodoridis, "Adaptive kernel-based image denoising employing semi-parametric regularization," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1465–1479, 2010.

[19] 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*, vol. 59, no. 3, pp. 964–978, 2011.



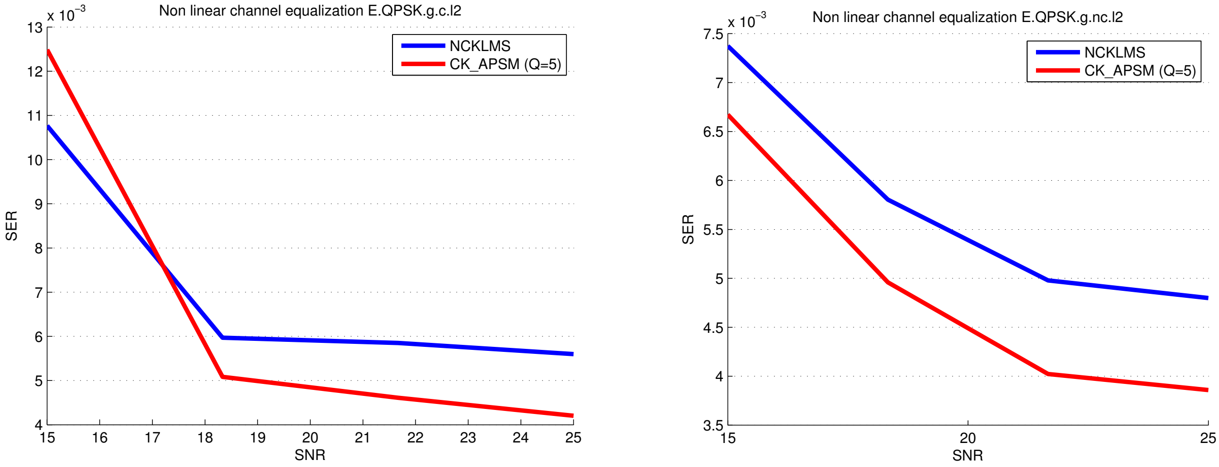
(a) (b)

Fig. 5. Learning curves for NCKLMS ($\mu = 1/2$), ANCKLMS, ($\mu = 1/4$), CKAPSM and ACKAPSM (filter length $L = 5$, delay $D = 2$) for the nonlinear channel equalization problem with gaussian input and gaussian noise at 20dB, for (a) the circular input case ($\rho = \sqrt{2}/2$) and (b) the non-circular input case ($\rho = 0.1$). In the realization of the CKAPSM and ACKAPSM the l_2 loss function was employed.



(a) (b)

Fig. 6. Learning curves for NCKLMS ($\mu = 1/2$), ANCKLMS, ($\mu = 1/4$), CKAPSM and ACKAPSM (filter length $L = 5$, delay $D = 2$) for the nonlinear channel equalization problem with uniform input and gaussian noise at 20dB, for (a) the circular input case ($\rho = \sqrt{2}/2$) and (b) the non-circular input case ($\rho = 0.1$). In the realization of the CKAPSM and ACKAPSM the l_2 loss function was employed.



(a) (b)

Fig. 7. Symbol Error Rate versus SNR for NCKLMS ($\mu = 1/2$) and CKAPSM (filter length $L = 5$, delay $D = 2$) for the nonlinear QPSK equalization problem with gaussian noise, for (a) the circular input case and (b) the non-circular input case. In the realization of the CKAPSM the l_2 loss function was employed.

[20] B. Scholkopf, K. Tsuda, and J.-P. Vert, *Kernel methods in computational biology*. MIT Press, 2004.

[21] G. Wachman, R. Khardon, P. Protopapas, and C. R. Alcock, "Kernels for periodic time series arising in astronomy," *Machine Learning and Knowledge Discovery in Databases, LNCS*, vol. 5782, pp. 489–505, 2009.

