Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2008, Article ID 735351, 16 pages
doi:10.1155/2008/735351

Research Article

Sliding Window Generalized Kernel Affine Projection Algorithm Using Projection Mappings

Konstantinos Slavakis[1] and Sergios Theodoridis[2]

[1] *Department of Telecommunications Science and Technology, University of Peloponnese, Karaiskaki St., Tripoli 22100, Greece*
[2] *Department of Informatics and Telecommunications, University of Athens, Ilissia, Athens 15784, Greece*

Correspondence should be addressed to Konstantinos Slavakis, slavakis@uop.gr

Received 8 October 2007; Revised 25 January 2008; Accepted 17 March 2008

Recommended by Theodoros Evgeniou

Very recently, a solution to the kernel-based online classification problem has been given by the adaptive projected subgradient method (APSM). The developed algorithm can be considered as a generalization of a kernel affine projection algorithm (APA) and the kernel normalized least mean squares (NLMS). Furthermore, sparsification of the resulting kernel series expansion was achieved by imposing a closed ball (convex set) constraint on the norm of the classifiers. This paper presents another sparsification method for the APSM approach to the online classification task by generating a sequence of linear subspaces in a reproducing kernel Hilbert space (RKHS). To cope with the inherent memory limitations of online systems and to embed tracking capabilities to the design, an upper bound on the dimension of the linear subspaces is imposed. The underlying principle of the design is the notion of projection mappings. Classification is performed by metric projection mappings, sparsification is achieved by orthogonal projections, while the online system's memory requirements and tracking are attained by oblique projections. The resulting sparsification scheme shows strong similarities with the classical sliding window adaptive schemes. The proposed design is validated by the adaptive equalization problem of a nonlinear communication channel, and is compared with classical and recent stochastic gradient descent techniques, as well as with the APSM's solution where sparsification is performed by a closed ball constraint on the norm of the classifiers.

1. INTRODUCTION

Kernel methods play a central role in modern classification and nonlinear regression tasks and they can be viewed as the nonlinear counterparts of linear supervised and unsupervised learning algorithms [1–3]. They are used in a wide variety of applications from pattern analysis [1–3], equalization or identification in communication systems [4, 5], to time series analysis and probability density estimation [6–8].

A positive-definite kernel function defines a high- or even infinite-dimensional *reproducing kernel Hilbert space (RKHS)* $\mathcal{H}$, widely called *feature space* [1–3, 9, 10]. It also gives a way to map *data*, collected from the Euclidean *data space*, to the feature space $\mathcal{H}$. In such a way, processing is transfered to the high-dimensional feature space, and the classification task in $\mathcal{H}$ is expected to be linearly separable according to Cover's theorem [1]. The inner product in $\mathcal{H}$ is given by a simple evaluation of the kernel function on the data space, while the explicit knowledge of the feature space $\mathcal{H}$ is unnecessary. This is well known as the *kernel trick* [1–3].

We will focus on the two-class classification task, where the goal is to classify an unknown feature vector $\mathbf{x}$ to one of the two classes, based on the classifier value $f(\mathbf{x})$. The online setting will be considered here, where data arrive sequentially. If these data are represented by the sequence $(\mathbf{x}_n)_{n\geq 0} \subset \mathbb{R}^m$, where m is a positive integer, then the objective of online kernel methods is to form an estimate of f in $\mathcal{H}$ given by a kernel series expansion:

$$\hat{f} := \sum_{n=0}^{\infty} \gamma_n \kappa(\mathbf{x}_n, \cdot) \in \mathcal{H}, \tag{1}$$

where κ stands for the kernel function, $(\mathbf{x}_n)_{n\geq 0}$ parameterizes the kernel function, $(\gamma_n)_{n\geq 0} \subset \mathbb{R}$, and we assume, of course, that the right-hand side of (1) converges.

A convex analytic viewpoint of the online classification task in an RKHS was given in [11]. The standard classification problem was viewed as the problem of finding a point in a *closed half-space* (a special closed convex set) of $\mathcal{H}$. Since data arrive sequentially in an online setting, online classification was considered as the task of finding a point in the nonempty intersection of an infinite sequence of closed half-spaces. A solution to such a problem was given by the recently developed *adaptive projected subgradient method (APSM)*, a convex analytic tool for the convexly constrained asymptotic minimization of an infinite sequence of nonsmooth, nonnegative convex, but not necessarily differentiable objectives in real Hilbert spaces [12–14]. It was discovered that many projection-based adaptive filtering [15] algorithms like the classical *normalized least mean squares (NLMS)* [16, 17], the more recently explored *affine projection algorithm (APA)* [18, 19], as well as more recently developed algorithms [20–28] become special cases of the APSM [13, 14]. In the same fashion, the present algorithm can be viewed as a generalization of a kernel affine projection algorithm.

To form the functional representation in (1), the coefficients $(\gamma_n)_{n\geq 0}$ must be kept in memory. Since the number of incoming data increases, the memory requirements as well as the necessary computations of the system increase linearly with time [29], leading to a conflict with the limitations and complexity issues as posed by any online setting [29, 30]. Recent research focuses on *sparsification* techniques, that is, on introducing criteria that lead to an approximate representation of (1) using a finite subset of $(\gamma_n)_{n\geq 0}$. This is equivalent to identifying those kernel functions whose removal is expected to have a negligible effect, in some predefined sense, or, equivalently, building dictionaries out of the sequence $(\kappa(\mathbf{x}_n, \cdot))_{n\geq 0}$ [31–36].

To introduce sparsification, the design in [30], apart from the sequence of closed half-spaces, imposes an additional constraint on the norm of the classifier. This leads to a sparsified representation of the expansion of the solution given in (1), with an effect similar to that of a forgetting factor which is used in *recursive-least-squares- (RLS-)* [15] type algorithms.

This paper follows a different path to the sparsification in the line with the rationale adopted in [36]. A sequence of linear subspaces $(M_n)_{n\geq 0}$ of $\mathcal{H}$ is formed, by using the incoming data together with an *approximate linear dependency/independency* criterion. To satisfy the memory requirements of the online system, and in order to provide with tracking capabilities to our design, a bound on the dimension of the generating subspaces $(M_n)_{n\geq 0}$ is imposed. This upper bound turns out to be equivalent to the length of a memory buffer. Whenever the buffer becomes full and each time a new data enters the system, an old observation is discarded. Hence, an upper bound on dimension results into a *sliding window* effect. The underlying principle of the proposed design is the notion of projection mappings. Indeed, classification is performed by *metric projection* mappings, sparsification is conducted by *orthogonal projections* onto the generated linear subspaces $(M_n)_{n\geq 0}$, and memory limitations (which lead to enhanced tracking capabilities) are established by employing *oblique projections*. Note that although the classification problem is considered here, the tools can readily be adopted for regression tasks, with different cost functions that can be either differentiable or nondifferentiable.

The paper is organized as follows. Mathematical preliminaries and elementary facts on projection mappings are given in Section 2. A short description of the convex analytic perspective introduced in [11, 30] is presented in Sections 3 and 4, respectively. A byproduct of this approach, a kernel affine projection algorithm (APA), is introduced in Section 4.2. The sparsification procedure based on the generation of a sequence of linear subspaces is given in Section 5. To validate the design, the adaptive equalization problem of a nonlinear channel is chosen. We compare the present scheme with the classical kernel perceptron algorithm, its generalization, the NORMA method [29], as well as the APSM's solution but with the norm constraint sparsification [30] in Section 7. In Section 8, we conclude our discussion, and several clarifications as well as a table of the main symbols, used in the paper, are gathered in the appendices.

2. MATHEMATICAL PRELIMINARIES

Henceforth, the set of all integers, nonnegative integers, positive integers, real and complex numbers will be denoted by $\mathbb{Z}$, $\mathbb{Z}_{\geq 0}$, $\mathbb{Z}_{>0}$, $\mathbb{R}$ and $\mathbb{C}$, respectively. Moreover, the symbol $\text{card}(\mathcal{J})$ will stand for the cardinality of a set $\mathcal{J}$, and $\overline{j_1, j_2} := \{j_1, j_1 + 1, \ldots, j_2\}$, for any integers $j_1 \leq j_2$.

2.1. Reproducing kernel Hilbert space

We provide here with a few elementary facts about reproducing kernel Hilbert spaces (RKHS). The symbol $\mathcal{H}$ will stand for an infinite-dimensional, in general, real Hilbert space [37, 38] equipped with an inner product denoted by $\langle \cdot, \cdot \rangle$. The induced norm in $\mathcal{H}$ will be given by $\|f\| := \langle f, f \rangle^{1/2}$, for all $f \in \mathcal{H}$. An example of a finite-dimensional real Hilbert space is the well-known Euclidean space $\mathbb{R}^m$ of dimension $m \in \mathbb{Z}_{>0}$. In this space, the inner product is nothing but the vector dot product $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle := \mathbf{x}_1^t \mathbf{x}_2$, for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^m$, where the superscript $(\cdot)^t$ stands for vector transposition.

Assume a real Hilbert space $\mathcal{H}$ which consists of functions defined on $\mathbb{R}^m$, that is, $f : \mathbb{R}^m \to \mathbb{R}$. The function $\kappa(\cdot, \cdot) : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ is called a *reproducing kernel* of $\mathcal{H}$ if

(1) for every $\mathbf{x} \in \mathbb{R}^m$, the function $\kappa(\mathbf{x}, \cdot) : \mathbb{R}^m \to \mathbb{R}$ belongs to $\mathcal{H}$,
(2) the *reproducing property* holds, that is,

$$f(\mathbf{x}) = \langle f, \kappa(\mathbf{x}, \cdot) \rangle, \quad \forall \mathbf{x} \in \mathbb{R}^m, \ \forall f \in \mathcal{H}. \tag{2}$$

In this case, $\mathcal{H}$ is called a *reproducing kernel Hilbert space (RKHS)* [2, 3, 9]. If such a function $\kappa(\cdot, \cdot)$ exists, it is unique [9]. A reproducing kernel is positive definite and symmetric in its arguments [9]. (A kernel κ is called positive definite if $\sum_{l,j=1}^{N} \xi_l \xi_j \kappa(\mathbf{x}_l, \mathbf{x}_j) \geq 0$, for all $\xi_l, \xi_j \in \mathbb{R}$, for all $\mathbf{x}_l, \mathbf{x}_j \in \mathbb{R}^m$, and for any $N \in \mathbb{Z}_{>0}$ [9]. This property underlies the kernel functions firstly studied by Mercer [10].) In addition, the Moore-Aronszajn theorem [9] guarantees that to every

positive definite function $\kappa(\cdot,\cdot) : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ there corresponds a unique RKHS $\mathcal{H}$ whose reproducing kernel is κ itself [9]. Such an RKHS is generated by taking first the space of all finite combinations $\sum_j \gamma_j \kappa(\mathbf{x}_j, \cdot)$, where $\gamma_j \in \mathbb{R}$, $\mathbf{x}_j \in \mathbb{R}^m$, and then completing this space by considering also all its limit points [9]. Notice here that, by (2), the inner product of $\mathcal{H}$ is realized by a simple evaluation of the kernel function, which is well known as the *kernel trick* [1, 2]; $\langle \kappa(\mathbf{x}_i, \cdot), \kappa(\mathbf{x}_j, \cdot)\rangle = \kappa(\mathbf{x}_i, \mathbf{x}_j)$, for all $i, j \in \mathbb{Z}_{\geq 0}$.

There are numerous kernel functions and associated RKHS $\mathcal{H}$, which have extensively been used in pattern analysis and nonlinear regression tasks [1–3]. Celebrated examples are (i) the linear kernel $\kappa(\mathbf{x}, \mathbf{y}) := \mathbf{x}^t\mathbf{y}$, for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^m$ (here the RKHS $\mathcal{H}$ is the data space $\mathbb{R}^m$ itself), and (ii) the Gaussian or radial basis function (RBF) kernel $\kappa(\mathbf{x}, \mathbf{y}) := \exp(-((\mathbf{x}-\mathbf{y})^t(\mathbf{x}-\mathbf{y}))/2\sigma^2)$, for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^m$, where $\sigma > 0$ (here the associated RKHS is of infinite dimension [2, 3]). For more examples and systematic ways of generating more involved kernel functions by using fundamental ones, the reader is referred to [2, 3]. Hence, an RKHS offers a unifying framework for treating several types of nonlinearities in classification and regression tasks.

2.2. Closed convex sets, metric, orthogonal, and oblique projection mappings

A subset C of $\mathcal{H}$ will be called convex if for all $\widehat{f}_1, \widehat{f}_2 \in C$ the segment $\{\lambda\widehat{f}_1 + (1-\lambda)\widehat{f}_2 : \lambda \in [0,1]\}$ with endpoints $\widehat{f}_1$ and $\widehat{f}_2$ lies in C. A function $\Theta : \mathcal{H} \to \mathbb{R} \cup \{\infty\}$ will be called convex if for all $f_1, f_2 \in \mathcal{H}$ and for all $\lambda \in (0,1)$ we have $\Theta(\lambda f_1 + (1-\lambda)f_2) \leq \lambda\Theta(f_1) + (1-\lambda)\Theta(f_2)$.

Given any point $f \in \mathcal{H}$, we can quantify its distance from a nonempty closed convex set C by the *metric distance* function $d(\cdot, C) : \mathcal{H} \to \mathbb{R} : f \mapsto d(f, C) := \inf\{\|f - \widehat{f}\| : \widehat{f} \in C\}$ [37, 38], where inf denotes the infimum. The function $d(\cdot, C)$ is nonnegative, continuous, and convex [37, 38]. Note that any point $\widehat{f} \in C$ is of zero distance from C, that is, $d(\widehat{f}, C) = 0$, and that the set of all minimizers of $d(\cdot, C)$ over $\mathcal{H}$ is C itself.

Given a point $f \in \mathcal{H}$ and a closed convex set $C \subset \mathcal{H}$, an efficient way to move from f to a point in C, that is, to a minimizer of $d(\cdot, C)$, is by means of the *metric projection mapping* P_C *onto* C, which is defined as the mapping that takes f to the *uniquely* existing point $P_C(f)$ of C that achieves the infimum value $\|f - P_C(f)\| = d(f, C)$ [37, 38]. For a geometric interpretation refer to Figure 1. Clearly, if $f \in C$ then $P_C(f) = f$.

A well-known example of a closed convex set is a *closed linear subspace* M [37, 38] of a real Hilbert space $\mathcal{H}$. The metric projection mapping P_M is called now *orthogonal projection* since the following property holds: $\langle f - P_M(f), \widehat{f}\rangle = 0$, for all $\widehat{f} \in M$, for all $f \in \mathcal{H}$ [37, 38]. Given an $f' \in \mathcal{H}$, the shift of a closed linear subspace M by f', that is, $V := f' + M := \{f' + f : f \in M\}$, is called an *(affine) linear variety* [38].

Given $a \neq 0$ in $\mathcal{H}$ and $\xi \in \mathbb{R}$, let a *closed half-space* be the closed convex set $\Pi^+ := \{\widehat{f} \in \mathcal{H} : \langle a, \widehat{f}\rangle \geq \xi\}$, that is, Π^+ is the set of all points that lie on the "positive" side of the *hyperplane* $\Pi := \{\widehat{f} \in \mathcal{H} : \langle a, \widehat{f}\rangle = \xi\}$, which defines the boundary of Π^+ [37]. The vector a is usually called the *normal vector* of Π^+. The metric projection operator P_{Π^+} can easily be obtained by simple geometric arguments, and it is shown to have the closed-form expression [37, 39]:

$$P_{\Pi^+}(f) = f + \frac{(\xi - \langle a, f\rangle)^+}{\|a\|^2}a, \quad \forall f \in \mathcal{H}, \tag{3}$$

where $\tau^+ := \max\{0, \tau\}$ denotes the positive part of a $\tau \in \mathbb{R}$.

Given the center $\widehat{f}_0 \in \mathcal{H}$ and the radius $\delta > 0$, we define the *closed ball* $B[\widehat{f}_0, \delta] := \{\widehat{f} \in \mathcal{H} : \|\widehat{f}_0 - \widehat{f}\| \leq \delta\}$ [37]. The closed ball $B[\widehat{f}_0, \delta]$ is clearly a closed convex set, and its metric projection mapping is given by the simple formula: for all $f \in \mathcal{H}$,

$$P_{B[\widehat{f}_0,\delta]}(f) = \begin{cases} f, & \text{if } \|f - \widehat{f}_0\| \leq \delta, \\ \widehat{f}_0 + \dfrac{\delta}{\|f - \widehat{f}_0\|}(f - \widehat{f}_0), & \text{if } \|f - \widehat{f}_0\| > \delta, \end{cases} \tag{4}$$

which is the point of intersection of the sphere and the segment joining f and the center of the sphere in the case where $f \notin B[\widehat{f}_0, \delta]$ (see Figure 1).

