INTERNATIONAL JOURNAL OF ADAPTIVE CONTROL AND SIGNAL PROCESSING

Int. J. Adapt. Control Signal Process. 2015; **00**:2–28

Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/acs

Nonlinear Adaptive Filtering using Kernel-based Algorithms with Dictionary Adaptation

Chafic SAIDE[1], Régis LENGELLE[1], Paul HONEINE[1*], Cédric RICHARD[2], Roger ACHKAR[3]

[1]*Institut Charles Delaunay (CNRS), Université de technologie de Troyes, Troyes, France.*

[2]*Laboratoire Lagrange (CNRS), Observatoire de la côte d'Azur. Université de Nice Sophia-Antipolis, Nice, France.*

[3]*American University of Science and Technology (AUST), Beirut, Lebanon.*

SUMMARY

Nonlinear adaptive filtering has been extensively studied in the literature, using for example Volterra filters or Neural Networks. Recently, kernel methods have been offering an interesting alternative since they provide a simple extension of linear algorithms to the nonlinear case. The main drawback of online system identification with kernel methods is that the filter complexity increases with time, a limitation resulting from the representer theorem which states that all past input vectors are required. To overcome this drawback, a particular subset of these input vectors (called dictionary) must be selected to ensure complexity control and good performance. Up to now, all authors considered that, after being introduced into the dictionary, elements stay unchanged even if, due to nonstationarity, they become useless to predict the system output. The objective of this paper is to present an adaptation scheme of dictionary elements, which are considered here as adjustable model parameters, by deriving a gradient-based method under collinearity constraints. The main interest is to ensure a better tracking performance. To evaluate our approach, dictionary adaptation is introduced into three well known kernel-based adaptive algorithms: Kernel Recursive Least Squares, Kernel Normalized Least Mean Squares and Kernel Affine Projection. The performance is evaluated on nonlinear adaptive filtering of simulated and real data sets. As confirmed by experiments, our dictionary adaptation scheme allows either complexity reduction or a decrease of the instantaneous quadratic error, or both simultaneously.

Received . . .

KEY WORDS: Adaptive filters; machine learning; nonlinear systems; online system identification; kernel methods.

Prepared using acsauth.cls *[Version: 2010/03/27 v2.00]*

1. INTRODUCTION

Nonlinear system modeling and identification have been widely studied in the last decades [1, 2], using for example Volterra filters [3–5] and neural networks [6]. More recently, function approximation based on Reproducing Kernel Hilbert Spaces (RKHS) defined kernel-based regression methods such as support vector regression [7–9]. Kernel-based regression generally extends standard methods used in the linear case by replacing the computation of some inner products by a Mercer kernel [10]. This provides an efficient way to implicitly map the data into a high, even infinite, dimensional RKHS and apply the original algorithm in this space without making direct reference to the nonlinear mapping of input vectors [11, 12]. Kernel-based regression requires the use of the Gram matrix composed of all the inner products (evaluated in the RKHS) of the different input vectors. Accordingly, the size of the Gram matrix linearly increases with time. Without introducing some rules to control the model complexity, this makes these methods intractable for real time applications.

Various algorithms were proposed to overcome this difficulty. The basic idea consists in introducing a new input sample to the model if it contributes in reducing the resulting approximation error and of removing the less relevant element, if necessary. A sparsification rule based on orthogonal projection was proposed in [13–16]. The computational burden of this method is its main drawback. Another method considering the linear dependence criterion is detailed in [17, 18].

The *coherence* criterion, introduced in [19, 20], has been used to control the number of kernel functions introduced in the dictionary[†] so as to limit the model complexity and the size of the Gram matrix. When using the coherence criterion, inclusion of a new input sample into the dictionary is performed if this element cannot be approximated by a linear combination (in the RKHS) of the actual elements of the dictionary. The main result in [20] is that the use of the coherence criterion induces a finite size of the dictionary when time goes to infinity.