[22] D. B. Ion, "Reproducing kernel Hilbert spaces and extremal problems for scattering of particles with arbitrary spins," *International Journal of Theoretical Physics*, vol. 24, no. 12, pp. 1217–1231, 1985.

[23] H. Yoshino, C. Dong, Y. Washizawa, and Y. Yamashita, "Kernel wiener filter and its application to pattern recognition," *IEEE Trans. Neural Networks*, vol. 21, no. 11, pp. 1719–1730, 2010.

[24] A. Tanaka, H. Imai, and M. Miyakoshi, "Kernel-induced sampling theorem," *IEEE Trans. Signal Process.*, vol. 58, no. 7, pp. 3569–3577, 2010.

[25] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American Mathematical Society*, vol. 68, pp. 337–404, 1950.

[26] 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*, vol. 209, pp. 415–446, 1909.

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

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

[29] B. Picinbono and P. Chevalier, "Widely linear estimation with complex data," *IEEE Transactions on Signal Processing*, vol. 43, no. 8, pp. 2030–2033, 1995.

[30] B. Picinbono, "On circularity," *IEEE Transactions on Signal Processing*, vol. 42, no. 12, pp. 3473–3482, 1994.

[31] M. Novey and T. Adali, "On extending the complex ICA algorithm to noncircular sources," *IEEE Transanctions on Signal Processing*, vol. 56, no. 5, pp. 2148–2154, 2008.

[32] P. Chevalier and F. Pipon, "New insights into optimal widely linear receivers for the demodulation of BPSK, MSK and GMSK signals corrupted by noncircular interferences - application to SAIC," *IEEE Transactions on Signal Processing*, vol. 54, no. 3, pp. 870–883, 2006.

[33] J. J. Jeon, J. G. Andrews, and K. M. Sung, "The blind widely linear output energy algorithm for DS-CDMA systems," *IEEE Transactions on Signal Processing*, vol. 54, no. 5, pp. 1926–1931, 2006.

[34] A. Aghaei, K. Plataniotis, and S. Pasupathy, "Widely linear MMSE receivers for linear dispersion space-time block-codes," *IEEE Transactions on Wireless Communications*, vol. 9, no. 1, pp. 8 – 13, 2010.

[35] J. Via, D. Ramirez, and I. Santamaria, "Properness and widely linear processing of quaternion random vectors," *IEEE Transactions on Information Theory*, vol. 56, no. 7, pp. 3502 – 3515, 2010.

[36] A. Cacciapuoti, G. Gelli, L. Paura, and F. Verde, "Finite-sample performance analysis of widely linear multiuser receivers for DS-CDMA systems," *IEEE Transactions on Signal Processing*, vol. 56, no. 4, pp. 1572 – 1588, 2008.

[37] ——, "Widely linear versus linear blind multiuser detection with subspace-based channel estimation: finite sample-size effects," *IEEE Transactions on Signal Processing*, vol. 57, no. 4, pp. 1426 – 1443, 2008.

[38] K. Kuchi and V. Prabhu, "Performance evaluation for widely linear demodulation of PAM/QAM signals in the presence of Rayleigh fading and co-channel interference," *IEEE Transactions on Communications*, vol. 57, no. 1, pp. 183 – 193, 2009.

[39] J. Navarro-Moreno, J. Moreno-Kayser, R. Fernandez-Alcala, and J. Ruiz-Molina, "Widely linear estimation algorithms for second-order stationary signals," *IEEE Transactions on Signal Processing*, vol. 57, no. 12, pp. 4930 – 4935, 2009.

[40] R. Schober, W. Gerstacker, and L.-J. Lampe, "Data-aided and blind stochastic gradient algorithms for widely linear MMSE MAI suppression for DS-CDMA," *IEEE Transactions on Signal Processing*, vol. 52, no. 3, pp. 746 – 756, 2004.