Let, now, M and M' be linear subspaces of a finite-dimensional linear subspace $V \subset \mathcal{H}$. Then, let $M + M'$ be defined as the subspace $M + M' := \{h + h' : h \in M, h' \in M'\}$. If also $M \cap M' = \{0\}$, then $M + M'$ is called the direct sum of M and M' and is denoted by $M \oplus M'$ [40, 41]. In the case where $V = M \oplus M'$, then every $f \in V$ can be expressed uniquely as a sum $f = h + h'$, where $h \in M$ and $h' \in M'$ [40, 41]. Then, we define here a mapping $P_{M,M'} : V = M \oplus M' \to M$ which takes any $f \in V$ to that unique $h \in M$ that appears in the decomposition $f = h + h'$. We will call h the *(oblique) projection of* f *on* M *along* M' [40] (see Figure 1).

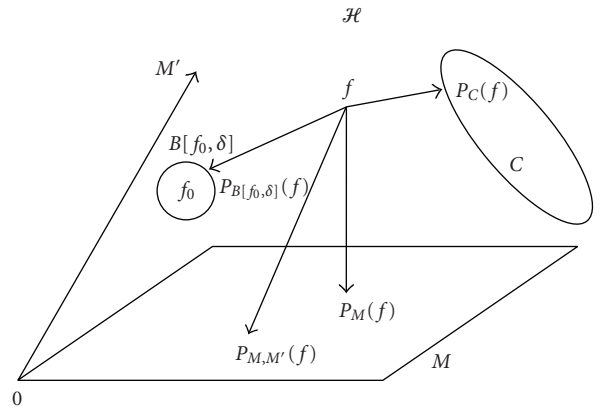

Figure 1: An illustration of the metric projection mapping P_C onto the closed convex subset C of $\mathcal{H}$, the projection $P_{B[f_0,\delta]}$ onto the closed ball $B[f_0, \delta]$, the orthogonal projection P_M onto the closed linear subspace M, and the oblique projection $P_{M,M'}$ on M along the closed linear subspace M'.

3. CONVEX ANALYTIC VIEWPOINT OF KERNEL-BASED CLASSIFICATION

In pattern analysis [1, 2], *data* are usually given by a sequence of vectors $(\mathbf{x}_n)_{n\in\mathbb{Z}_{\geq 0}} \subset \mathcal{X} \subset \mathbb{R}^m$, for some $m \in \mathbb{Z}_{>0}$. We will assume that each vector in $\mathcal{X}$ is drawn from two classes and is thus associated to a label $y_n \in \mathcal{Y} := \{\pm 1\}$, $n \in \mathbb{Z}_{\geq 0}$. As such, a sequence of (training) pairs $\mathcal{D} := ((\mathbf{x}_n, y_n))_{n\in\mathbb{Z}_{\geq 0}} \subset \mathcal{X} \times \mathcal{Y}$ is formed.

To benefit from a larger than m or even infinite-dimensional space, modern pattern analysis reformulates the classification problem in an RKHS $\mathcal{H}$ (implicitly defined by a predefined kernel function κ), which is widely known as the *feature space* [1–3]. A mapping $\phi : \mathbb{R}^m \to \mathcal{H}$ which takes $(\mathbf{x}_n)_{n\in\mathbb{Z}_{\geq 0}} \subset \mathbb{R}^m$ onto $(\phi(\mathbf{x}_n))_{n\in\mathbb{Z}_{\geq 0}} \subset \mathcal{H}$ is given by the kernel function associated to the RKHS feature space $\mathcal{H}$: $\phi(\mathbf{x}) := \kappa(\mathbf{x}, \cdot) \in \mathcal{H}$, for all $\mathbf{x} \in \mathbb{R}^m$. Then, the *classification problem* is defined in the feature space $\mathcal{H}$ as selecting a point $\widehat{f} \in \mathcal{H}$ and an offset $\widehat{b} \in \mathbb{R}$ such that $y(\widehat{f}(\mathbf{x}) + \widehat{b}) \geq \rho$, for all $(\mathbf{x}, y) \in \mathcal{D}$, and for some *margin* $\rho \geq 0$ [1, 2].

For convenience, we merge $f \in \mathcal{H}$ and $b \in \mathbb{R}$ into a single vector $u := (f, b) \in \mathcal{H} \times \mathbb{R}$, where $\mathcal{H} \times \mathbb{R}$ stands for the product space [37, 38] of $\mathcal{H}$ and $\mathbb{R}$. Henceforth, we will call a point $u \in \mathcal{H} \times \mathbb{R}$ a *classifier*, and $\mathcal{H} \times \mathbb{R}$ the space of all classifiers. The space $\mathcal{H} \times \mathbb{R}$ of all classifiers can be endowed with an inner product as follows: for any $u_1 := (f_1, b_1)$, $u_2 := (f_2, b_2) \in \mathcal{H} \times \mathbb{R}$, let $\langle u_1, u_2\rangle_{\mathcal{H}\times\mathbb{R}} := \langle f_1, f_2\rangle_{\mathcal{H}} + b_1 b_2$. The space $\mathcal{H} \times \mathbb{R}$ of all classifiers becomes then a Hilbert space. The notation $\langle\cdot,\cdot\rangle$ will be used for both $\langle\cdot,\cdot\rangle_{\mathcal{H}\times\mathbb{R}}$ and $\langle\cdot,\cdot\rangle_{\mathcal{H}}$.

A standard penalty function to be minimized in classification problems is the *soft margin loss function* [1, 29] defined on the space of all classifiers $\mathcal{H} \times \mathbb{R}$ as follows: given a pair $(\mathbf{x}, y) \in \mathcal{D}$ and the margin parameter $\rho \geq 0$,

$$l_{\mathbf{x},y,\rho}(u) : \mathcal{H}\times\mathbb{R} \longrightarrow \mathbb{R} : \underbrace{(f,b)}_{u} \longmapsto (\rho - y(f(\mathbf{x}) + b))^+ = (\rho - y g_{f,b}(\mathbf{x}))^+, \tag{5}$$

where the function $g_{f,b}$ is defined by

$$g_{f,b}(\mathbf{x}) := f(\mathbf{x}) + b, \quad \forall \mathbf{x} \in \mathbb{R}^m,\ \forall (f,b) \in \mathcal{H}\times\mathbb{R}. \tag{6}$$

If the classifier $\widehat{u} := (\widehat{f}, \widehat{b})$ is such that $y g_{\widehat{f},\widehat{b}}(\mathbf{x}) < \rho$, then this classifier fails to achieve the margin ρ at $(\mathbf{x}, y)$ and (5) scores a penalty. In such a case, we say that the classifier committed a *margin error*. A *misclassification* occurs at $(\mathbf{x}, y)$ if $y g_{\widehat{f},\widehat{b}}(\mathbf{x}) < 0$.

The studies in [11, 30] approached the classification task from a convex analytic perspective. By the definition of the classification problem, our goal is to look for classifiers (points in $\mathcal{H} \times \mathbb{R}$) that belong to the set $\Pi^+_{\mathbf{x},y,\rho} := \{(\widehat{f}, \widehat{b}) \in \mathcal{H}\times\mathbb{R} : y(\widehat{f}(\mathbf{x}) + \widehat{b}) \geq \rho\}$. If we recall the reproducing property (2), a desirable classifier satisfies $y(\langle \widehat{f}, \kappa(\mathbf{x},\cdot)\rangle + \widehat{b}) \geq \rho$ or $\langle \widehat{f}, y\kappa(\mathbf{x},\cdot)\rangle_{\mathcal{H}} + y\widehat{b} \geq \rho$. Thus, for a given pair $(\mathbf{x}, y)$ and a margin ρ, by the definition of the inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}\times\mathbb{R}}$, the set of all *desirable* classifiers (that do not commit a margin error at $(\mathbf{x}, y)$) is

$$\Pi^+_{\mathbf{x},y,\rho} = \{\widehat{u} \in \mathcal{H}\times\mathbb{R} : \langle \widehat{u}, a_{\mathbf{x},y}\rangle_{\mathcal{H}\times\mathbb{R}} \geq \rho\}, \tag{7}$$

where $a_{\mathbf{x},y} := (y\kappa(\mathbf{x},\cdot), y) = y(\kappa(\mathbf{x},\cdot), 1) \in \mathcal{H}\times\mathbb{R}$. The vector $(\kappa(\mathbf{x},\cdot), 1) \in \mathcal{H}\times\mathbb{R}$ is an extended (to account for the constant factor $\widehat{b}$) vector that is completely specified by the point $\mathbf{x}$ and the adopted kernel function. By (7), we notice that $\Pi^+_{\mathbf{x},y,\rho}$ is a closed half-space of $\mathcal{H} \times \mathbb{R}$ (see Section 2.2). *That is, all classifiers that do not commit a margin error at $(\mathbf{x}, y)$ belong in the closed half-space $\Pi^+_{\mathbf{x},y,\rho}$ specified by the chosen kernel function.*

The following proposition builds the bridge between the standard loss function $l_{\mathbf{x},y,\rho}$ and the closed convex set $\Pi^+_{\mathbf{x},y,\rho}$.

Proposition 1 (see [11, 30]). *Given the parameters $(\mathbf{x}, y, \rho)$, the closed half-space $\Pi^+_{\mathbf{x},y,\rho}$ coincides with the set of all minimizers of the soft margin loss function, that is,* $\arg\min\{l_{\mathbf{x},y,\rho}(u) : u \in \mathcal{H}\times\mathbb{R}\} = \Pi^+_{\mathbf{x},y,\rho}$.

Starting from this viewpoint, the following section describes shortly a convex analytic tool [11, 30] which tackles the online classification task, where a sequence of parameters $(\mathbf{x}_n, y_n, \rho_n)_{n\in\mathbb{Z}_{\geq 0}}$, and thus a sequence of closed half-spaces $(\Pi^+_{\mathbf{x}_n,y_n,\rho_n})_{n\in\mathbb{Z}_{\geq 0}}$, is assumed.

4. THE ONLINE KERNEL-BASED CLASSIFICATION TASK AND THE ADAPTIVE PROJECTED SUBGRADIENT METHOD

At every time instant $n \in \mathbb{Z}_{\geq 0}$, a pair $(\mathbf{x}_n, y_n) \in \mathcal{D}$ becomes available. If we also assume a nonnegative margin parameter ρ_n, then we can define the set of all classifiers that achieve this margin by the closed half-space $\Pi^+_{\mathbf{x}_n,y_n,\rho_n} := \{\widehat{u} = (\widehat{f}, \widehat{b}) \in \mathcal{H}\times\mathbb{R} : y_n(\widehat{f}(\mathbf{x}_n) + \widehat{b}) \geq \rho_n\}$. Clearly, in an online setting, we deal with a sequence of closed half-spaces $(\Pi^+_{\mathbf{x}_n,y_n,\rho_n})_{n\in\mathbb{Z}_{\geq 0}} \subset \mathcal{H}\times\mathbb{R}$ and since each one of them contains the set of all desirable classifiers, our objective is to find a classifier that belongs to or satisfies most of these half-spaces or, more precisely, to find a classifier that belongs to all but a finite number of $\Pi^+_{\mathbf{x}_n,y_n,\rho_n}$s, that is, a $\widehat{u} \in \cap_{n\geq N_0}\Pi^+_{\mathbf{x}_n,y_n,\rho_n} \subset \mathcal{H}\times\mathbb{R}$, for some $N_0 \in \mathbb{Z}_{\geq 0}$. In other words, we look for a classifier in the intersection of these half-spaces.

The studies in [11, 30] propose a solution to the above problem by the recently developed *adaptive projected subgradient method (APSM)* [12–14]. The APSM approaches the above problem as an asymptotic minimization of a sequence of *not necessarily differentiable nonnegative convex functions* over a closed convex set in a real Hilbert space.

Instead of processing a single pair $(\mathbf{x}_n, y_n)$ at each n, APSM offers the freedom to process concurrently a set $\{(\mathbf{x}_j, y_j)\}_{j\in\mathcal{J}_n}$, where the index set $\mathcal{J}_n \subset \overline{0,n}$ for every $n \in \mathbb{Z}$, and where $\overline{j_1, j_2} := \{j_1, j_1+1, \ldots, j_2\}$ for every integers $j_1 \leq j_2$. Intuitively, concurrent processing is expected to increase the speed of an algorithm. Indeed, in adaptive filtering [15], it is the motivation behind the leap from NLMS [16, 17], where no concurrent processing is available, to the potentially faster APA [18, 19].

To keep the discussion simple, we assume that $n \in \mathcal{J}_n$, for all $n \in \mathbb{Z}_{\geq 0}$. An example of such an index set $\mathcal{J}_n$ is given in (13). In other words, (13) treats the case where at time instant n, the pairs $\{(\mathbf{x}_j, y_j)\}_{j\in\overline{n-q+1,n}}$, for some $q \in \mathbb{Z}_{>0}$, are considered. This is in line with the basic rationale of the celebrated affine projection algorithm (APA), which has extremely been used in adaptive filtering [15].

Each pair $(\mathbf{x}_j, y_j)$, and thus each index j, defines a half-space $\Pi^+_{\mathbf{x}_j,y_j,\rho_j^{(n)}}$ by (7). In order to point out explicitly the dependence of such a half-space on the index set $\mathcal{J}_n$, we slightly modify the notation for $\Pi_{\mathbf{x}_j,y_j,\rho_j^{(n)}}$ and use $\Pi^+_{j,n}$ for any $j \in \mathcal{J}_n$, and for any $n \in \mathbb{Z}_{\geq 0}$. The metric projection mapping $P_{\Pi^+_{j,n}}$ is analytically given by (3). To assign different importance to each one of the projections corresponding to $\mathcal{J}_n$, we associate to each half-space, that is, to each $j \in \mathcal{J}_n$, a weight $\omega_j^{(n)}$ such that $\omega_j^{(n)} \geq 0$, for all $j \in \mathcal{J}_n$, and $\sum_{j\in\mathcal{J}_n}\omega_j^{(n)} = 1$, for all $n \in \mathbb{Z}_{\geq 0}$. This is in line with the adaptive filtering literature that tends to assign higher importance in the most recent samples. For the less familiar reader, we point out that if $\mathcal{J}_n := \{n\}$, for all $n \in \mathbb{Z}_{\geq 0}$, the algorithm breaks down to the NLMS. Regarding the APA, a discussion can be found below.

As it is also pointed out in [29, 30], the major drawback of online kernel methods is the linear increase of complexity with time. To deal with this problem, it was proposed in [30] to further constrain the norm of the desirable classifiers by a closed ball. To be more precise, one constrains the desirable classifiers in [30] by $\mathcal{K} := B[0,\delta] \times \mathbb{R} \subset \mathcal{H} \times \mathbb{R}$, for some predefined $\delta > 0$. As a result, one seeks for classifiers that belong to $\mathcal{K} \cap (\bigcap_{j\in\mathcal{J}_n, n\geq N_0}\Pi^+_{j,n})$, for $\exists N_0 \in \mathbb{Z}_{\geq 0}$. By the definition of the closed ball $B[0,\delta]$ in Section 2.2, we easily see that the addition of $\mathcal{K}$ imposes a constraint on the norm of $\widehat{f}$ in the vector $\widehat{u} = (\widehat{f}, \widehat{b})$ by $\|\widehat{f}\| \leq \delta$. The associated metric projection mapping is analytically given by the simple computation $P_{\mathcal{K}}(u) = (P_{B[0,\delta]}(f), b)$, for all $u := (f, b) \in \mathcal{H} \times \mathbb{R}$, where $P_{B[0,\delta]}$ is obtained by (4). It was observed that constraining the norm results into a sequence of classifiers with a fading memory, where old data can be eliminated [30].

For the sake of completeness, we give a summary of the sparsified algorithm proposed in [30].