However, for all previous approaches, each element introduced into the dictionary remains permanently unchanged, even if the non stationarity makes it later useless for estimating the solution. Therefore, in this paper we also consider the dictionary elements as adjustable model parameters in order to minimize the instantaneous output quadratic error. The main contribution of this paper is to perform dictionary adaptation. We propose an adaptation scheme based on a

*Correspondence to: Institut Charles Delaunay (CNRS), Université de technologie de Troyes, 10010 Troyes, France. E-mail: paul.honeine@utt.fr

[†]The term dictionary stands for a selected set of inputs (or their corresponding kernel functions in the RKHS) used to estimate the nonlinear model.

Prepared using acsauth.cls
DOI: 10.1002/acs

stochastic gradient algorithm. Our algorithm fulfills the constraints resulting from the coherence criterion. We also show that there is no universal dictionary adaptation algorithm, in the sense that each kernel function requires a specific study. We consider here the widely used gaussian, exponential and polynomial kernels and, depending on the used kernel function, we propose specific heuristics for selecting, at each iteration, the dictionary element(s) to be adapted.

In order to demonstrate the effectiveness of our method, we compare the performance obtained with and without dictionary adaptation on the following adaptive filtering algorithms: Kernel Affine Projection Algorithm (KAPA), Kernel Normalized Least Mean Squares (KNLMS), and Kernel Recursive Least Squares (KRLS). We analyze the performance obtained in terms of reduction of the approximation error and/or reduction of the size of the dictionary on several simulated and real data sets.

The paper is organized as follows. In section II, we propose a brief review of nonlinear system modeling methods that make the use of complexity control techniques. Investigations to adapt the elements of a dictionary using polynomial kernel functions and unit-norm kernel functions with radial basis functions are explored in section III. Section IV gives a brief review of the kernel-based adaptive algorithms used in our experiments (i.e., KAPA, KNLMS, and KRLS). The results obtained on synthetic and real data sets are presented in section V. Finally, our conclusion is presented in section VI.

2. A PRIMER ON KERNEL METHODS AND MODEL REDUCTION

Consider a nonlinear online prediction problem and let $\boldsymbol{u}_n \in \mathcal{U} \subset \mathbb{R}^\ell$ be the input data vector and $d_n \in \mathbb{R}$ the corresponding desired output of the model at time step n. Mapping the input data $\boldsymbol{u}_n$ into a high-dimensional space using some nonlinear function $\varphi(\cdot)$ allows the use of linear modeling techniques to the transformed data $\varphi(\boldsymbol{u}_n)$.

2.1. Reproducing kernel Hilbert space and nonlinear regression

Considering a Hilbert space of real-valued functions $\psi(\cdot)$ denoted $\mathcal{H}$, a compact $\mathcal{U} \subset \mathbb{R}^\ell$, and let $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ be the inner product in $\mathcal{H}$. According to Riesz's representation theorem in [11], there exists a unique positive definite kernel denoted by $\kappa(\cdot, \boldsymbol{u}_j)$ called *representer of evaluation* at $\boldsymbol{u}_j$, which satisfies:

$$\psi(\boldsymbol{u}_j) = \langle \psi(\cdot), \kappa(\cdot, \boldsymbol{u}_j) \rangle_{\mathcal{H}}, \quad \forall \psi \in \mathcal{H},$$

Prepared using acsauth.cls
DOI: 10.1002/acs

for every fixed $\boldsymbol{u}_j \in \mathcal{U}$. Replacing $\psi(\cdot)$ by $\kappa(\cdot, \boldsymbol{u}_i)$, yields

$$\kappa(\boldsymbol{u}_i, \boldsymbol{u}_j) = \langle \kappa(\cdot, \boldsymbol{u}_i), \kappa(\cdot, \boldsymbol{u}_j) \rangle_{\mathcal{H}},$$

for all $\boldsymbol{u}_i, \boldsymbol{u}_j \in \mathcal{U}$. If $\kappa(\cdot, \cdot)$ is a Mercer kernel [21], there exists a function $\varphi(\cdot)$ so that $\kappa(\boldsymbol{u}_i, \boldsymbol{u}_j) = \langle \varphi(\boldsymbol{u}_i), \varphi(\boldsymbol{u}_j) \rangle_{\mathcal{H}}$. The kernel then evaluates the inner product of any pair of elements of $\mathcal{U}$ mapped into $\mathcal{H}$ without the explicit knowledge of $\varphi(\cdot)$ or $\mathcal{H}$. This is known as the *kernel trick*.