[41] D. Mattera, L. Paura, and F. Sterle, "Widely linear MMSE equaliser for MIMO linear time-dispersive channel," *Electronics Letters*, vol. 39, no. 20, pp. 1481 – 1482, 2003.

[42] F. Sterle, "Widely linear MMSE transceivers for MIMO channels," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 4258 – 4270, 2007.

[43] K. C. Pun and T. Nguyen, "Widely linear filter bank equalizer for real STBC," *IEEE Transactions on Signal Processing*, vol. 56, no. 9, pp. 4544 – 4548, 2008.

[44] I. Yamada and N. Ogura, "Adaptive projected subgradient method for asymptotic minimization of sequence of nonnegative convex functions," *Numerical Functional Analysis and Optimization*, vol. 25, no. 7&8, pp. 593–617, 2004.

[45] K. Slavakis, I. Yamada, and N. Ogura, "The adaptive projected subgradient method over the fixed point set of strongly attracting nonexpansive mappings," *Numerical Functional Analysis and Optimization*, vol. 27, no. 7&8, pp. 905–930, 2006.

[46] K. Slavakis and I. Yamada, "The adaptive projected subgradient method constrained by families of quasi-nonexpansive mappings and its application to online learning," July 2011, submitted for publication. [Online]. Available: arxiv.org/abs/1008.5231

[47] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, 2011.

[48] S. Haykin, *Adaptive Filter Theory*, 3rd ed. New Jersey: Prentice-Hall, 1996.

[49] A. H. Sayed, *Fundamentals of Adaptive Filtering*. New Jersey: John Wiley & Sons, 2003.

[50] S. Theodoridis, K. Slavakis, and I. Yamada, "Adaptive learning in a world of projections: a unifying framework for linear and nonlinear classification and regression tasks," *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 97–123, Jan. 2011.

[51] P. J. Huber, *Robust Statistics*. Wiley, 1981.

[52] V. I. Paulsen, "An Introduction to the theory of Reproducing Kernel Hilbert Spaces," September 2009, notes. [Online]. Available: www.math.uh.edu/∼vern/rkhs.pdf

[53] P. Bouboulis and M. Mavroforakis, "Reproducing kernel Hilbert spaces and fractal interpolation," *Journal of Computational and Applied Mathematics*, 2011.

[54] I. Steinwart, D. Hush, and C. Scovel, "An explicit description of the Reproducing Kernel Hilbert spaces of Gaussian RBF kernels," *IEEE Trans. Info. Theory*, vol. 52, no. 10, pp. 4635–4643, 2006.

[55] W. Wirtinger, "Zur formalen theorie der functionen von mehr complexen veranderlichen," *Mathematische Annalen*, vol. 97, pp. 357–375, 1927.

[56] H. Li, "Complex-valued adaptive signal processing using Wirtinger calculus and its application to Independent Component Analysis," Ph.D. dissertation, University of Maryland Baltimore County, 2008.

[57] T. Adali, H. Li, M. Novey, and J. F. Cardoso, "Complex ICA using nonlinear functions," *IEEE Trans. Signal Process.*, vol. 56, no. 9, pp. 4536–4544, 2008.

[58] P. Bouboulis, "Wirtingers calculus in general Hilbert spaces," *Tech Report, University of Athens*, 2010. [Online]. Available: http://arxiv.org/abs/1005.5170

[59] R. T. Rockafellar and R. J.-B. Wets, *Variational Analysis*. Berlin: Springer, 2004.

[60] 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).

[61] R. T. Rockafellar, *Convex analysis*, ser. Princeton Mathematics Series. Princeton: Princeton University Press, 1970, vol. 28.

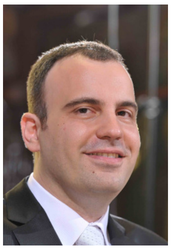
[62] P. Bouboulis, S. Theodoridis, and M. Mavroforakis, "Widely linear kernel-based adaptive filters," *19th European Signal Processing Conference (EUSIPCO 2011), Barcelona, Spain, August 29 - September 2, 2011.*

[63] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. Springer, 2009.