Algorithm 1 (see [30]). For any $n \in \mathbb{Z}_{\geq 0}$, consider the index set $\mathcal{J}_n \subset \overline{0,n}$, such that $n \in \mathcal{J}_n$. An example of $\mathcal{J}_n$ can be found in (13). For any $j \in \mathcal{J}_n$ and for any $n \in \mathbb{Z}_{\geq 0}$, let the closed half-space $\Pi^+_{j,n} := \{\widehat{u} = (\widehat{f}, \widehat{b}) \in \mathcal{H} \times \mathbb{R} : y_j(\widehat{f}(\mathbf{x}_j) + \widehat{b}) \geq \rho_j^{(n)}\}$, and the weight $\omega_j^{(n)} \geq 0$ such that $\sum_{j\in\mathcal{J}_n}\omega_j^{(n)} = 1$, for all $n \in \mathbb{Z}_{\geq 0}$. For an arbitrary initial offset $b_0 := (0, b_0) \in \mathcal{H} \times \mathbb{R}$ and generate the following point (classifier) sequence in $\mathcal{H} \times \mathbb{R}$ by

$$u_{n+1} := P_{\mathcal{K}}\left(u_n + \mu_n\left(\sum_{j\in\mathcal{J}_n}\omega_j^{(n)}P_{\Pi^+_{j,n}}(u_n) - u_n\right)\right), \quad \forall n \in \mathbb{Z}_{\geq 0}, \tag{8a}$$

where the *extrapolation coefficient* $\mu_n \in [0, 2\mathcal{M}_n]$ with

$$\mathcal{M}_n := \begin{cases} \dfrac{\sum_{j\in\mathcal{J}_n}\omega_j^{(n)}\left\|P_{\Pi^+_{j,n}}(u_n) - u_n\right\|^2}{\left\|\sum_{j\in\mathcal{J}_n}\omega_j^{(n)}P_{\Pi^+_{j,n}}(u_n) - u_n\right\|^2}, & \text{if } u_n \notin \bigcap\limits_{j\in\mathcal{J}_n}\Pi^+_{j,n}, \\ 1, & \text{otherwise.} \end{cases} \tag{8b}$$

Due to the convexity of $\|\cdot\|^2$, the parameter $\mathcal{M}_n \geq 1$, for all $n \in \mathbb{Z}_{\geq 0}$, so that μ_n can take values larger than or equal to 2. The parameters that can be preset by the designer are the concurrency index set $\mathcal{J}_n$ and μ_n. The bigger the cardinality of $\mathcal{J}_n$, the more closed half-spaces to be concurrently processed at the time instant n, which results into a potentially increased convergence speed. An example of $\mathcal{J}_n$, which will be followed in the numerical examples, can be found in (13). In the same fashion, for extrapolation parameter values μ_n close to $2\mathcal{M}_n$ ($\mu_n \leq 2\mathcal{M}_n$), increased convergence speed can be also observed (see Figure 6).

If we define

$$\beta_j^{(n)} := \omega_j^{(n)} y_j \frac{\left(\rho_j^{(n)} - y_j g_n(\mathbf{x}_j)\right)^+}{1 + \kappa(\mathbf{x}_j, \mathbf{x}_j)}, \quad \forall j \in \mathcal{J}_n, \ \forall n \in \mathbb{Z}_{\geq 0}, \tag{8c}$$

where $g_n := g_{f_n,b_n}$ by (6), then the algorithmic process (8a) can be written equivalently as follows:

$$(f_{n+1}, b_{n+1}) = \left(P_{B[0,\delta]}\left(f_n + \mu_n\sum_{j\in\mathcal{J}_n}\beta_j^{(n)}\kappa(\mathbf{x}_j, \cdot)\right), b_n + \mu_n\sum_{j\in\mathcal{J}_n}\beta_j^{(n)}\right), \quad \forall n \in \mathbb{Z}_{\geq 0}. \tag{8d}$$

The parameter $\mathcal{M}_n$ takes the following form after the proper algebraic manipulations:

$$\mathcal{M}_n := \begin{cases} \dfrac{\sum_{j\in\mathcal{J}_n}\omega_j^{(n)}\left(\left[\left(\rho_j^{(n)} - y_j g_n(\mathbf{x}_j)\right)^+\right]^2/(1+\kappa(\mathbf{x}_j,\mathbf{x}_j))\right)}{\sum_{i,j\in\mathcal{J}_n}\beta_i^{(n)}\beta_j^{(n)}(1+\kappa(\mathbf{x}_i,\mathbf{x}_j))}, \\ \qquad\qquad \text{if } u_n \notin \bigcap\limits_{j\in\mathcal{J}_n}\Pi^+_{j,n}, \\ 1, \qquad\qquad \text{otherwise.} \end{cases} \tag{8e}$$

As explained in [30], the introduction of the closed ball constraint $B[0,\delta]$ on the norm of the estimates $(f_n)_n$ results into a potential elimination of the coefficients γ_n that correspond to time instants close to index 0 in (1), so that a buffer with length N_b can be introduced to keep only the most recent N_b data $(\mathbf{x}_l)_{l=n-N_b+1}^n$. This introduces sparsification to the design. Since the complexity of all the metric projections in Algorithm 1 is linear, the overall complexity is linear on the number of the kernel function, or after inserting the buffer with length N_b, it is of order $\mathcal{O}(N_b)$.

4.1. Computation of the margin levels

We will now discuss in short the dynamic adjustment strategy of the margin parameters, introduced in [11, 30].

For simplicity, all the concurrently processed margins are assumed to be equal to each other, that is, $\rho_n := \rho_j^{(n)}$, for all $j \in \mathcal{J}_n$, for all $n \in \mathbb{Z}_{\geq 0}$. Of course, more elaborate schemes can be adopted.

Whenever $(\rho_n - y_j g_n(\mathbf{x}_j))^+ = 0$, the soft margin loss function $l_{\mathbf{x}_j, y_j, \rho_n}$ in (5) attains a global minimum, which means by Proposition 1 that $u_n := (f_n, b_n)$ belongs to $\Pi_{j,n}^+$. In this case, we say that we have *feasibility* for $j \in \mathcal{J}_n$. Otherwise, that is, if $u_n \notin \Pi_{j,n}^+$, *infeasibility* occurs. To describe such situations, let us denote the feasibility cases by the index set $\mathcal{J}_n' := \{j \in \mathcal{J}_n : (\rho_n - y_j g_n(\mathbf{x}_j))^+ = 0\}$. The infeasibility cases are obviously $\mathcal{J}_n \setminus \mathcal{J}_n'$.

If we set $\mathrm{card}(\varnothing) := 0$, then we define the *feasibility rate* as the quantity $R_{\mathrm{feas}}^{(n)} := \mathrm{card}(\mathcal{J}_n')/\mathrm{card}(\mathcal{J}_n)$, for all $n \in \mathbb{Z}_{\geq 0}$. For example, $R_{\mathrm{feas}}^{(n)} = 1/2$ denotes that the number of feasibility cases is equal to the number of infeasibility ones at the time instant $n \in \mathbb{Z}_{\geq 0}$.

If, at time n, $R_{\mathrm{feas}}^{(n)}$ is larger than or equal to some predefined R, we assume that this will also happen for the next time instant $n+1$, provided we work in a slowly changing environment. More than that, we expect $R_{\mathrm{feas}}^{(n+1)} \geq R$ to hold for a margin ρ_{n+1} slightly larger than ρ_n. Hence, at time n, if $R_{\mathrm{feas}}^{(n)} \geq R$, we set $\rho_{n+1} > \rho_n$ under some rule to be discussed below. On the contrary, if $R_{\mathrm{feas}}^{(n)} < R$, then we assume that if the margin parameter value is slightly decreased to $\rho_{n+1} < \rho_n$, it may be possible to have $R_{\mathrm{feas}}^{(n+1)} \geq R$. For example, if we set $R := 1/2$, this scheme aims at keeping the number of feasibility cases larger than or equal to those of infeasibilities, while at the same time it tries to push the margin parameter to larger values for better classification at the test phase.

In the design of [11, 30], the small variations of the parameters $(\rho_n)_{n \in \mathbb{Z}_{\geq 0}}$ are controlled by the linear parametric model $\nu_{\mathrm{APSM}}(\theta - \theta_0) + \rho_0$, $\theta \in \mathbb{R}$, where $\theta_0, \rho_0 \in \mathbb{R}$, $\rho_0 \geq 0$, are predefined parameters and ν_{APSM} is a sufficiently small positive slope (e.g., see Section 7). For example, in [30], $\rho_n := (\nu_{\mathrm{APSM}}(\theta_n - \theta_0) + \rho_0)^+$, where $\theta_{n+1} := \theta_n \pm \delta\theta$, for all n, and where the $\pm$ symbol refers to the dichotomy of either $R_{\mathrm{feas}}^{(n+1)} \geq R$ or $R_{\mathrm{feas}}^{(n+1)} < R$. In this way, an increase of θ by $\delta\theta > 0$ will increase ρ, whereas a decrease of θ by $-\delta\theta$ will force ρ to take smaller values. Of course, other models, other than this simple linear one, can also be adopted.

4.2. Kernel affine projection algorithm

Here we introduce a byproduct of Algorithm 1, namely, a kernelized version of the standard affine projection algorithm [15, 18, 19].

Motivated by the discussion in Section 3, Algorithm 1 was devised in order to find at each time instant n a point in the set of all desirable classifiers $\bigcap_{j \in \mathcal{J}_n} \Pi_{j,n}^+ \neq \varnothing$. Since any point in this intersection is suitable for the classification task at time n, any nonempty subset of $\bigcap_{j \in \mathcal{J}_n} \Pi_{j,n}^+$ can be used for the problem at hand. In what follows we see that if we limit the set of desirable classifiers and deal with the boundaries $\{\Pi_{j,n}\}_{j \in \mathcal{J}_n}$, that is, hyperplanes (Section 2.2), of the closed half-spaces $\{\Pi_{j,n}^+\}_{j \in \mathcal{J}_n}$, we end up with a kernelized version of the classical affine projection algorithm [18, 19].

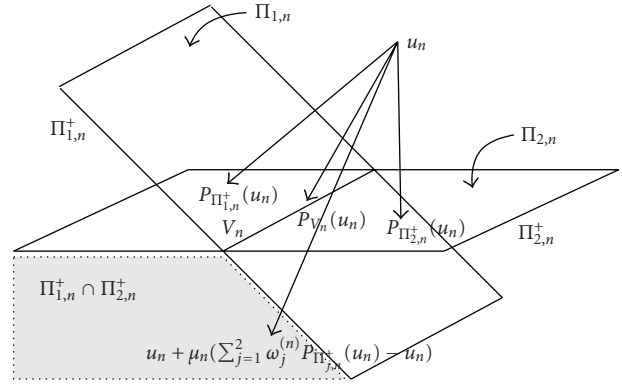

Figure 2: For simplicity, we assume that at some time instant $n \in \mathbb{Z}_{\geq 0}$, the cardinality $\mathrm{card}(\mathcal{J}_n) = 2$. This figure illustrates the closed half-spaces $\{\Pi_{j,n}^+\}_{j=1}^2$ and their boundaries, that is, the hyperplanes $\{\Pi_{j,n}\}_{j=1}^2$. In the case where $\bigcap_{j=1}^2 \Pi_{j,n} \neq \varnothing$, the defined in (11) linear variety becomes $V_n = \bigcap_{j=1}^2 \Pi_{j,n}$, which is a subset of $\bigcap_{j=1}^2 \Pi_{j,n}^+$. The kernel APA aims at finding a point in the linear variety V_n, while Algorithm 1 and the APSM consider the more general setting of finding a point in $\bigcap_{j=1}^2 \Pi_{j,n}^+$. Due to the range of the extrapolation parameter $\mu_n \in [0, 2\mathcal{M}_n]$ and $\mathcal{M}_n \geq 1$, the APSM can rapidly furnish solutions close to the large intersection of the closed half-spaces (see also Figure 6), without suffering from instabilities in the calculation of a Moore-Penrose pseudoinverse matrix necessary for finding the projection P_{V_n}.

Definition 1 (kernel affine projection algorithm). Fix $n \in \mathbb{Z}_{\geq 0}$ and let $q_n := \mathrm{card}(\mathcal{J}_n)$. Define the set of hyperplanes $\{\Pi_{j,n}\}_{j \in \mathcal{J}_n}$ by

$$\begin{aligned}\Pi_{j,n} &:= \left\{(\hat{f}, \hat{b}) \in \mathcal{H} \times \mathbb{R} : \langle (\hat{f}, \hat{b}), (y_j \kappa(\mathbf{x}_j, \cdot), y_j) \rangle_{\mathcal{H} \times \mathbb{R}} = \rho_j^{(n)} \right\} \\ &= \left\{\hat{u} \in \mathcal{H} \times \mathbb{R} : \langle \hat{u}, a_{j,n} \rangle_{\mathcal{H} \times \mathbb{R}} = \rho_j^{(n)} \right\}, \quad \forall j \in \mathcal{J}_n, \end{aligned} \tag{9}$$

where $a_{j,n} := y_j(\kappa(\mathbf{x}_j, \cdot), 1)$, for all $j \in \mathcal{J}_n$. These hyperplanes are the boundaries of the closed half-spaces $\{\Pi_{j,n}^+\}_{j \in \mathcal{J}_n}$ (see Figure 2). Note that such hyperplane constraints as in (9) are often met in regression problems with the difference that there the coefficients $\{\rho_j^{(n)}\}_{j \in \mathcal{J}_n}$ are part of the given data and not parameters as in the present classification task.

Since we will be looking for classifiers in the assumed nonempty intersection $\bigcap_{j \in \mathcal{J}_n} \Pi_{j,n}$, we define the function $\mathbf{e}_n : \mathcal{H} \times \mathbb{R} \to \mathbb{R}^{q_n}$ by

$$\mathbf{e}_n(u) := \begin{bmatrix} \rho_1^{(n)} - \langle a_{1,n}, u \rangle \\ \vdots \\ \rho_{q_n}^{(n)} - \langle a_{q_n,n}, u \rangle \end{bmatrix}, \quad \forall u \in \mathcal{H} \times \mathbb{R}, \tag{10}$$

and let the set (see Figure 2)

$$V_n := \arg\min_{u \in \mathcal{H} \times \mathbb{R}} \sum_{j=1}^{q_n} |\rho_j^{(n)} - \langle u, a_{j,n} \rangle|^2 = \arg\min_{u \in \mathcal{H} \times \mathbb{R}} \|\mathbf{e}_n(u)\|_{\mathbb{R}^{q_n}}^2. \tag{11}$$

This set is a linear variety (for a proof see Appendix A). Clearly, if $\bigcap_{j \in \mathcal{J}_n} \Pi_{j,n} \neq \varnothing$, then $V_n = \bigcap_{j \in \mathcal{J}_n} \Pi_{j,n}$. Now, given

an arbitrary initial u_0, the *kernel affine projection algorithm* is defined by the following point sequence:

$$\begin{aligned} u_{n+1} &:= u_n + \mu_n (P_{V_n}(u_n) - u_n) \\ &= u_n + \mu_n (a_{1,n}, \ldots, a_{q_n,n}) G_n^{\dagger} \mathbf{e}_n(u_n), \quad \forall n \in \mathbb{Z}_{\geq 0}, \end{aligned} \tag{12}$$

where the extrapolation parameter $\mu_n \in [0, 2]$, G_n is a matrix of dimension $q_n \times q_n$, where its (i, j)th element is defined by $y_i y_j (\kappa(\mathbf{x}_i, \mathbf{x}_j) + 1)$, for all $i, j \in \overline{1, q_n}$, the symbol $\dagger$ stands for the (Moore-Penrose) pseudoinverse operator [40], and the notation $(a_{1,n}, \ldots, a_{q_n,n})\boldsymbol{\lambda} := \sum_{j=1}^{q_n} \lambda_j a_{j,n}$, for all $\boldsymbol{\lambda} \in \mathbb{R}^{q_n}$. For the proof of the equality in (12), refer to Appendix A.

Remark 1. The fact that the classical (linear kernel) APA [18, 19] can be seen as a projection algorithm onto a sequence of linear varieties was also demonstrated in [26, Appendix B]. The proof in Appendix A extends the defining formula of the APA, and thus the proof given in [26, Appendix B], to infinite-dimensional Hilbert spaces. Extending [26], the APSM [12–14] devised a convexly constrained asymptotic minimization framework which contains APA, the NLMS, as well as a variety of recently developed projection-based algorithms [20–25, 27, 28].