Let $\kappa(\cdot, \cdot) : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$ be a kernel and $\mathcal{H}$ the RKHS associated with $\kappa(\cdot, \cdot)$. Using the kernel trick, linear algorithms expressed in term of inner products are easily transformed into nonlinear ones. The least-squares algorithm consists of determining a function $\psi^*(\cdot)$ of $\mathcal{H}$ that minimizes a cost function i.e., the sum of the squared error between the actual response of the system (desired output) d_j and the corresponding output of the model $\psi(\boldsymbol{u}_j) = \langle \psi(\cdot), \kappa(\cdot, \boldsymbol{u}_j) \rangle_{\mathcal{H}}$, i.e., $\psi^*(\cdot)$ is the solution of the regularized risk functional

$$\min_{\psi \in \mathcal{H}} \sum_{j=1}^{n} |d_j - \psi(\boldsymbol{u}_j)|^2 + \xi \, g(\|\psi\|^2), \tag{1}$$

where $\| \cdot \|$ is the norm in the RKHS, $\xi \, g(\|\psi\|^2)$ is a regularization term ($\xi \geq 0$) and $g(\cdot)$ is a strictly monotonic function on $\mathbb{R}^+$. According to the representer theorem [22–24], the optimum model $\psi^*(\cdot)$ can be written

$$\psi^*(\cdot) \equiv \psi_n(\cdot) = \sum_{j=1}^{n} \alpha_j \kappa(\cdot, \boldsymbol{u}_j). \tag{2}$$

The minimization of the risk functional (1) leads to an optimization problem with n variables which are the coefficients of the model $\{\alpha_j\}_{j=1 \cdots n}$.

2.2. *A model reduction method using the coherence criterion*

The main problem with online identification methods is the increasing number of observations with time so that adaptation of the model (2) for every new observation becomes more and more complex.

The sparsification is the process of selecting the most relevant kernel functions among the past observations and use them to predict the model. The selected kernel functions form the dictionary and the number of these functions denotes its order m. Many sparsification measures have been used to characterize a dictionary in linear sparse approximation techniques [18, 25–27]. It was also introduced by Mallat and Zhang for matching pursuit [28]. In [20], Richard, Bermudez and Honeine proposed to define the coherence parameter as

$$\mu = \max_{i \neq j} |\langle \kappa(\cdot, \boldsymbol{u}_i), \kappa(\cdot, \boldsymbol{u}_j) \rangle_{\mathcal{H}}| = \max_{i \neq j} |\kappa(\boldsymbol{u}_i, \boldsymbol{u}_j)|,$$

where κ is a unit-norm kernel. If κ is not a unit-norm kernel, $\kappa(\boldsymbol{u}_i, \boldsymbol{u}_j)$ can be replaced by $\kappa(\boldsymbol{u}_i, \boldsymbol{u}_j) / \sqrt{\kappa(\boldsymbol{u}_i, \boldsymbol{u}_i) \kappa(\boldsymbol{u}_j, \boldsymbol{u}_j)}$. The parameter μ is the largest absolute value of the off-diagonal

Prepared using acsauth.cls
DOI: 10.1002/acs

entries in the Gram matrix and it reflects the largest cross-correlations in the dictionary. Therefore, coherence equals zero for every orthogonal basis.