[64] A. H. Sayed, *Fundamentals of adaptive filtering*. New Jersey: John Wiley & Sons, 2003.

[65] G. S. Kimeldorf and G. Wahba, "Some results on Tchebycheffian spline functions," *J. Math. Anal. Applic.*, vol. 33, pp. 82–95, 1971.

[66] K. Zarifi and A. B. Gershman, "Blind subspace-based signature waveform estimation in BPSK-modulated DS-CDMA systems with circular noise," *IEEE Trans. Sig. Proc.*, vol. 54, no. 9, pp. 3592–3602, 2006.



Pantelis Bouboulis (M' 10) received the M.Sc. and Ph.D. degrees in informatics and telecommunications from the National and Kapodistrian University of Athens, Greece, in 2002 and 2006, respectively. From 2007 till 2008, he served as an Assistant Professor in the Department of Informatics and Telecommunications, University of Athens. Since 2008, he teaches mathematics in Greek High Schools. His current research interests lie in the areas of machine learning, fractals, wavelets and image processing. Dr. Bouboulis serves as an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS and the Greek magazine Euclid of the Hellenic Mathematical Society.

Konstantinos Slavakis (M' 08) received the M.E. and Ph.D. degrees in electrical and electronic engineering from the Tokyo Institute of Technology (TokyoTech), Tokyo, Japan, in 1999 and 2002, respectively. From 1996 to 2002, he was a recipient of the Japanese Government (MEXT) Scholarship. For the period of 2004 to 2006, he was with TokyoTech as a Japan Society for the Promotion of Science (JSPS) Postdoctoral Fellow, and from 2006 till 2007, he was a Postdoctoral Fellow in the Department of Informatics and Telecommunications, University of Athens, Greece, under the ENTER Program. Since September of 2007, he has been an Assistant Professor for the Department of Telecommunications Science and Technology, University of Peloponnese, Tripolis, Greece. Dr. Slavakis serves as an Associate and Area Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING. His current research interests are the applications of convex analysis and computational algebraic geometry to signal processing, machine learning, array, and multidimensional systems problems. He received the Tejima Memorial Award of TokyoTech for his Ph.D. dissertation.

Sergios Theodoridis (F08) is currently Professor of signal processing and communications in the Department of Informatics and Telecommunications, University of Athens, Athens, Greece. His research interests lie in the areas of adaptive algorithms and communications, machine learning and pattern recognition, and signal processing for audio processing and retrieval. He is the co-editor of the book Efficient Algorithms for Signal Processing and System Identification (Prentice-Hall, 1993), coauthor of the best selling book Pattern Recognition (Academic, 4th ed., 2008), and the coauthor of three books in Greek, two of them for the Greek Open University. He is currently an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II, and a member of the editorial board of the EURASIP Wireless Communications and Networking. He has served in the past as an Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING, the IEEE Signal Processing Magazine, the EURASIP Journal on Signal Processing, and the EURASIP Journal on Advances on Signal Processing. He was the general chairman of EUSIPCO 1998, the Technical Program cochair for ISCAS 2006, and cochairman of ICIP 2008. He has served as President of the European Association for Signal Processing (EURASIP) and he is currently a member of the Board of Governors for the IEEE CAS Society. He is the coauthor of four papers that have received best paper awards including the 2009 IEEE Computational Intelligence Society Transactions on Neural Networks Outstanding paper Award. He serves as an IEEE Signal Processing Society Distinguished Lecturer. He is a member of the Greek National Council for Research and Technology and Chairman of the SP advisory committee for the Edinburgh Research Partnership (ERP). He has served as vice chairman of the Greek Pedagogical Institute and for four years, he was a member of the Board of Directors of COSMOTE (the Greek mobile phone operating company). He is Fellow of IET and a Corresponding Fellow of FRSE.