By Definition 1 and Appendix A, at each time instant n, the kernel APA produces its estimate by projecting onto the linear variety V_n. In the special case where $q_n := 1$, that is, $\mathcal{J}_n = \{n\}$, for all n, then (12) gives the kernel NLMS [42]. Note also that in this case, the pseudoinverse is simplified to $G_n^{\dagger} = a_n / \|a_n\|^2$, for all n. Since V_n is a closed convex set, the kernel APA can be included in the wide frame of the APSM (see also the remarks just after Lemma 3.3 or Example 4.3 in [14]). Under the APSM frame, more directions become available for the kernel APA, not only in terms of theoretical properties, but also in devising variations and extensions of the kernel APA by considering more general convex constraints than V_n as in [26], and by incorporating a priori information about the model under study [14].

Note that in the case where $\bigcap_{j \in \mathcal{J}_n} \Pi_{j,n} \neq \varnothing$, then $V_n = \bigcap_{j \in \mathcal{J}_n} \Pi_{j,n}$. Since $\Pi_{j,n}$ is the boundary and thus a subset of the closed half-space $\Pi_{j,n}^{+}$, it is clear that looking for points in $\bigcap_{j \in \mathcal{J}_n} \Pi_{j,n}$, in the kernel APA and not in the larger $\bigcap_{j \in \mathcal{J}_n} \Pi_{j,n}^{+}$ as in Algorithm 1, limits our view of the online classification task (see Figure 2). Under mild conditions, Algorithm 1 produces a point sequence that enjoys properties like monotone approximation, strong convergence to a point in the intersection $\mathcal{K} \cap (\bigcap_{j \in \mathcal{J}_n} \Pi_{j,n}^{+})$, asymptotic optimality, as well as a characterization of the limit point.

To speed up convergence, Algorithm 1 offers the extrapolation parameter μ_n which has a range of $\mu_n \in [0, 2\mathcal{M}_n]$ with $\mathcal{M}_n \geq 1$. The calculation of the upper bound $\mathcal{M}_n$ is given by simple operations that do not suffer by instabilities as in the computation of the (Moore-Penrose) pseudoinverses $(G_n^{\dagger})_n$ in (12) [40]. A usual practice for the efficient computation of the pseudoinverse matrix is to diagonally load some matrix with positive values prior inversion, leading thus to solutions towards an approximation of the original problem at hand [15, 40].

The above-introduced kernel APA is based on the fundamental notion of metric projection mapping on linear varieties in a Hilbert space, and it can thus be straightforwardly extended to regression problems. In the sequel, we will focus on the more general view offered to classification by Algorithm 1 and not pursue further the kernel APA approach.

5. SPARSIFICATION BY A SEQUENCE OF FINITE-DIMENSIONAL SUBSPACES

In this section, sparsification is achieved by the construction of a sequence of linear subspaces $(M_n)_{n \in \mathbb{Z}_{\geq 0}}$, together with their bases $(\mathfrak{B}_n)_{n \in \mathbb{Z}_{\geq 0}}$, in the space $\mathcal{H}$. The present approach is in line with the rationale presented in [36], where a monotonically increasing sequence of subspaces $(M_n)_{n \in \mathbb{Z}_{\geq 0}}$ was constructed, that is, $M_n \subseteq M_{n+1}$, for all $n \in \mathbb{Z}_{\geq 0}$. Such a monotonic increase of the subspaces' dimension undoubtedly raises memory resources issues. In this paper, such a monotonicity restriction is not followed.

To accomodate memory limitations and tracking requirements, two parameters, namely L_b and α, will be of central importance in our design. The parameter L_b establishes a bound on the dimensions of $(M_n)_{n \in \mathbb{Z}_{\geq 0}}$, that is, if we define $L_n := \dim(M_n)$, then $L_n \leq L_b$, for all $n \in \mathbb{Z}_{\geq 0}$. Given a basis $\mathfrak{B}_n$, a buffer is needed in order to keep track of the L_n basis elements. The larger the dimension for the subspace M_n, the larger the buffer necessary for saving the basis elements. Here, L_b gives the designer the freedom to preset an upper bound for the dimensions $(L_n)_n$, and thus upper-bound the size of the buffer according to the available computational resources. Note that this introduces a tradeoff between memory savings and representation accuracy; the larger the buffer, the more basis elements to be used in the kernel expansion, and thus the larger the accuracy of the functional representation, or, in other words, the larger the span of the basis, which gives us more candidates for our classifier. We will see below that such a bound L_b results into a *sliding window* effect. Note also that if the data $\{\mathbf{x}_n\}_{n \in \mathbb{Z}_{\geq 0}}$ are drawn from a compact set in $\mathbb{R}^m$, then the algorithmic procedure introduced in [36] produces a sequence of monotonically increasing subspaces with dimensions upper-bounded by some bound not known a priori.

The parameter α is a measure of approximate linear dependency or independency. Every time a new element $\kappa(\mathbf{x}_{n+1}, \cdot)$ becomes available, we compare its distance from the available finite-dimensional linear subspace $M_n = \operatorname{span}(\mathfrak{B}_n)$ with α, where span stands for the linear span operation. If the distance is larger than α, then we say that $\kappa(\mathbf{x}_{n+1}, \cdot)$ is sufficiently linearly independent of the basis elements of $\mathfrak{B}_n$, we decide that it carries enough "new information," and we add this element to the basis, creating a new $\mathfrak{B}_{n+1}$ which clearly contains $\mathfrak{B}_n$. However, if the above distance is smaller than or equal to α, then we say that $\kappa(\mathbf{x}_{n+1}, \cdot)$ is approximately linearly dependent on the elements of $\mathfrak{B}_n$, so that augmenting $\mathfrak{B}_n$

is not needed. In other words, α controls the frequency by which new elements enter the basis. Obviously, the larger the α, the more "difficult" for a new element to contribute to the basis. Again, a tradeoff between the cardinality of the basis and the functional representation accuracy is introduced, as also seen above for the parameter L_b.

To increase the speed of convergence of the proposed algorithm, concurrent processing is introduced by means of the index set $\mathcal{J}_n$, which indicates which closed half-spaces will be processed at the time instant n. Note once again that such a processing is behind the increase of the convergence speed met in APA [18, 19] when compared to that of the NLMS [16, 17], in classical adaptive filtering [15]. Without any loss of generality, and in order to keep the discussion simple, we consider here the following simple case for $\mathcal{J}_n$:

$$\mathcal{J}_n := \begin{cases} \overline{0,n}, & \text{if } n < q-1, \\ \overline{n-q+1,n}, & \text{if } n \geq q-1, \end{cases} \quad \forall\, n \in \mathbb{Z}_{\geq 0}, \tag{13}$$

where $q \in \mathbb{Z}_{>0}$ is a predefined constant denoting the number of closed half-spaces to be processed at each time instant $n \geq q-1$. In other words, for $n \geq q-1$, at each time instant n, we consider concurrent projections on the closed half-spaces associated with the q most recent samples. We state now a definition whose motivation is the geometrical framework of the oblique projection mapping given in Figure 1.

Definition 2. Given $n \in \mathbb{Z}_{\geq 0}$, assume the finite-dimensional linear subspaces $M_n, M_{n+1} \subset \mathcal{H}$ with dimensions L_n and L_{n+1}, respectively. Then it is well known that there exists a linear subspace $\mathcal{W}_n$, such that $M_n + M_{n+1} = \mathcal{W}_n \oplus M_{n+1}$, where the symbol $\oplus$ stands for the direct sum [40, 41]. Then, the following mapping is defined:

$$\begin{aligned} &\pi_n : M_n + M_{n+1} \longrightarrow M_{n+1} \\ &\quad : f \longmapsto \pi_n(f) := \begin{cases} f, & \text{if } M_n \subseteq M_{n+1} \\ P_{M_{n+1},\mathcal{W}_n}(f), & \text{if } M_n \not\subseteq M_{n+1}, \end{cases} \end{aligned} \tag{14}$$

where $P_{M_{n+1},\mathcal{W}_n}$ denotes the oblique projection mapping on M_{n+1} along $\mathcal{W}_n$. To visualize this in the case when $M_n \not\subseteq M_{n+1}$, refer to Figure 1, where M becomes M_{n+1}, and M' becomes $\mathcal{W}_n$.

To exhibit the sparsification method, the constructive approach of mathematical induction on $n \in \mathbb{Z}_{\geq 0}$ is used as follows.

5.1. Initialization

Let us begin, now, with the construction of the bases $(\mathfrak{B}_n)_{n\in\mathbb{Z}_{\geq 0}}$ and the linear subspaces $(M_n)_{n\in\mathbb{Z}_{\geq 0}}$. At the starting time 0, our basis $\mathfrak{B}_0$ consists of only one vector $\psi_1^{(0)} := \kappa(\mathbf{x}_0, \cdot) \in \mathcal{H}$, that is, $\mathfrak{B}_0 := \{\psi_1^{(0)}\}$. This basis defines the linear subspace $M_0 := \operatorname{span}(\mathfrak{B}_0)$. The characterization of the element $\kappa(\mathbf{x}_0, \cdot)$ by the basis $\mathfrak{B}_0$ is obvious here: $\kappa(\mathbf{x}_0, \cdot) = 1 \cdot \psi_1^{(0)}$. Hence, we can associate to $\kappa(\mathbf{x}_0, \cdot)$ the one-dimensional vector $\boldsymbol{\theta}_{\mathbf{x}_0}^{(0)} := 1$, which completely describes $\kappa(\mathbf{x}_0, \cdot)$ by the basis $\mathfrak{B}_0$. Let also $K_0 := \kappa(\mathbf{x}_0, \mathbf{x}_0) > 0$, which guarantees the existence of the inverse $K_0^{-1} = 1/\kappa(\mathbf{x}_0, \mathbf{x}_0)$.

5.2. At the time instant $n \in \mathbb{Z}_{>0}$

We assume, now, that at time $n \in \mathbb{Z}_{>0}$ the basis $\mathfrak{B}_n = \{\psi_1^{(n)}, \ldots, \psi_{L_n}^{(n)}\}$ is available, where $L_n \in \mathbb{Z}_{>0}$. Define also the linear subspace $M_n := \operatorname{span}(\mathfrak{B}_n)$, which is of dimension L_n.

Without loss of generality, we assume that $n \geq q-1$, so that the index set $\mathcal{J}_n := \overline{n-q+1,n}$ is available. Available are also the kernel functions $\{\kappa(\mathbf{x}_j, \cdot)\}_{j\in\mathcal{J}_n}$. Our sparsification method is built on the sequence of closed linear subspaces $(M_n)_n$. At every time instant n, all the information needed for the realization of the sparsification method will be contained within M_n. As such, each $\kappa(\mathbf{x}_j, \cdot)$, for $j \in \mathcal{J}_n$, must be associated or approximated by a vector in M_n. Thus, we associate to each $\kappa(\mathbf{x}_j, \cdot)$, $j \in \mathcal{J}_n$, a set of vectors $\{\boldsymbol{\theta}_{\mathbf{x}_j}^{(n)}\}_{j\in\mathcal{J}_n}$, as follows

$$\kappa(\mathbf{x}_j, \cdot) \longmapsto k_{\mathbf{x}_j}^{(n)} := \sum_{l=1}^{L_n} \theta_{\mathbf{x}_j,l}^{(n)} \psi_l^{(n)} \in M_n, \quad \forall\, j \in \mathcal{J}_n. \tag{15}$$

For example, at time 0, $\kappa(\mathbf{x}_0, \cdot) \mapsto k_{\mathbf{x}_0}^{(0)} := \psi_1^{(0)}$. Since we follow the constructive approach of mathematical induction, the above set of vectors is assumed to be known.

Available is also the matrix $K_n \in \mathbb{R}^{L_n \times L_n}$ whose (i,j)th component is $(K_n)_{i,j} := \langle \psi_i^{(n)}, \psi_j^{(n)} \rangle$, for all $i, j \in \overline{1, L_n}$. It can be readily verified that K_n is a Gram matrix which, by the assumption that $\{\psi_l^{(n)}\}_{l=1}^{L_n}$ are linearly independent, is also positive definite [40, 41]. Hence, the existence of its inverse K_n^{-1} is guaranteed. We assume here that K_n^{-1} is also available.

5.3. At time $n+1$, the new data $\mathbf{x}_{n+1}$ becomes available

At time $n+1$, a new element $\kappa(\mathbf{x}_{n+1}, \cdot)$ of $\mathcal{H}$ becomes available. Since M_n is a closed linear subspace of $\mathcal{H}$, the orthogonal projection of $\kappa(\mathbf{x}_{n+1}, \cdot)$ onto M_n is well defined and given by

$$P_{M_n}(\kappa(\mathbf{x}_{n+1}, \cdot)) = \sum_{l=1}^{L_n} \zeta_{\mathbf{x}_{n+1},l}^{(n+1)} \psi_l^{(n)} \in M_n, \tag{16}$$

where the vector $\boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)} := [\zeta_{\mathbf{x}_{n+1},1}^{(n+1)}, \ldots, \zeta_{\mathbf{x}_{n+1},L_n}^{(n+1)}]^t \in \mathbb{R}^{L_n}$ satisfies the normal equations $K_n \boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)} = \mathbf{c}_{\mathbf{x}_{n+1}}^{(n+1)}$ with $\mathbf{c}_{\mathbf{x}_{n+1}}^{(n+1)}$ given by [37, 38]

$$\mathbf{c}_{\mathbf{x}_{n+1}}^{(n+1)} := \begin{bmatrix} \langle \kappa(\mathbf{x}_{n+1}, \cdot), \psi_1^{(n)} \rangle \\ \vdots \\ \langle \kappa(\mathbf{x}_{n+1}, \cdot), \psi_{L_n}^{(n)} \rangle \end{bmatrix} \in \mathbb{R}^{L_n}. \tag{17}$$

Since K_n^{-1} was assumed available, we can compute $\boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)}$ by

$$\boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)} = K_n^{-1} \mathbf{c}_{\mathbf{x}_{n+1}}^{(n+1)}. \tag{18}$$

Now, the distance d_{n+1} of $\kappa(\mathbf{x}_{n+1}, \cdot)$ from M_n (in Figure 1 this is the quantity $\|f - P_M(f)\|$) can be calculated as follows:

$$\begin{aligned} 0 \leq d_{n+1}^2 &:= \left\| \kappa(\mathbf{x}_{n+1}, \cdot) - P_{M_n}(\kappa(\mathbf{x}_{n+1}, \cdot)) \right\|^2 \\ &= \kappa(\mathbf{x}_{n+1}, \mathbf{x}_{n+1}) - \left(\mathbf{c}_{\mathbf{x}_{n+1}}^{(n+1)}\right)^t \boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)}. \end{aligned} \tag{19}$$

In order to derive (19), we used the fact that the linear operator P_{M_n} is selfadjoint and the linearity of the inner product $\langle \cdot, \cdot \rangle$ [37, 38]. Let us define now $\mathfrak{B}_{n+1} := \{\psi_l^{(n+1)}\}_{l=1}^{L_{n+1}}$.

5.3.1. *Approximate linear dependency* ($d_{n+1} \leq \alpha$)

If the metric distance of $\kappa(\mathbf{x}_{n+1}, \cdot)$ from M_n satisfies $d_{n+1} \leq \alpha$, then we say that $\kappa(\mathbf{x}_{n+1}, \cdot)$ is *approximately linearly dependent* on $\mathfrak{B}_n := \{\psi_l^{(n)}\}_{l=1}^{L_n}$, and that it is not necessary to insert $\kappa(\mathbf{x}_{n+1}, \cdot)$ into the new basis $\mathfrak{B}_{n+1}$. That is, we keep $\mathfrak{B}_{n+1} := \mathfrak{B}_n$, which clearly implies that $L_{n+1} := L_n$, and $\psi_l^{(n+1)} := \psi_l^{(n)}$, for all $l \in \overline{1, L_n}$. Moreover, $M_{n+1} := \operatorname{span}(\mathfrak{B}_{n+1}) = M_n$. Also, we let $K_{n+1} := K_n$, and $K_{n+1}^{-1} := K_n^{-1}$.