This measure of linear dependence in $\mathcal{H}$ is used to decide if, at each time step n, a candidate kernel function $\kappa(\cdot, \boldsymbol{u}_n)$ must be introduced into the dictionary or not. If this candidate kernel function can be represented by a linear combination of the kernel functions of the dictionary it is dropped, else it is included into the dictionary and the order m is incremented ($m = m + 1$). Introduction of the candidate function $\kappa(\cdot, \boldsymbol{u}_n)$ into the dictionary is decided when coherence remains smaller than a threshold μ_0,

$$\max_{w_j \in \mathcal{J}_{n-1}} |\kappa(\boldsymbol{u}_n, \boldsymbol{u}_{w_j})| \leq \mu_0, \tag{3}$$

where μ_0 is a parameter $\in [0, 1[$ determining the level of sparsity and the coherence of the dictionary, $\mathcal{J}_{n-1}$ denotes a subset of size m of $\{1, \cdots, n-1\}$ and $\mathcal{D}_{n-1} = \{\kappa(\cdot, \boldsymbol{u}_{w_j})\}_{j=1}^{m}$ is the dictionary at time step $n-1$. A dictionary fulfilling (3) is said μ_0-coherent[‡]. At time step n, if $\kappa(\cdot, \boldsymbol{u}_n)$ satisfies the condition (3) it will be introduced into the dictionary which becomes $\mathcal{D}_n = \mathcal{D}_{n-1} \cup \{\kappa(\cdot, \boldsymbol{u}_n)\}$

The key point in the use of the coherence criterion is, as shown in [20], that the size of the dictionary remains finite with time n so the obtained model becomes of fixed-size. It can be written

$$\psi_n(\cdot) = \sum_{j=1}^{m} \alpha_{n,j}\, \kappa(\cdot, \boldsymbol{u}_{w_j}), \tag{4}$$

where $m \equiv m(n)$ converges to some finite value as $n \to \infty$ for any value of $\mu_0 < 1$ and $\boldsymbol{\alpha}_n = [\alpha_{n,1}, \cdots, \alpha_{n,m}]^t$ is the optimal solution vector at time step n.

3. DICTIONARY ADAPTATION

In all previous approaches using sparsification criterion to select dictionary elements, a kernel function $\kappa(\cdot, \boldsymbol{u}_{w_i})$ once introduced into the dictionary, remains unchanged even if it becomes useless in evaluating the model output. This is due to possible non-stationarity. In our method, the dictionary is adapted at each time step in order to minimize the instantaneous quadratic output error. In our approach, the $\{\boldsymbol{u}_{w_j}\}_{j=1}^{m}$ become variables to be optimized, as well as $\{\alpha_{n,j}\}_{j=1}^{m}$. The main constraint is that the dictionary must remain μ_0-coherent after adaptation.

Let $\mathcal{D}_n = \{\kappa(\cdot, \boldsymbol{u}_{w_i})\}_{i=1}^{m}$ be a μ_0-coherent dictionary at time step n and let m denotes its cardinality. Our objective is to adjust the elements of the dictionary $\mathcal{D}_n$ to obtain $\mathcal{D}_n^A$ using an adaptive method. We use an alternate directions optimization scheme thus, in a first step, the optimal value of the solution vector $\boldsymbol{\alpha}_n = [\alpha_{n,1}, \cdots, \alpha_{n,m}]^t$ is found using an online adaptive algorithm.

[‡]A dictionary is μ_0-coherent if all its kernel functions verify the following condition: $\max_{i \neq j} |\kappa(\boldsymbol{u}_{w_i}, \boldsymbol{u}_{w_j})| \leq \mu_0$.

DOI: 10.1002/acs

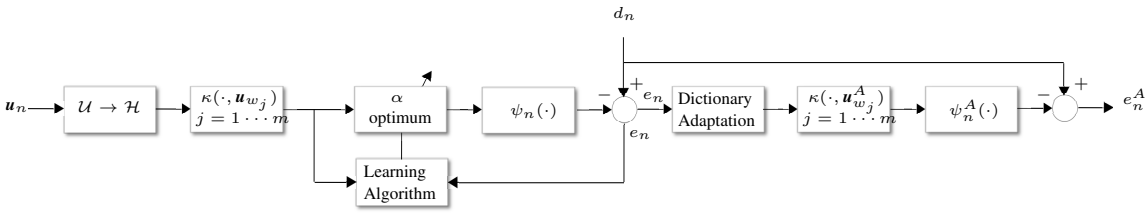


Figure 1. Global adaptation algorithm.

Second, the dictionary is adapted with the objective of reducing the instantaneous quadratic output error e_n^2 and in such a way that the new dictionary still fulfills the coherence criterion, as follows:

$$\min_{\boldsymbol{u}_{w_i} \in \mathcal{J}_n} e_n^2, \tag{5}$$

subject to the constraint

$$\max_{i \neq j, i, j \in \mathcal{J}_n} |\kappa(\boldsymbol{u}_{w_i}, \boldsymbol{u}_{w_j})| \leq \mu_0,$$

where $e_n = d_n - \psi_n(\boldsymbol{u}_n) = d_n - \sum_{i=1}^{m} \alpha_{n,i}\, \kappa(\boldsymbol{u}_n, \boldsymbol{u}_{w_i})$. Here, d_n represents the desired output of the model.

The global adaptation algorithm is represented in Figure 1. The i-th element of the dictionary $\boldsymbol{u}_{w_i}$ is updated according to:

$$\boldsymbol{u}^A_{w_i} = \boldsymbol{u}_{w_i} - \nu_n \mathbf{g}_{w_i} \quad \forall i = 1 \cdots m, \tag{6}$$

where $\boldsymbol{u}^A_{w_i}$ is the i-th dictionary element after adaptation, $\mathbf{g}_{w_i}$ is the gradient of the instantaneous quadratic error with respect to $\boldsymbol{u}_{w_i}$, and ν_n represents the gradient step size used to adjust the elements of the dictionary. Obviously, the gradient step size must be selected so as the new dictionary remains in the feasible domain.

- By construction of our algorithm, the coherence criterion is satisfied at each time step n before adaptation

$$\max_{i \neq j} |\kappa(\boldsymbol{u}_{w_i}, \boldsymbol{u}_{w_j})| \leq \mu_0, \quad \forall i, j = 1 \cdots m, \tag{7}$$

  and this must remain true after adaptation, so we must have

$$\max_{i \neq j} |\kappa(\boldsymbol{u}^A_{w_i}, \boldsymbol{u}^A_{w_j})| \leq \mu_0. \quad \forall i, j = 1 \cdots m. \tag{8}$$

- Using (5), the gradient with respect to $\boldsymbol{u}_{w_i}$ is:

$$\mathbf{g}_{w_i} = \nabla_{\boldsymbol{u}_{w_i}}(e_n^2) = 2\, e_n\, \nabla_{\boldsymbol{u}_{w_i}} \left(d_n - \sum_{i=1}^{m} \alpha_{n,i}\, \kappa(\boldsymbol{u}_n, \boldsymbol{u}_{w_i}) \right) = -2\, e_n\, \alpha_{n,i}\, \nabla_{\boldsymbol{u}_{w_i}} \Big(\kappa(\boldsymbol{u}_n, \boldsymbol{u}_{w_i}) \Big). \tag{9}$$

  From the expression of the gradient $\mathbf{g}_{w_i}$ (9), it is obvious that the choice of the kernel function to be used has a major influence in the adaptation process. In this paper, we explore the radial basis kernel, which belongs to the class of translation-invariant kernels, and the polynomial kernel.

Prepared using acsauth.cls
DOI: 10.1002/acs

3.1. Case of the radial basis kernel

Most of translation-invariant kernels, mainly the gaussian and the exponential kernel functions, can be expressed in the form:

$$\kappa(\boldsymbol{u}_i, \boldsymbol{u}_j) = f(\|\boldsymbol{u}_i - \boldsymbol{u}_j\|^2), \tag{10}$$

where $f \in \mathcal{C}^\infty$. A sufficient condition for this function to be a valid positive-definite kernel is the complete monotonicity of the function f, i.e.,

$$(-1)^k f^{(k)}(r) \geq 0, \forall r \geq 0, \tag{11}$$

where $f^{(k)}(\cdot)$ denotes the k-th derivative of $f(\cdot)$ [29]. From (10), we get

$$\nabla_{\boldsymbol{u}_j} \kappa(\boldsymbol{u}_i, \boldsymbol{u}_j) = -2(\boldsymbol{u}_i - \boldsymbol{u}_j)\, f^{(1)}(\|\boldsymbol{u}_i - \boldsymbol{u}_j\|^2). \tag{12}$$