Notice here that $\mathcal{J}_{n+1} := \overline{n-q+2, n+1}$. The approximations given by (15) have to be transfered now to the new linear subspace M_{n+1}. To do so, we employ the mapping π_n given in Definition 2: for all $j \in \mathcal{J}_{n+1} \setminus \{n+1\}$, $k_{\mathbf{x}_j}^{(n+1)} := \pi_n(k_{\mathbf{x}_j}^{(n)})$. Since, $M_{n+1} = M_n$, then by (14),

$$k_{\mathbf{x}_j}^{(n+1)} := \pi_n(k_{\mathbf{x}_j}^{(n)}) = k_{\mathbf{x}_j}^{(n)}. \tag{20}$$

As a result, $\boldsymbol{\theta}_{\mathbf{x}_j}^{(n+1)} := \boldsymbol{\theta}_{\mathbf{x}_j}^{(n)}$, for all $j \in \mathcal{J}_n \setminus \{n+1\}$. As for $k_{\mathbf{x}_{n+1}}^{(n+1)}$, we use (16) and let $k_{\mathbf{x}_{n+1}}^{(n+1)} := P_{M_n}(\kappa(\mathbf{x}_{n+1}, \cdot))$. In other words, $\kappa(\mathbf{x}_{n+1}, \cdot)$ is approximated by its orthogonal projection $P_{M_n}(\kappa(\mathbf{x}_{n+1}, \cdot))$ onto M_n, and this information is kept in memory by the coefficient vector $\boldsymbol{\theta}_{\mathbf{x}_{n+1}}^{(n+1)} := \boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)}$.

5.3.2. *Approximate linear independency* ($d_{n+1} > \alpha$)

On the other hand, if $d_{n+1} > \alpha$, then $\kappa(\mathbf{x}_{n+1}, \cdot)$ becomes *approximately linearly independent* on $\mathfrak{B}_n$, and we add it to our new basis. If we also have $L_n \leq L_b - 1$, then we can increase the dimension of the basis without exceeding the memory of the buffer: $L_{n+1} := L_n + 1$ and $\mathfrak{B}_{n+1} := \mathfrak{B}_n \cup \{\kappa(\mathbf{x}_{n+1}, \cdot)\}$, such that the elements $\{\psi_l^{(n+1)}\}_{l=1}^{L_{n+1}}$ of $\mathfrak{B}_{n+1}$ become $\psi_l^{(n+1)} := \psi_l^{(n)}$, for all $l \in \overline{1, L_n}$, and $\psi_{L_{n+1}}^{(n+1)} := \kappa(\mathbf{x}_{n+1}, \cdot)$. We also update the Gram matrix by

$$K_{n+1} := \begin{bmatrix} K_n & \mathbf{c}_{\mathbf{x}_{n+1}}^{(n+1)} \\ (\mathbf{c}_{\mathbf{x}_{n+1}}^{(n+1)})^t & \kappa(\mathbf{x}_{n+1}, \mathbf{x}_{n+1}) \end{bmatrix} =: \begin{bmatrix} r_{n+1} & \mathbf{h}_{n+1}^t \\ \mathbf{h}_{n+1} & H_{n+1} \end{bmatrix}. \tag{21}$$

The fact $d_{n+1} > \alpha \geq 0$ guarantees that the vectors in $\mathfrak{B}_{n+1}$ are linearly independent. In this way the Gram matrix K_{n+1} is positive definite. It can be verified by simple algebraic manipulations that

$$K_{n+1}^{-1} = \begin{bmatrix} K_n^{-1} + \dfrac{\boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)} (\boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)})^t}{d_{n+1}^2} & -\dfrac{\boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)}}{d_{n+1}^2} \\ -\dfrac{(\boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)})^t}{d_{n+1}^2} & \dfrac{1}{d_{n+1}^2} \end{bmatrix} =: \begin{bmatrix} s_{n+1} & \mathbf{p}_{n+1}^t \\ \mathbf{p}_{n+1} & P_{n+1} \end{bmatrix}. \tag{22}$$

Since $\mathfrak{B}_n \subsetneq \mathfrak{B}_{n+1}$, we immediately obtain that $M_n \subsetneq M_{n+1}$. All the information given by (15) has to be translated now to the new linear subspace M_{n+1} by the mapping π_n as we did above in (20): $k_{\mathbf{x}_j}^{(n+1)} := \pi_n(k_{\mathbf{x}_j}^{(n)}) = k_{\mathbf{x}_j}^{(n)}$. Since the cardinality of $\mathfrak{B}_{n+1}$ is larger than the cardinality of $\mathfrak{B}_n$ by one, then $\boldsymbol{\theta}_{\mathbf{x}_j}^{(n+1)} = [(\boldsymbol{\theta}_{\mathbf{x}_j}^{(n)})^t, 0]^t$, for all $j \in \mathcal{J}_{n+1} \setminus \{n+1\}$. The new vector $\kappa(\mathbf{x}_{n+1}, \cdot)$, being a basis vector itself, satisfies $\kappa(\mathbf{x}_{n+1}, \cdot) \in M_{n+1}$, so that $k_{\mathbf{x}_{n+1}}^{(n+1)} := \kappa(\mathbf{x}_{n+1}, \cdot)$. Hence, it has the following representation with respect to the new basis $\mathfrak{B}_{n+1}$: $\boldsymbol{\theta}_{\mathbf{x}_{n+1}}^{(n+1)} := [\mathbf{0}^t, 1]^t \in \mathbb{R}^{L_{n+1}}$.

5.3.3. *Approximate linear independency* ($d_{n+1} > \alpha$) *and buffer overflow* ($L_n + 1 > L_b$)*; the sliding window effect*

Now, assume that $d_{n+1} > \alpha$ and that $L_n = L_b$. According to the above methodology, we still need to add $\kappa(\mathbf{x}_{n+1}, \cdot)$ to our new basis, but if we do so the cardinality $L_n + 1$ of this new basis will exceed our buffer's memory L_b. We choose here to discard the oldest element $\psi_1^{(n)}$ in order to make space for $\kappa(\mathbf{x}_{n+1}, \cdot)$: $\mathfrak{B}_{n+1} := (\mathfrak{B}_n \setminus \{\psi_1^{(n)}\}) \cup \{\kappa(\mathbf{x}_{n+1}, \cdot)\}$. This discard of $\psi_1^{(n)}$ and the addition of $\kappa(\mathbf{x}_{n+1}, \cdot)$ results in the sliding window effect. We stress here that instead of discarding $\psi_1^{(n)}$, other elements of $\mathfrak{B}_n$ can be removed, if we use different criteria than the present ones. Here, we choose $\psi_1^{(n)}$ for simplicity, and for allowing the algorithm to focus on recent system changes by making its dependence on the remote past diminishing as time moves on.

We define here $L_{n+1} := L_b$, such that the elements of $\mathfrak{B}_{n+1}$ become $\psi_l^{(n+1)} := \psi_{l+1}^{(n)}$, $l \in \overline{1, L_b - 1}$, and $\psi_{L_b}^{(n+1)} := \kappa(\mathbf{x}_{n+1}, \cdot)$. In this way, the update for the Gram matrix becomes $K_{n+1} := H_{n+1}$ by (21), where it can be verified that

$$K_{n+1}^{-1} = H_{n+1}^{-1} = P_{n+1} - \frac{1}{s_{n+1}} \mathbf{p}_{n+1} \mathbf{p}_{n+1}^t, \tag{23}$$

where P_{n+1} is defined by (22) (the proof of (23) is given in Appendix B).

Upon defining $M_{n+1} := \operatorname{span}(\mathfrak{B}_{n+1})$, it is easy to see that $M_n \not\subseteq M_{n+1}$. By the definition of the oblique projection, of the mapping π_n, and by $k_{\mathbf{x}_j}^{(n)} := \sum_{l=1}^{L_n} \theta_{\mathbf{x}_j,l}^{(n)} \psi_l^{(n)}$, for all $j \in \mathcal{J}_{n+1} \setminus \{n+1\}$, we obtain

$$\begin{aligned} k_{\mathbf{x}_j}^{(n+1)} := \pi_n(k_{\mathbf{x}_j}^{(n)}) &= \sum_{l=2}^{L_n} \theta_{\mathbf{x}_j,l}^{(n)} \psi_l^{(n)} + 0 \cdot \kappa(\mathbf{x}_{n+1}, \cdot) \\ &= \sum_{l=1}^{L_{n+1}} \theta_{\mathbf{x}_j,l}^{(n+1)} \psi_l^{(n+1)}, \quad \forall j \in \mathcal{J}_{n+1} \setminus \{n+1\}, \end{aligned} \tag{24}$$

where $\theta_{\mathbf{x}_j,l}^{(n+1)} := \theta_{\mathbf{x}_j,l+1}^{(n)}$, for all $l \in \overline{1, L_b - 1}$, and $\theta_{\mathbf{x}_j,L_b}^{(n+1)} := 0$, for all $j \in \mathcal{J}_{n+1} \setminus \{n+1\}$. Since $\kappa(\mathbf{x}_{n+1}, \cdot) \in M_{n+1}$, we set $k_{\mathbf{x}_{n+1}}^{(n+1)} := \kappa(\mathbf{x}_{n+1}, \cdot)$ with the following representation with respect to the new basis $\mathfrak{B}_{n+1}$: $\boldsymbol{\theta}_{\mathbf{x}_{n+1}}^{(n+1)} := [\mathbf{0}^t, 1]^t \in \mathbb{R}^{L_b}$. The sparsification scheme can be found in pseudocode format in Algorithm 2.

6. THE APSM WITH THE SUBSPACE-BASED SPARSIFICATION

In this section, we embed the sparsification strategy of Section 5 in the APSM. As a result, the following algorithmic procedure is obtained.

Subalgorithm

1. **Initialization**. Let $\mathfrak{B}_0 := \{\kappa(\mathbf{x}_0, \cdot)\}$, $K_0 := \kappa(\mathbf{x}_0, \mathbf{x}_0) > 0$, and $K_0^{-1} := 1/\kappa(\mathbf{x}_0, \mathbf{x}_0)$. Also, $\mathcal{J}_0 := \{0\}$, $\theta_{\mathbf{x}_0}^{(0)} := 1$, and $\widetilde{\gamma}_1^{(0)} := 0$. Fix $\alpha \geq 0$, and $L_b \in \mathbb{Z}_{>0}$.
2. Assume $n \in \mathbb{Z}_{>0}$. Available are $\mathfrak{B}_n$, $\{\boldsymbol{\theta}_{\mathbf{x}_j}^{(n)}\}_{j\in\mathcal{J}_n}$, where $\mathcal{J}_n := \overline{n-q+1, n}$, as well as $K_n \in \mathbb{R}^{L_n\times L_n}$, $K_n^{-1} \in \mathbb{R}^{L_n\times L_n}$, and the coefficients $\{\widetilde{\gamma}_l^{(n+1)}\}_{l=1}^{L_n}$ for the estimate in (26).
3. Time becomes $n+1$, and $\kappa(\mathbf{x}_{n+1}, \cdot)$ arrives. Notice that $\mathcal{J}_{n+1} := \overline{n-q+2, n+1}$.
4. Calculate $\mathbf{c}_{\mathbf{x}_{n+1}}^{(n+1)}$ and $\boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)}$ by (17) and (18), respectively, and the distance d_{n+1} by (19).
5. **if** $d_{n+1} \leq \alpha$ **then**
6. $L_{n+1} := L_n$.
7. Set $\mathfrak{B}_{n+1} := \mathfrak{B}_n$.
8. Let $\boldsymbol{\theta}_{\mathbf{x}_j}^{(n+1)} := \boldsymbol{\theta}_{\mathbf{x}_j}^{(n)}$, for all $j \in \mathcal{J}_{n+1} \setminus \{n+1\}$, and $\boldsymbol{\theta}_{\mathbf{x}_{n+1}}^{(n+1)} := \boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)}$.
9. $K_{n+1} := K_n$, and $K_{n+1}^{-1} := K_n^{-1}$.
10. Let $\{\widetilde{\gamma}_l^{(n+2)}\}_{l=1}^{L_{n+1}} := \{\widetilde{\gamma}_l^{(n+1)}\}_{l=1}^{L_n}$.
11. **else**
12. **if** $L_n \leq L_b - 1$ **then**
13. $L_{n+1} := L_n + 1$.
14. Set $\mathfrak{B}_{n+1} := \mathfrak{B}_n \cup \{\kappa(\mathbf{x}_{n+1}, \cdot)\}$.
15. Let $\boldsymbol{\theta}_{\mathbf{x}_j}^{(n+1)} := [(\boldsymbol{\theta}_{\mathbf{x}_j}^{(n)})^t, 0]^t$, for all $j \in \mathcal{J}_{n+1} \setminus \{n+1\}$, and $\boldsymbol{\theta}_{\mathbf{x}_{n+1}}^{(n+1)} := [\mathbf{0}^t, 1]^t \in \mathbb{R}^{L_n+1}$.
16. Define K_{n+1} and its inverse K_{n+1}^{-1} by (21) and (22), respectively.
17. $\widetilde{\gamma}_l^{(n+2)} := \widetilde{\gamma}_l^{(n+1)} + \widetilde{\mu}_{n+1}\sum_{j\in\mathcal{J}_{n+1}} \widetilde{\beta}_j^{(n+1)}\theta_{\mathbf{x}_j,l}^{(n+1)}$, for all $l \in \overline{1, L_{n+1}-1}$, and $\widetilde{\gamma}_{L_{n+1}}^{(n+2)} := \widetilde{\mu}_{n+1}\widetilde{\beta}_{n+1}^{(n+1)}\theta_{\mathbf{x}_{n+1},L_{n+1}}^{(n+1)}$.
18. **else if** $L_n = L_b$ **then**
19. $L_{n+1} := L_b$.
20. Let $\mathfrak{B}_{n+1} := (\mathfrak{B}_n \setminus \{\psi_1^{(n)}\}) \cup \{\kappa(\mathbf{x}_{n+1}, \cdot)\}$.
21. Set $\theta_{\mathbf{x}_j,l}^{(n+1)} = \theta_{\mathbf{x}_j,l+1}^{(n)}$, for all $l \in \overline{1, L_b - 1}$, and $\theta_{\mathbf{x}_j,L_b}^{(n+1)} := 0$, for all $j \in \mathcal{J}_{n+1} \setminus \{n+1\}$. Moreover, $\boldsymbol{\theta}_{\mathbf{x}_{n+1}}^{(n+1)} := [\mathbf{0}^t, 1]^t \in \mathbb{R}^{L_b}$.
22. Set $K_{n+1} := H_{n+1}$ by (21). Then, K_{n+1}^{-1} is given by (23).
23. $\widetilde{\gamma}_l^{(n+2)} := \widetilde{\gamma}_{l+1}^{(n+1)} + \widetilde{\mu}_{n+1}\sum_{j\in\mathcal{J}_{n+1}} \widetilde{\beta}_j^{(n+1)}\theta_{\mathbf{x}_j,l}^{(n+1)}$, for all $l \in \overline{1, L_{n+1}-1}$, and $\widetilde{\gamma}_{L_{n+1}}^{(n+2)} := \widetilde{\mu}_{n+1}\widetilde{\beta}_{n+1}^{(n+1)}\theta_{\mathbf{x}_{n+1},L_{n+1}}^{(n+1)}$.
24. **end**
25. Increase n by one, that is, $n \leftarrow n+1$ and go to line 2.

Algorithm 2: Sparsification scheme by a sequence of finite-dimensional linear subspaces.