Note that, in the case of a radial basis kernel, the condition in (7) for any two elements of a μ_0-coherent dictionary is expressed as follows:

$$f(\|\boldsymbol{u}_{w_i} - \boldsymbol{u}_{w_j}\|^2) \leq \mu_0. \tag{13}$$

3.1.1. Gradient and coherence condition for the gaussian kernel

The kernel function for the i-th element of the dictionary is:

$$\kappa(\cdot, \boldsymbol{u}_{w_i}) = \exp\left(-\frac{\|\cdot - \boldsymbol{u}_{w_i}\|^2}{2\sigma^2}\right),$$

where σ is the bandwidth of the kernel. Taking the derivative of this kernel w.r.t. $\boldsymbol{u}_{w_i}$, the gradient of the instantaneous quadratic error defined in (9) and (12) yields

$$\mathbf{g}_{w_i} = -2\,\frac{e_n\,\alpha_{n,i}}{\sigma^2}\,\kappa(\boldsymbol{u}_n, \boldsymbol{u}_{w_i})\,(\boldsymbol{u}_n - \boldsymbol{u}_{w_i}).$$

the coherence condition for the gaussian kernel, according to expressions (7) and (8) is:

$$\|\boldsymbol{u}^A_{w_i} - \boldsymbol{u}^A_{w_j}\|^2 \geq -2\,\sigma^2 \ln(\mu_0),$$

with $-2\,\sigma^2 \ln(\mu_0) \geq 0$ because $\mu_0 \in [0, 1[$.

3.1.2. Gradient and coherence condition for an exponential kernel

The kernel function for the i-th element of the dictionary is:

$$\kappa(\cdot, \boldsymbol{u}_{w_i}) = \exp\left(-\frac{\|\cdot - \boldsymbol{u}_{w_i}\|}{2\sigma^2}\right),$$

where σ is the bandwidth of the kernel function. The gradient of the instantaneous quadratic error with respect to $\boldsymbol{u}_{w_i}$ at time step n is:

$$\mathbf{g}_{w_i} = -\frac{e_n\,\alpha_{n,i}}{\sigma^2\,\|\boldsymbol{u}_n - \boldsymbol{u}_{w_i}\|}\,\kappa(\boldsymbol{u}_n, \boldsymbol{u}_{w_i})\,(\boldsymbol{u}_n - \boldsymbol{u}_{w_i}).$$

Prepared using acsauth.cls
DOI: 10.1002/acs

Obviously, from the above expression, $\|\boldsymbol{u}_n - \boldsymbol{u}_{w_i}\|$ must be non zero, meaning that when introducing $\boldsymbol{u}_n$ in the dictionary, the newly introduced element cannot be adjusted.

The coherence condition for an exponential kernel, according to equations (7) and (8) is:

$$\|\boldsymbol{u}^A_{w_i} - \boldsymbol{u}^A_{w_j}\|^2 \geq \left(-2\sigma^2 \ln(\mu_0)\right)^2.$$

3.1.3. Finding a feasible gradient step size in the case of a radial basis kernel

The main issue is clearly to find an adequate gradient step size to simultaneously adjust all the dictionary elements without violating the coherence criterion. For any pair of dictionary elements, we obtain

$$\boldsymbol{u}^A_{w_i} - \boldsymbol{u}^A_{w_j} = \delta\boldsymbol{u} - \nu_n \delta\mathbf{g} \;\; \forall i, j = 1 \cdots m,$$

where $\delta\boldsymbol{u} = \boldsymbol{u}_{w_i} - \boldsymbol{u}_{w_j}$ and $\delta\mathbf{g} = \mathbf{g}_{w_i} - \mathbf{g}_{w_j}$. Because of the constraint induced by the coherence criterion, ν_n cannot be chosen arbitrarily. The problem is now to determine an appropriate ν_n at time step n in order to adapt the dictionary. The coherence condition (8) leads to

$$f(\|\delta\boldsymbol{u} - \nu_n\, \delta\mathbf{g}\|^2) \leq \mu_0. \tag{14}$$