Algorithm 3. For any $n \in \mathbb{Z}_{\geq 0}$, consider the index set $\mathcal{J}_n$ defined by (13). For any $j \in \mathcal{J}_n$ and for any $n \in \mathbb{Z}_{\geq 0}$, let the closed half-space $\Pi_{j,n}^+ := \{\hat{u} = (\hat{f}, \hat{b}) \in \mathcal{H}\times\mathbb{R} : y_j(\hat{f}(\mathbf{x}_j) + \hat{b}) \geq \rho_j^{(n)}\}$ and the weight $\omega_j^{(n)} \geq 0$ such that $\sum_{j\in\mathcal{J}_n}\omega_j^{(n)} = 1$. For an arbitrary initial offset $\widetilde{b}_0 \in \mathbb{R}$, consider as an initial classifier the point $\widetilde{u}_0 := (0, \widetilde{b}_0) \in \mathcal{H}\times\mathbb{R}$ and generate the following sequences by

$$\widetilde{f}_{n+1} := \pi_{n-1}(\widetilde{f}_n) + \widetilde{\mu}_n \sum_{j\in\mathcal{J}_n} \widetilde{\beta}_j^{(n)} k_{\mathbf{x}_j}^{(n)} \tag{25a}$$

$$= \pi_{n-1}(\widetilde{f}_n) + \sum_{l=1}^{L_n}\left(\widetilde{\mu}_n \sum_{j\in\mathcal{J}_n} \widetilde{\beta}_j^{(n)}\theta_{\mathbf{x}_j,l}^{(n)}\right)\psi_l^{(n)}, \quad \forall n \in \mathbb{Z}_{\geq 0}, \tag{25b}$$

where $\pi_{-1}(\widetilde{f}_0) := 0$, the vectors $\{\boldsymbol{\theta}_{\mathbf{x}_j}^{(n)}\}_{j\in\mathcal{J}_n}$, for all $n \in \mathbb{Z}_{\geq 0}$, are given by Algorithm 2, and

$$\widetilde{b}_{n+1} := \widetilde{b}_n + \widetilde{\mu}_n \sum_{j\in\mathcal{J}_n} \widetilde{\beta}_j^{(n)}, \quad \forall n \in \mathbb{Z}_{\geq 0}, \tag{25c}$$

where

$$\widetilde{\beta}_j^{(n)} := \omega_j^{(n)} y_j \frac{(\rho_n - y_j\widetilde{g}_n(\mathbf{x}_j))^+}{1 + \kappa(\mathbf{x}_j, \mathbf{x}_j)}, \quad \forall n \in \mathbb{Z}_{\geq 0}. \tag{25d}$$

The function $\widetilde{g}_n := g_{\widetilde{f}_n,\widetilde{b}_n}$, and g is defined by (6). Moreover ρ_n is given by the procedure described in Section 4.1. Also, $\widetilde{\mu}_n \in [0, 2\widetilde{\mathcal{M}}_n]$, where

$$\widetilde{\mathcal{M}}_n := \begin{cases} \dfrac{\sum_{j\in\mathcal{J}_n}\omega_j^{(n)}([(\rho_n - y_j\widetilde{g}_n(\mathbf{x}_j))^+]^2/(1+\kappa(\mathbf{x}_j,\mathbf{x}_j)))}{\sum_{i,j\in\mathcal{J}_n}\widetilde{\beta}_i^{(n)}\widetilde{\beta}_j^{(n)}(1+\kappa(\mathbf{x}_j,\mathbf{x}_j))}, \\ \qquad\qquad \text{if } \widetilde{u}_n := (\widetilde{f}_n, \widetilde{b}_n) \notin \bigcap_{j\in\mathcal{J}_n}\Pi_{j,n}^+, \\ 1, \qquad\qquad \text{otherwise}, \end{cases} \quad \forall n \in \mathbb{Z}_{\geq 0}. \tag{25e}$$

The following proposition holds.

Proposition 2. *Let the sequence of estimates $(\widetilde{f}_n)_{n\in\mathbb{Z}_{\geq 0}}$ obtained by Algorithm 3. Then, for all $n \in \mathbb{Z}_{\geq 0}$, there exists $(\widetilde{\gamma}_l^{(n)})_{l=1}^{L_{n-1}} \subset \mathbb{R}$ such that*

$$\widetilde{f}_n = \sum_{l=1}^{L_{n-1}} \widetilde{\gamma}_l^{(n)}\psi_l^{(n-1)} \in M_{n-1}, \quad \forall n \in \mathbb{Z}_{\geq 0}, \tag{26}$$

where $\mathfrak{B}_{-1} := \{0\}$, $M_{-1} := \{0\}$, and $L_{-1} := 1$.

Proof. See Appendix C. □

Now that we have a kernel series expression for the estimate $\widetilde{f}_n$ by (26), we can give also an expression for the quantity $\pi_{n-1}(\widetilde{f}_n)$ in (25b), by using also the definition (14):

$$\pi_{n-1}(\widetilde{f}_n) = \begin{cases} \widetilde{f}_n, & \text{if } M_{n-1} \subseteq M_n, \\ \sum_{l=2}^{L_{n-1}} \widetilde{\gamma}_l^{(n)}\psi_l^{(n-1)}, & \text{if } M_{n-1} \not\subseteq M_n. \end{cases} \tag{27}$$

That is, whenever $M_{n-1} \not\subseteq M_n$, we remove from the kernel series expansion (26) the term corresponding to the basis element $\psi_1^{(n-1)}$. This is due to the sliding window effect and

1. Initialization. Let $\mathfrak{B}_0 := \{\kappa(\mathbf{x}_0, \cdot)\}$, $\theta_{\mathbf{x}_0}^{(0)} := 1$, $\widetilde{\gamma}_1^{(0)} := 0$, $\mathcal{J}_0 := \{0\}$, and choose for the initial offset $\widetilde{b}_0$ any value in $\mathbb{R}$. Fix $\alpha \geq 0$ and $L_b \in \mathbb{Z}_{>0}$.

2. Assume the time instant $n \in \mathbb{Z}_{>0}$. Now, the index set $\mathcal{J}_n$ becomes $\mathcal{J}_n := \overline{n-q+1, n}$ by (13). We already know $\mathfrak{B}_{n-1}$, $\{\boldsymbol{\theta}_{\mathbf{x}_j}^{(n-1)}\}_{j\in\mathcal{J}_{n-1}}$, as well as $\{\widetilde{\gamma}_l^{(n)}\}_{l=1}^{L_{n-1}}$ and $\widetilde{b}_n$.

3. Calculate the new basis $\mathfrak{B}_n$ and the vectors $\{\boldsymbol{\theta}_{\mathbf{x}_j}^{(n)}\}_{j\in\mathcal{J}_n}$ by Algorithm 2.

4. Compute $\{\widetilde{\beta}_j^{(n)}\}_{j\in\mathcal{J}_n}$ by (25d).

5. Choose an extrapolation parameter value $\widetilde{\mu}_n$ from the interval $[0, 2\widetilde{\mathcal{M}}_n]$ where $\widetilde{\mathcal{M}}_n$ is computed by (25e).

6. Calculate the coefficients $\{\widetilde{\gamma}_l^{(n+1)}\}_{l=1}^{L_n}$ by (28).

7. The classifier $(\widetilde{f}_{n+1}, \widetilde{b}_{n+1})$ is given by (26) and (25c).

8. Increase n by one, that is, $n \leftarrow n+1$ and go to line 2.

Algorithm 3: Proposed algorithm.

refers to the case of Section 5.3.3. According to our strategy, the case $M_{n-1} \not\subseteq M_n$ happens only when approximate linear independency $d_n > \alpha$ and a buffer overflow $L_{n-1} + 1 > L_b$ occurs. To prevent this buffer overflow, we have to cut off the term corresponding to $\psi_1^{(n-1)}$, and keep an empty position in the buffer in order for the new element $\kappa(\mathbf{x}_n, \cdot)$ to contribute to the basis. Having the knowledge of (27), the coefficients $\{\widetilde{\gamma}_l^{(n)}\}_{l=1}^{L_{n-1}}$, for all $n \in \mathbb{Z}_{\geq 0}$, will be given by the following iterative formula: let $\widetilde{\gamma}_1^{(0)} := 0$, and for all $n \in \mathbb{Z}_{\geq 0}$,

$$\{\widetilde{\gamma}_l^{(n+1)}\}_{l=1}^{L_n} := \begin{cases} \widetilde{\gamma}_l^{(n)} + \widetilde{\mu}_n \sum_{j\in\mathcal{J}_n} \widetilde{\beta}_j^{(n)} \theta_{\mathbf{x}_j,l}^{(n)}, & \forall l \in \overline{1, L_n}, \\ \text{if } d_n \leq \alpha, \\ \begin{cases} \widetilde{\gamma}_l^{(n)} + \widetilde{\mu}_n \sum_{j\in\mathcal{J}_n} \widetilde{\beta}_j^{(n)} \theta_{\mathbf{x}_j,l}^{(n)}, & \forall l \in \overline{1, L_n - 1}, \\ \widetilde{\mu}_n \widetilde{\beta}_n^{(n)} \theta_{\mathbf{x}_n,L_n}^{(n)}, & l = L_n, \end{cases} \\ \qquad \text{if } d_n > \alpha,\ L_{n-1} + 1 \leq L_b, \\ \begin{cases} \widetilde{\gamma}_{l+1}^{(n)} + \widetilde{\mu}_n \sum_{j\in\mathcal{J}_n} \widetilde{\beta}_j^{(n)} \theta_{\mathbf{x}_j,l}^{(n)}, & \forall l \in \overline{1, L_n - 1}, \\ \widetilde{\mu}_n \widetilde{\beta}_n^{(n)} \theta_{\mathbf{x}_n,L_n}^{(n)}, & l = L_n, \end{cases} \\ \qquad \text{if } d_n > \alpha,\ L_{n-1} + 1 > L_b. \end{cases} \tag{28}$$

Our proposed algorithm is summarized as shown in Algorithm 3.

Notice that the calculation of all the metric and oblique projections is of linear complexity with respect to the dimension L_n. The main computational load of the proposed algorithm comes from the calculation of the orthogonal projection onto the subspace M_n by (18) which is of order $\mathcal{O}(L_n^2)$ where L_n is the dimension of M_n. Since, however, we have upper bounded $L_n \leq L_b$, for all $n \in \mathbb{Z}_{\geq 0}$, it follows that the computational load of our method is upper bounded by $\mathcal{O}(L_b^2)$.

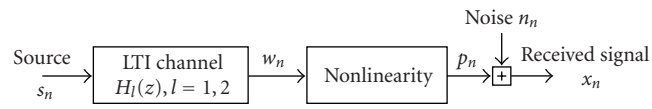

Figure 3: The model of the nonlinear channel for which adaptive equalization is needed.

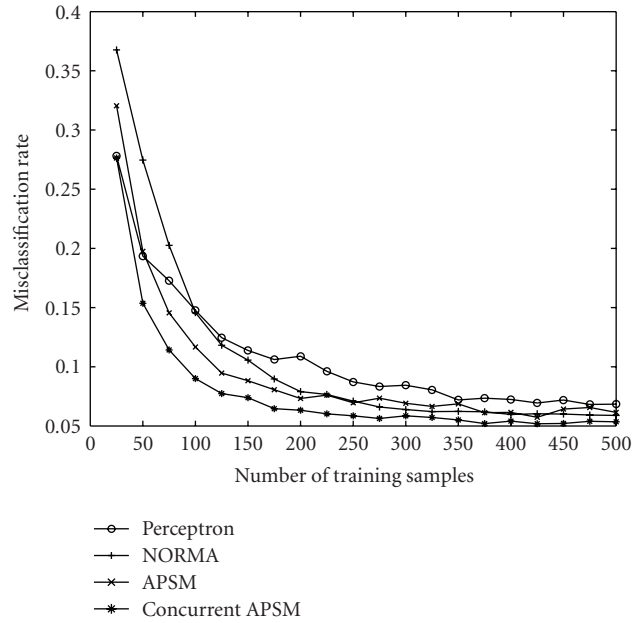

Figure 4: Tracking performance for the channel in Figure 3 where the LTI system is set to H_1. To allow concurrent processing, we let $q := \text{card}(\mathcal{J}_n) := 4$, for all n. The variance of the Gaussian kernel takes the value of $\sigma^2 := 0.5$. The buffer length $L_b := 500$, and $\alpha := 0.5$. The average number of basis elements is 110.

7. NUMERICAL EXAMPLES

An adaptive equalization problem for the nonlinear channel depicted in Figure 3 is chosen to validate the proposed design. The same model was chosen also in [11, 30]. The sparsification scheme of Section 5 was applied also to the stochastic gradient descent methods of NORMA and kernel perceptron [29].

The source signal $(s_n)_n$ is a sequence of numbers taking values from $\{\pm 1\}$ with equal probability. A linear time-invariant (LTI) [43] channel follows in order to produce the signal $(w_n)_n$. Available are two transfer functions for the LTI system: $H_l(z) := \sin(\theta_l)/\sqrt{2} + \cos(\theta_l)z^{-1} + (\sin(\theta_l)/\sqrt{2})z^{-2}$, for all $z \in \mathbb{C}$, $l = 1, 2$, where $\theta_1 := 29.5^\circ$ and $\theta_2 := -35^\circ$. In such a way, we can test our design under a sudden system change. The transfer functions $H_l(z) := \sum_{i=0}^{2} h_{li} z^{-i}$, $z \in \mathbb{C}$, $l = 1, 2$, were chosen as above in order to simplify computations, since $\sum_{i=0}^{2} h_{li}^2 = 1$, $l = 1, 2$. This choice comes from [5, equation (28)]. The nonlinearity in Figure 3 is given by $p_n := w_n + 0.2w_n^2 - 0.1w_n^3$, for all n, as in [5, equation (29)]. Gaussian i.i.d. noise $(n_n)_n$, with zero mean and SNR = 10 dB with respect to $(p_n)_n$, is added to give the received signal $(x_n)_n$.

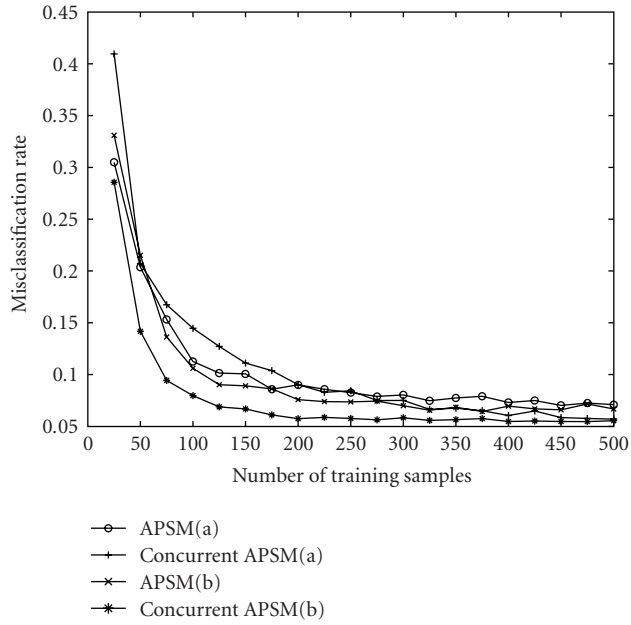

Figure 5: Tracking performance for the channel in Figure 3 when the LTI system is H_1. We let $\text{card}(\mathcal{J}_n) := 16$, for all n. The variance of the Gaussian kernel takes the value of $\sigma^2 := 0.5$. The APSM(a) refers to Algorithm 1 while APSM(b) refers to Algorithm 3. The radius of the closed ball is set to $\delta := 2$. The buffer length $L_b := 500$, and $\alpha := 0.5$.

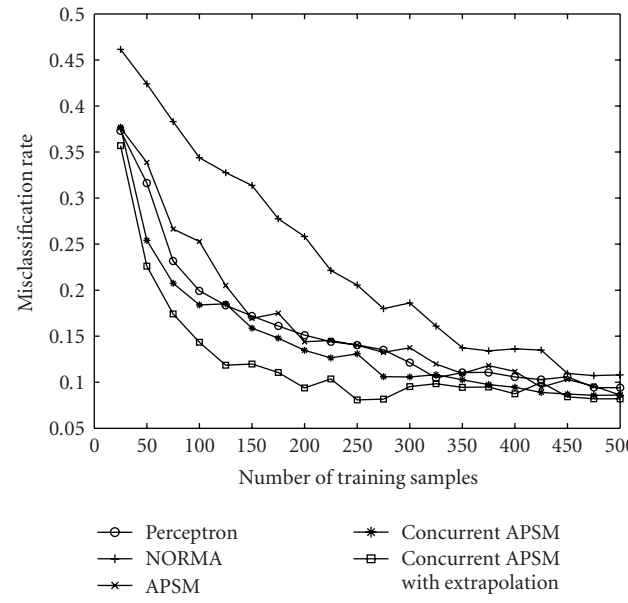

Figure 6: Here, the LTI system is again H_1, with $\text{card}(\mathcal{J}_n) := 8$, for all n. The variance of the Gaussian kernel takes the value of $\sigma^2 := 0.2$. The buffer length $L_b := 500$, and $\alpha := 0.5$. The extrapolation coefficient is $\widetilde{\mu}_n := 1.9\,\widetilde{\mathcal{M}}_n$, for all n.

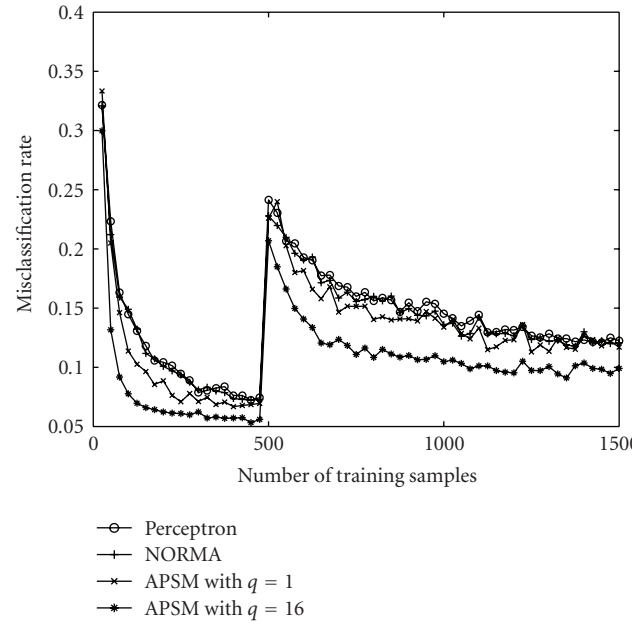

Figure 7: A channel switch occurs at time $n = 500$, from H_1 to H_2, for the LTI system in Figure 3. No sparsification for the APSMs, and no regularization for NORMA is considered here. The variance of the Gaussian kernel function is kept to the value of $\sigma^2 := 0.5$.

As in [11, 30], the data space is the Euclidean $\mathbb{R}^4$, and the data are formed as $\mathbf{x}_n := (x_n, x_{n-1}, x_{n-2}, x_{n-3})^t \in \mathbb{R}^4$, for all $n \in \mathbb{Z}_{\geq 0}$. The label y_n, at time instant n, is defined by the transmitted training symbol $s_{n-\tau}$, for all $n \in \mathbb{Z}_{\geq 0}$, where $\tau := 1$ [5]. The dimension of the data space and the parameter τ are the equalizer order and delay, respectively [5]. The Gaussian (RBF) kernel was used (cf. Section 2.1) in order to perform the classification task in an infinite dimensional RKHS $\mathcal{H}$ [1–3].

We compared the proposed methodology with the stochastic gradient descent method NORMA [29, Section III.A], which is a soft margin generalization of the classical kernel perceptron algorithm [29, Section VI.A]. The results are demonstrated in Figures 4, 5, 6, 7, and 8. The misclassification rate is defined as the ratio of the misclassifications (cf. Section 3) to the number of the test data, which are taken to be 100. A number of 100 experiments were performed and uniformly averaged to produce each curve in the figures.

In Figure 4, the transfer function of the LTI system in Figure 3 is set to $H_1(z)$, $z \in \mathbb{C}$. The variance σ^2 of the Gaussian kernel is set to $\sigma^2 := 0.5$. Recall here that the value of L_b is closely related to the available computational resources of our system (refer to Section 5). Here we choose the value $L_b = 500$, which was set to coincide with the time instant a sudden system change occurs in Figures 7 and 8. The same buffer with length L_b was also used for the NORMA and the kernel perceptron methods, with a learning rate of $\eta_n := 1/\sqrt{n}$, for all $n \in \mathbb{Z}_{>0}$, as suggested in [29]. The physical meaning of the parameter α is given in Section 5, where we have already seen that it defines a threshold for the distance of a point from a closed linear subspace. In the present numerical examples, we use RBF kernels, for which the length of every element $\kappa(\mathbf{x}_n, \cdot)$ is equal to 1 since $\|\kappa(\mathbf{x}, \cdot)\|^2 = \kappa(\mathbf{x}, \mathbf{x}) = 1$, for all $\mathbf{x} \in \mathbb{R}^m$. As such, for the following numerical examples, we let α take values less than or equal to 1. Here we set $\alpha := 0.5$.

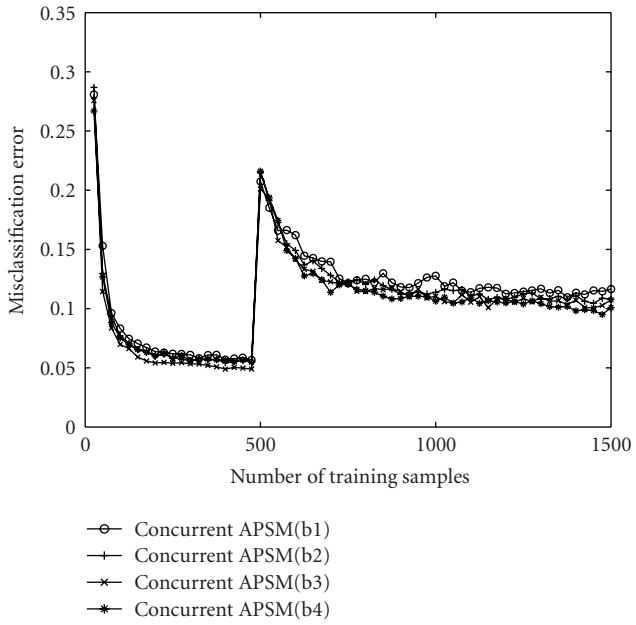

FIGURE 8: A channel switch occurs at time $n = 500$, from H_1 to H_2, for the LTI system in Figure 3. The variance of the Gaussian kernel function is $\sigma^2 := 0.5$. The parameter $q = 16$. These curves correspond to different values of the pair (α, L_b), and more specifically, "APSM(b1)" corresponds to $(0.9, 150)$, "APSM(b2)" to $(0.75, 200)$, "APSM(b3)" to $(0.5, 500)$, and "APSM(b4)" to $(0.1, 1000)$.

Depending on the application, and the sparsity the designer wants to impose on the system, different ranges for α are expected (see [36] and Figure 8). The parameter ν_{NORMA} which controls the soft margin adjustments of NORMA method is set to $\nu_{\text{NORMA}} := 0.01$, since it produced the best results after extensive experimentation. This value is also suggested in [29]. The APSM with $q = 1$ (no concurrent processing) and the APSM with $q = 4$ are employed here. Both the simple and the concurrent APSMs use the extrapolation parameter $\widetilde{\mu}_n := 1$, for all $n \in \mathbb{Z}_{\geq 0}$. For the parameters which control the margin (see Section 4.1), we let $\rho_0 := 1$, $\theta_0 := 1$. This choice of ρ_0 and θ_0 provides for the initial value of 1 for the margin in Section 4.1, which is also a typical initial value in online [29] and SVM [1] settings. We have seen, by extensive experimentation, that the best results were produced for a slowly changing sequence $(\rho_n)_n$. To guarantee such a behaviour, we assign small values to the step size $\delta\theta := 10^{-3}$ and to the slope $\nu_{\text{APSM}} := 10^{-1}$. We also let the threshold for the feasibility rate of Section 4.1 be $R := 1/2$. It can be verified by Figure 4 that both of the APSMs, that is, the nonconcurrent ($q = 1$) and the concurrent ($q = 4$), show faster convergence than the stochastic gradient descent methods of NORMA and kernel perceptron. Moreover, the concurrent APSM ($q = 4$) exhibits also a lower misclassification error level but with a computational cost of $q = 4$ times the cost of NORMA and of the kernel perceptron methods. Notice that the extrapolation parameter $\widetilde{\mu}_n$ was set to the value 1, that is, we did not take advantage of the freedom of choosing $\widetilde{\mu}_n \in [0, 2\widetilde{\mathcal{M}}_n]$ which necessitates, however, an additional computational complexity of order $\mathcal{O}(q^2)$ for the calculation of the parameter $\widetilde{\mathcal{M}}_n$ in (25e). The average number of the basis elements was found to be 110.

In Figure 5, we compare two different sparsification methods for the APSM: one presented in [30], that is, Algorithm 1 and denoted by APSM(a), and the other presented in Section 5 and denoted by APSM(b). The parameters for both methods were fixed in order to produce the same misclassification error level. For both realizations, the concurrent APSM used a $q = 16$ for the index set $\mathcal{J}_n$, $n \in \mathbb{Z}_{\geq 0}$. The variance of the Gaussian kernel is set to $\sigma^2 := 0.5$, the radius of the closed ball in (8a) to $\delta := 2$, the parameter $\alpha := 0.5$, and the buffer length $L_b := 500$. The buffer length N_b associated with the sparsification method APSM(a) (see the comments below Algorithm 1) was set to $N_b := 500$. We notice that the concurrent APSM(b) converges faster than the APSM(a). This is achieved, however, with an additional cost of order $\mathcal{O}(L_n^2)$ due to the operation (18). Even slower, the concurrent APSM(a) achieves the same misclassification error level as the concurrent APSM(b). Moreover, we do not notice such big differences between the nonconcurrent versions of the APSMs for both types of sparsification.

To exploit the extrapolation parameter $\widetilde{\mu}_n$ and its range $[0, 2\widetilde{\mathcal{M}}_n]$, we conducted the experiment depicted in Figure 6. The cardinality of the index set $\mathcal{J}_n$ was set to $q := 8$, and all the parameters regarding the APSMs, as well as the NORMA and the kernel perceptron method, are the same as in the previous figures, but the variance of the Gaussian kernel function was set to $\sigma^2 := 0.2$. The extrapolated version of the APSM uses a parameter value $\widetilde{\mu}_n := 1.9\,\widetilde{\mathcal{M}}_n$, for all $n \in \mathbb{Z}_{\geq 0}$. We observe that extrapolation indeed speeds up convergence, with an increased cost of order $\mathcal{O}(q^2)$ due to the necessary calculation of $\widetilde{\mathcal{M}}_n$ in (25e). It is also worth mentioning that the NORMA performs poorly, even compared to the kernel perceptron method for this RKHS $\mathcal{H}$.

To study the effect of the coefficient α together with the length L_b of the buffer, we refer to Figures 7 and 8, where a sudden channel change occurs, from the H_1 LTI system to the H_2 one, at the time instant 500. The coefficient α, in Figure 7, was set to 0, while we assume that the buffer length is infinite, that is, $L_b := \infty$. In both figures the variance of the Gaussian kernel is set to 0.5, and the parameter $q := 16$ for the concurrents APSMs, that is, for the cardinality of $\mathcal{J}_n$, for all $n \geq 16$ (see (13)). It is clear that the concurrent processing offered by the APSM remains by far the more robust approach since it achieves fast convergence as well as low misclassification rate level. In Figure 8, we examine the performance of the proposed sparsification scheme for various values of (α, L_b) and only for the concurrent version of the APSM. First, we notice that the introduction of sparsification in Figure 8 raises the misclassification rate level when compared with the design of unlimited computational resources, that is, $(\alpha, L_b) := (0, \infty)$ of Figure 7. In Figure 8, the pair (α, L_b) takes various values, so that "APSM(b1)" associates to the pair $(0.9, 150)$, "APSM(b2)" to $(0.75, 200)$, "APSM(b3)" to $(0.5, 500)$, and "APSM(b4)" to $(0.1, 1000)$. These values were chosen in order to produce the same

misclassification rate level for all the curves. This experiment shows a way to choose the values of (α, L_b), whenever a constraint is imposed on the length L_b of the buffer to be used. The more the buffer length is decreased, or in other words, the less the cardinality of the basis we want to build, and in order to keep the same misclassification rate level, the more the parameter α has to be increased in order for the new elements in the sequence $(\kappa(\mathbf{x}_n, \cdot))_n$ to enter the basis less frequently.

8. CONCLUSIONS

This paper presents a sparsification method to the online classification task, based on a sequence of linear subspaces and combined with the convex analytic approach of the adaptive projected subgradient method (APSM). Limitations on memory and computational resources, which are inherent in online systems, are accommodated by inserting an upper bound on the dimension of the sequence of the subspaces. The design obtains a geometric perspective by means of projection mappings. To validate the design, an adaptive equalization problem for a nonlinear channel is considered, and the proposed method was compared not only with classical and recent stochastic gradient descent methods, but also with a sparsified version of the APSM with a norm constraint.

APPENDICES

A. PROOF (I) OF V_n IS A LINEAR VARIETY AND (II) OF (12)

Fix $n \in \mathbb{Z}_{\geq 0}$ and define the mapping $A : \mathcal{H} \times \mathbb{R} \to \mathbb{R}^{q_n}$ by

$$A(u) := \begin{bmatrix} \langle a_{1,n}, u \rangle \\ \cdots \\ \langle a_{q_n,n}, u \rangle \end{bmatrix}, \quad \forall u \in \mathcal{H} \times \mathbb{R}. \tag{A.1}$$

The mapping A is clearly linear and also bounded [37, 38] since if we recall that the norm of A is $\|A\| := \sup_{\|u\| \leq 1} \|A(u)\|$, we can easily verify that

$$\begin{aligned} \|A(u)\|^2 &= \sum_{j=1}^{q_n} |\langle a_{j,n}, u \rangle|^2 \leq \sum_{j=1}^{q_n} \|a_{j,n}\|^2 \|u\|^2 \\ &\leq \sum_{j=1}^{q_n} \|a_{j,n}\|^2 < \infty, \end{aligned} \tag{A.2}$$

for all u such that $\|u\| \leq 1$. The adjoint operator $A^* : \mathbb{R}^{q_n} \to \mathcal{H} \times \mathbb{R}$ of A is then linear and bounded [38, Theorem 6.5.1]. To find its expression, we know by definition that $\boldsymbol{\lambda}^t A(u) = \langle u, A^*(\boldsymbol{\lambda}) \rangle$, for all $u \in \mathcal{H} \times \mathbb{R}$, for all $\boldsymbol{\lambda} \in \mathbb{R}^{q_n}$. Now, by simple algebraic manipulations, we obtain that

$$\sum_{j=1}^{q_n} \lambda_j \langle a_{j,n}, u \rangle = \langle u, A^*(\boldsymbol{\lambda}) \rangle \iff \left\langle u, A^*(\boldsymbol{\lambda}) - \sum_{j=1}^{q_n} \lambda_j a_{j,n} \right\rangle = 0, \quad \forall u \in \mathcal{H} \times \mathbb{R}, \ \forall \boldsymbol{\lambda} \in \mathbb{R}^{q_n}, \tag{A.3}$$

which suggests that

$$A^*(\boldsymbol{\lambda}) = \sum_{j=1}^{q_n} \lambda_j a_{j,n} =: (a_{1,n}, \ldots, a_{q_n,n})\boldsymbol{\lambda}. \tag{A.4}$$

The mapping AA^* is given clearly by $AA^*(\boldsymbol{\lambda}) = \begin{bmatrix} \langle a_{1,n}, A^*(\boldsymbol{\lambda}) \rangle \\ \cdots \\ \langle a_{q_n,n}, A^*(\boldsymbol{\lambda}) \rangle \end{bmatrix}$, for all $\boldsymbol{\lambda} \in \mathbb{R}^{q_n}$. Moreover, one can easily verify that for all $i \in \overline{1, q_n}$,

$$\langle a_{i,n}, A^*(\boldsymbol{\lambda}) \rangle = \left\langle a_{i,n}, \sum_{j=1}^{q_n} \lambda_j a_{j,n} \right\rangle = \sum_{j=1}^{q_n} \lambda_j \langle a_{i,n}, a_{j,n} \rangle, \tag{A.5}$$

so that we have $AA^*(\boldsymbol{\lambda}) = G_n \boldsymbol{\lambda}$, for all $\boldsymbol{\lambda} \in \mathbb{R}^{q_n}$, where the (i, j)th element of G_n is defined as $\langle a_{i,n}, a_{j,n} \rangle_{\mathcal{H} \times \mathbb{R}}$, for all $i, j \in \overline{1, q_n}$. Since $a_{j,n}$ was defined as $a_{j,n} := y_j(\kappa(\mathbf{x}_j, \cdot), 1)$, it can be easily seen by the inner product in $\mathcal{H} \times \mathbb{R}$ that $\langle a_{i,n}, a_{j,n} \rangle_{\mathcal{H} \times \mathbb{R}} = y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j) + y_i y_j$, for all $i, j \in \overline{1, q_n}$. As a result, $AA^* = G_n$.