It is possible to construct a local model of the kernel function, by approximating it with a Taylor series around $\nu_n \sim 0$:

$$f(\|\delta\boldsymbol{u} - \nu_n\delta\mathbf{g}\|^2) = f(\|\delta\boldsymbol{u}\|^2) - 2\nu_n(\delta\boldsymbol{u}^t\delta\mathbf{g} - \nu_n\|\delta\mathbf{g}\|^2)f^{(1)}(\|\delta\boldsymbol{u}\|^2) + \mathcal{O}(\nu_n).$$

Using this approximation, condition (14) becomes

$$\begin{aligned} -\Big(2\|\delta\mathbf{g}\|^2\nu_n^2 - 2\nu_n\delta\boldsymbol{u}^t\delta\mathbf{g})f^{(1)}(\|\delta\boldsymbol{u}\|^2\Big) + \mu_0 - f(\|\delta\boldsymbol{u}\|^2) \geq 0 \\ -2\|\delta\mathbf{g}\|^2 f^{(1)}\Big(\|\delta\boldsymbol{u}\|^2\Big)\nu_n^2 + 2\delta\boldsymbol{u}^t\delta\mathbf{g}f^{(1)}(\|\delta\boldsymbol{u}\|^2\Big)\nu_n + \mu_0 - f(\|\delta\boldsymbol{u}\|^2) \geq 0 \end{aligned} \tag{15}$$

It is very important to note that the first derivative of a radial basis function is always negative on $\mathbb{R}^+$, which means that the coefficient of the second degree term in ν_n is always positive, namely $-2\|\delta\mathbf{g}\|^2 f^{(1)}(\|\delta\boldsymbol{u}\|^2) > 0$. Second, from (13) we have $\mu_0 - f(\|\delta\boldsymbol{u}\|^2) \geq 0$ as the coherence criterion is fulfilled at any time step.

The reduced discriminant of this quadratic expression in ν_n is:

$$\Delta = \Big(\delta\boldsymbol{u}^t\delta\mathbf{g}\, f^{(1)}(\|\delta\boldsymbol{u}\|^2)\Big)^2 + 2\|\delta\mathbf{g}\|^2 f^{(1)}(\|\delta\boldsymbol{u}\|^2)(\mu_0 - f(\|\delta\boldsymbol{u}\|^2).$$

Let ν_{ij_n} be a valid gradient step size for adapting the two dictionary elements $\boldsymbol{u}_{w_i}$ and $\boldsymbol{u}_{w_j}$ (at time step n). If $\Delta < 0$, there is no constraint on the value of the step size ν_{ij_n}. Otherwise, if $\Delta \geq 0$, the boundary points of (15) will be $\nu_{i,j-}$ and $\nu_{i,j+}$ as follows:

$$\nu_{i,j\pm} = \frac{-\delta\boldsymbol{u}^t\delta\mathbf{g}\, f^{(1)}(\|\delta\boldsymbol{u}\|^2) \pm \sqrt{\Delta}}{-\|\delta\mathbf{g}\|^2 f^{(1)}(\|\delta\boldsymbol{u}\|^2)},$$