Now, by A the set V_n obtains an alternative expression; $V_n = \arg\min_{u \in \mathcal{H} \times \mathbb{R}} \|\boldsymbol{\rho}^{(n)} - A(u)\|$, where $\boldsymbol{\rho}^{(n)} := [\rho_1^{(n)}, \ldots, \rho_{q_n}^{(n)}]^t$. By this new expression of V_n, we see by [38, Theorem 6.9.1] that V_n is the set of all those elements that satisfy the equations $V_n = \{A^*A(u) = A^*(\boldsymbol{\rho}^{(n)})\}$. Hence, V_n is a linear variety, that is, a closed convex set. Define, now, the translation of V_n by $-u_n$, that is, $V_n' := V_n - u_n := \{u - u_n : u \in V_n\}$. Clearly, V_n' is also a linear variety. By the linearity of A^*, we obtain $V_n' = \{u' \in \mathcal{H} \times \mathbb{R} : A^*A(u') = A^*(\boldsymbol{\rho}^{(n)} - A(u_n)) = A^*(\mathbf{e}_n(u_n))\}$. Thus, by [38, Theorem 6.9.1], $V_n' = \arg\min_{u' \in \mathcal{H} \times \mathbb{R}} \|\mathbf{e}_n(u_n) - A(u')\|$.

By the definition of the pseudoinverse operator [38, Section 6.11], the unique element of V_n' with the smallest norm is given by $u_*' := A^\dagger(\mathbf{e}_n(u_n))$, where $A^\dagger$ is the pseudoinverse operator of A [38]. Thus,

$$\|P_{V_n}(u_n) - u_n\| = \min_{u \in V_n} \|u - u_n\| = \min_{u' \in V_n'} \|u'\| = \|u_*'\|, \tag{A.6}$$

and by the uniqueness of $P_{V_n}(u_n)$, we obtain $P_{V_n}(u_n) - u_n = u_*' = A^\dagger(\mathbf{e}_n(u_n))$.

Now, by [38, Proposition 6.11.1.9], $A^\dagger = A^*(AA^*)^\dagger = A^* G_n^\dagger$. Thus, by (A.4), $u_*' = A^\dagger(\mathbf{e}_n(u_n)) = A^* G_n^\dagger(\mathbf{e}_n(u_n)) = (a_{1,n}, \ldots, a_{q_n,n}) G_n^\dagger(\mathbf{e}_n(u_n))$, which completes the proof of (12).

B. PROOF OF (23)

Since $K_{n+1} K_{n+1}^{-1} = I_{L_{n+1}}$, by multiplying (21) with (22) we obtain the following two equations:

$$\mathbf{h}_{n+1} \mathbf{p}_{n+1}^t + H_{n+1} P_{n+1} = I_{L_{n+1}-1}, \tag{B.1}$$

$$s_{n+1} \mathbf{h}_{n+1} + H_{n+1} \mathbf{p}_{n+1} = \mathbf{0}, \tag{B.2}$$

where I_m stands for the identity matrix of dimension $m \in \mathbb{Z}_{>0}$. Notice that since both K_{n+1} and K_{n+1}^{-1} are positive definite, we obtain that $s_{n+1} > 0$ and that H_{n+1} is positive definite [41]. Hence, H_{n+1}^{-1} exists. If we multiply (B.1) on

the left-hand side by H_{n+1}^{-1}, we obtain $H_{n+1}^{-1} = P_{n+1} + H_{n+1}^{-1}\mathbf{h}_{n+1}\mathbf{p}_{n+1}^{t}$. Moreover, a multiplication of (B.2) by H_{n+1}^{-1} on the left-hand side results in $H_{n+1}^{-1}\mathbf{h}_{n+1} = -(1/s_{n+1})\mathbf{p}_{n+1}$. By combining the last two results, the desired (23) is obtained.

C. PROOF OF PROPOSITION 2

We will prove Proposition 2 by mathematical induction on $n \in \mathbb{Z}_{\geq 0}$. Since by definition $\widetilde{f}_0 := 0$, we have $\widetilde{f}_0 = \sum_{l=1}^{L_{-1}=1} 0 \cdot \psi_l^{(-1)} = 0 \in M_{-1}$. Assume, now, that $\widetilde{f}_n = \sum_{l=1}^{L_{n-1}} \widetilde{\gamma}_l^{(n)} \psi_l^{(n-1)} \in M_{n-1}$. By the definition of the mapping π_n in (14), we see that $\pi_{n-1}(\widetilde{f}_n) \in M_n$, which means that there exists a set of real numbers $\{\eta_1^{(n)}, \ldots, \eta_{L_n}^{(n)}\}$ such that $\pi_{n-1}(\widetilde{f}_n) = \sum_{l=1}^{L_n} \eta_l^{(n)} \psi_l^{(n)}$. Now, by (25b) define

$$\widetilde{\gamma}_l^{(n+1)} := \eta_l^{(n)} + \widetilde{\mu}_n \sum_{j \in \mathcal{J}_n} \widetilde{\beta}_j^{(n)} \theta_{\mathbf{x}_j,l}^{(n)}, \tag{C.1}$$

to establish the relation given in Proposition 2. Since $\{\psi_l^{(n)}\}_{l=1}^{L_n} \subset M_n$, we easily have by $\widetilde{f}_{n+1} = \sum_{l=1}^{L_n} \widetilde{\gamma}_l^{(n+1)} \psi_l^{(n)}$ that $\widetilde{f}_n \in M_n$. This completes the proof of Proposition 2.

MAIN NOTATIONS

$\mathcal{H}$, $\langle\cdot,\cdot\rangle$, and $\|\cdot\|$:	The reproducing kernel Hilbert space (RKHS), its inner product, and its norm
f:	An element of $\mathcal{H}$
$\kappa(\cdot,\cdot)$:	The kernel function
$(\mathbf{x}_n, y_n)_{n\in\mathbb{Z}_{\geq 0}}$:	Sequence of data and labels
P_C:	Metric projection mapping onto the closed convex set C
$P_{M,M'}$:	Oblique projection on the subspace M along the subspace M'
$g(\cdot) = f(\cdot) + b$:	The classifier given by means of $f \in \mathcal{H}$ and the offset b
$\overline{j_1, j_2} :=$ $\{j_1, j_1+1, \ldots, j_2\}$:	An index set of consecutive integers
$\mathcal{J}_n$:	The index set which shows which closed half-spaces are concurrently processed at each time instant n
$\Pi_{j,n}^{+}$:	The closed half-spaces to be concurrently processed
$(\mathbf{x}_j, y_j, \rho_j^{(n)})$:	The triplet of data, labels, and margin parameters that define $\Pi_{j,n}^{+}$
μ_n and $\widetilde{\mu}_n$:	Extrapolation parameters with ranges $\mu_n \in [0, 2\mathcal{M}_n]$ and $\widetilde{\mu}_n \in [0, 2\widetilde{\mathcal{M}}_n]$, where $\mathcal{M}_n$ and $\widetilde{\mathcal{M}}_n$ are given by (8e) and (25e), respectively
ν_{APSM}, θ_0, $\delta\theta$, ρ_0:	Parameters that control the margins in Section 4.1
M_n, $\mathfrak{B}_n$, and L_n:	A subspace, its base, and its dimension, used for sparsification
$\mathfrak{B}_n = \{\psi_l^{(n)}\}_{l=1}^{L_n}$:	The basis elements of the basis $\mathfrak{B}_n$
π_n:	The mapping defined by (14)
$k_{\mathbf{x}_j}^{(n)}$ and $\boldsymbol{\theta}_{\mathbf{x}_j}^{(n)}$:	An element of M_n and its coefficient vector, which approximate the point $\kappa(\mathbf{x}_j, \cdot)$ by (15)
K_n:	The Gram matrix formed by the elements of the basis $\mathfrak{B}_n$
$\boldsymbol{\zeta}_{\mathbf{x}_{n+1}}^{(n+1)}$ and $\mathbf{c}_{\mathbf{x}_{n+1}}^{(n+1)}$:	The coefficient vector of the projection $P_{M_n}(\kappa(\mathbf{x}_{n+1}, \cdot))$ onto M_n and the coefficient vector in the normal equations of (18)
d_{n+1}:	The distance of $\kappa(\mathbf{x}_{n+1}, \cdot)$ from M_n defined in (19)
α and L_b:	The threshold of approximate linear dependency/independency and the length of the buffer (upper bound for L_n) used for the kernel expansion in (26)
r_{n+1}, $\mathbf{h}_{n+1}$, H_{n+1}, and s_{n+1}, $\mathbf{p}_{n+1}$, P_{n+1}:	Auxiliary quantities defined in (21) and (22), respectively
$\{\widetilde{\gamma}_l^{(n)}\}_{l=1}^{L_{n-1}}$:	Coefficients for the kernel expansion in (26)

ACKNOWLEDGMENTS

This study was conducted during K. Slavakis' stay at the University of Athens, Department of Informatics and Telecommunications. This research project (ENTER) was co-financed by the EU-European Social Fund (75%) and the Greek Ministry of Development-GSRT (25%).

REFERENCES

[1] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, Academic Press, Amsterdam, The Netherlands, 3rd edition, 2006.

[2] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, New York, NY, USA, 2004.

[3] B. Schölkopf and A. J. Smola, *Learning with Kernels*, MIT Press, Cambridge, Mass, USA, 2001.

[4] F. Pérez-Cruz and O. Bousquet, "Kernel methods and their potential use in signal processing," *IEEE Signal Processing Magazine*, vol. 21, no. 3, pp. 57–65, 2004.

[5] S. Chen, B. Mulgrew, and P. M. Grant, "A clustering technique for digital communications channel equalization using radial basis function networks," *IEEE Transactions on Neural Networks*, vol. 4, no. 4, pp. 570–579, 1993.

[6] E. Parzen, "Probability density functionals and reproducing kernel Hilbert spaces," in *Proceedings of the Symposium on Time Series Analysis*, pp. 155–169, John Wiley & Sons, New York, NY, USA, 1963.

[7] G. Wahba, "Multivariate function and operator estimation based on smoothing splines and reproducing kernels," in *Nonlinear Modeling and Forecasting*, M. Casdagli, S. Eubank, et al., Eds., vol. 12 of *SFI Studies in the Sciences of Complexity*, pp. 95–112, Addison-Wesley, Reading, Mass, USA, 1992.

[8] V. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, New York, NY, USA, 1998.

[9] N. Aronszajn, "Theory of reproducing kernels," *Transactions on American Mathematical Society*, vol. 68, no. 3, pp. 337–404, 1950.

[10] J. Mercer, "Functions of positive and negative type and their connection with the theory of integral equations," *Philosophical Transactions of the Royal Society of London, Series A*, vol. 209, pp. 415–446, 1909.

[11] K. Slavakis, S. Theodoridis, and I. Yamada, "Online kernel-based classification by projections," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '07)*, vol. 2, pp. 425–428, Honolulu, Hawaii, USA, April 2007.

[12] I. Yamada, "Adaptive projected subgradient method: a unified view for projection based adaptive algorithms," *Journal of the Institute of Electronics, Information and Communication Engineers*, vol. 86, no. 8, pp. 654–658, 2003, (Japanese).

[13] 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.

[14] 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.

[15] A. H. Sayed, *Fundamentals of Adaptive Filtering*, John Wiley & Sons, Hoboken, NJ, USA, 2003.

[16] J. Nagumo and J. Noda, "A learning method for system identification," *IEEE Transactions on Automatic Control*, vol. 12, no. 3, pp. 282–287, 1967.

[17] A. E. Albert and L. A. Gardner, *Stochastic Approximation and Nonlinear Regression*, MIT Press, Cambridge, Mass, USA, 1967.

[18] T. Hinamoto and S. Maekawa, "Extended theory of learning identification," *Electrical Engineering in Japan*, vol. 95, no. 5, pp. 101–107, 1975, (Japanese).

[19] K. Ozeki and T. Umeda, "An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties," *Electronics & Communications in Japan*, vol. 67 A, no. 5, pp. 19–27, 1984, (Japanese).

[20] S. C. Park and J. F. Doherty, "Generalized projection algorithm for blind interference suppression in DS/CDMA communications," *IEEE Transactions on Circuits and Systems II*, vol. 44, no. 6, pp. 453–460, 1997.

[21] M. L. R. de Campos, S. Werner, and J. A. Apolinário Jr., "Constrained adaptation algorithms employing householder transformation," *IEEE Transactions on Signal Processing*, vol. 50, no. 9, pp. 2187–2195, 2002.

[22] S. Werner and P. S. R. Diniz, "Set-membership affine projection algorithm," *IEEE Signal Processing Letters*, vol. 8, no. 8, pp. 231–235, 2001.

[23] S. Werner, J. A. Apolinário Jr., M. L. R. de Campos, and P. S. R. Diniz, "Low-complexity constrained affine-projection algorithms," *IEEE Transactions on Signal Processing*, vol. 53, no. 12, pp. 4545–4555, 2005.

[24] S. Gollamudi, S. Nagaraj, S. Kapoor, and Y.-F. Huang, "Set-membership filtering and a set-membership normalized LMS algorithm with an adaptive step size," *IEEE Signal Processing Letters*, vol. 5, no. 5, pp. 111–114, 1998.

[25] L. Guo, A. Ekpenyong, and Y.-F. Huang, "Frequency-domain adaptive filtering: a set-membership approach," in *Proceedings of the 37th Asilomar Conference on Signals, Systems and Computers (ACSSC '03)*, vol. 2, pp. 2073–2077, Pacific Grove, Calif, USA, November 2003.

[26] I. Yamada, K. Slavakis, and K. Yamada, "An efficient robust adaptive filtering algorithm based on parallel subgradient projection techniques," *IEEE Transactions on Signal Processing*, vol. 50, no. 5, pp. 1091–1101, 2002.

[27] M. Yukawa, K. Slavakis, and I. Yamada, "Adaptive parallel quadratic-metric projection algorithms," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 5, pp. 1665–1680, 2007.

[28] M. Yukawa and I. Yamada, "Pairwise optimal weight realization—acceleration technique for set-theoretic adaptive parallel subgradient projection algorithm," *IEEE Transactions on Signal Processing*, vol. 54, no. 12, pp. 4557–4571, 2006.

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

[30] K. Slavakis, S. Theodoridis, and I. Yamada, "Online sparse kernel-based classification by projections," in *Proceedings of the IEEE Workshop on Machine Learning for Signal Processing (MLSP '07)*, pp. 294–299, Thessaloniki, Greece, August 2007.

[31] L. Hoegaerts, "Eigenspace methods and subset selection in kernel based learning," Ph.D. dissertation, Katholieke Universiteit Leuven, Leuven, Belgium, 2005.

[32] J. A. K. Suykens, J. de Brabanter, L. Lukas, and J. Vandewalle, "Weighted least squares support vector machines: robustness and sparce approximation," *Neurocomputing*, vol. 48, no. 1–4, pp. 85–105, 2002.

[33] B. J. de Kruif and T. J. A. de Vries, "Pruning error minimization in least squares support vector machines," *IEEE Transactions on Neural Networks*, vol. 14, no. 3, pp. 696–702, 2003.

[34] B. Mitchinson, T. J. Dodd, and R. F. Harrison, "Reduction of kernel models," Tech. Rep. 836, University of Sheffield, Sheffield, UK, 2003.

[35] S. van Vaerenbergh, J. Vía, and I. Santamaría, "A sliding-window kernel RLS algorithm and its application to nonlinear channel identification," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '06)*, vol. 5, pp. 789–792, Toulouse, France, May 2006.

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

[37] F. Deutsch, *Best Approximation in Inner Product Spaces*, Springer, New York, NY, USA, 2001.

[38] D. G. Luenberger, *Optimization by Vector Space Methods*, John Wiley & Sons, New York, NY, USA, 1969.

[39] H. H. Bauschke and J. M. Borwein, "On projection algorithms for solving convex feasibility problems," *SIAM Review*, vol. 38, no. 3, pp. 367–426, 1996.

[40] A. Ben-Israel and T. N. E. Greville, *Generalized Inverses: Theory and Applications*, Springer, New York, NY, USA, 2nd edition, 2003.

[41] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, New York, NY, USA, 1985.

[42] A. V. Malipatil, Y.-F. Huang, S. Andra, and K. Bennett, "Kernelized set-membership approach to nonlinear adaptive filtering," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '05)*, vol. 4, pp. 149–152, Philadelphia, Pa, USA, March 2005.

[43] N. K. Bose, *Digital Filters: Theory and Applications*, Krieger, Malabar, Fla, USA, 1993.