Prepared using acsauth.cls
DOI: 10.1002/acs

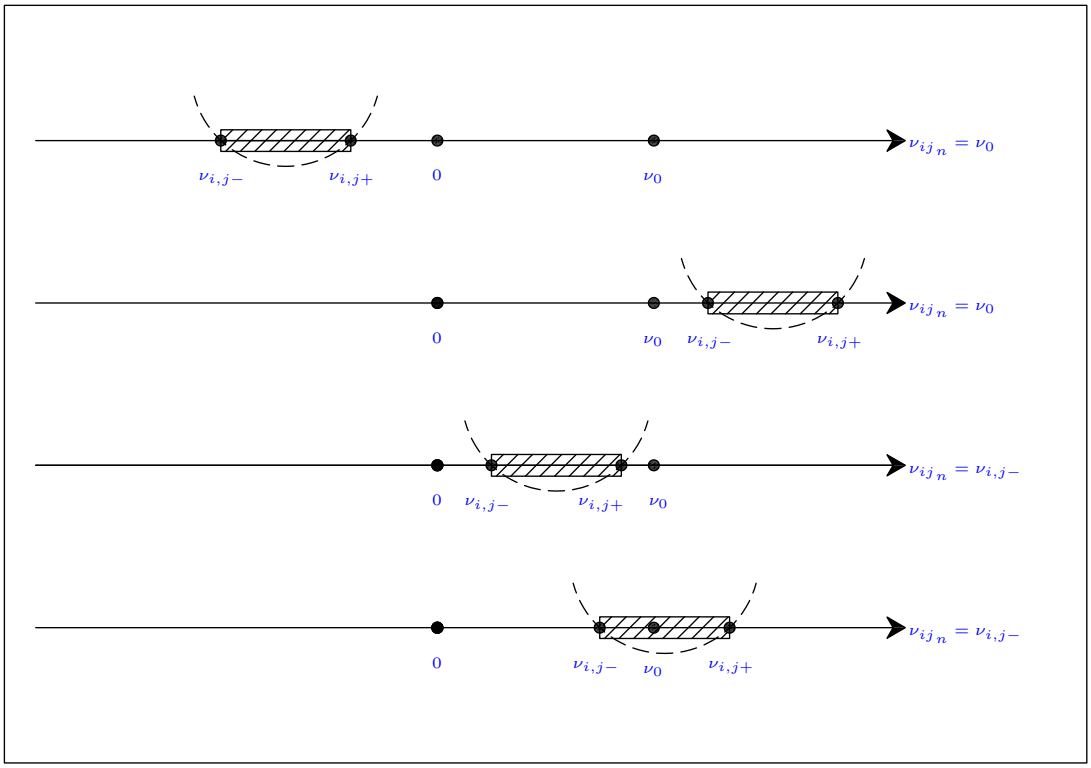


Figure 2. An illustration showing all the possibilities to select the gradient step size between any two dictionary elements $\boldsymbol{u}_{w_i}$ and $\boldsymbol{u}_{w_j}$ in the case of a radial basis kernel. $\nu_0 > 0$ is a reference step size.

and the interval of possible values for ν_{ij_n} will be $]-\infty, \nu_{i,j-}] \cup [\nu_{i,j+}, +\infty[$, since the quadratic expression in (15) must be positive:

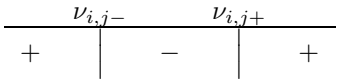


Figure 2 shows all the possibilities to select the gradient step size when adapting any two dictionary elements $\boldsymbol{u}_{w_i}$ and $\boldsymbol{u}_{w_j}$ in the case of a radial basis kernel. Obviously, for each pair of dictionary elements $(\boldsymbol{u}_{w_i}, \boldsymbol{u}_{w_j})$, $\nu_{ij_n} = 0$ always belong to the feasible domain, since there is no adaptation and the initial dictionary was μ_0-coherent. This is also the reason for which we can always use Taylor series around $\nu_n = 0$. Finally because the coefficient of the second degree term in ν_n and the constant term in (15) are positive, $\nu_{i,j-}$ and $\nu_{i,j+}$ have the same sign.

Interpretation of the two bounds for ν_n is straightforward. When adjusting any two dictionary elements, each one according to its own gradient of the quadratic error, the interval $]\nu_{i,j-}, \nu_{i,j+}[$ must be eliminated to avoid any overlap of the influence regions of the two considered elements, and thus violation of the coherence constraint (8); see Figure 3.

Prepared using acsauth.cls
DOI: 10.1002/acs

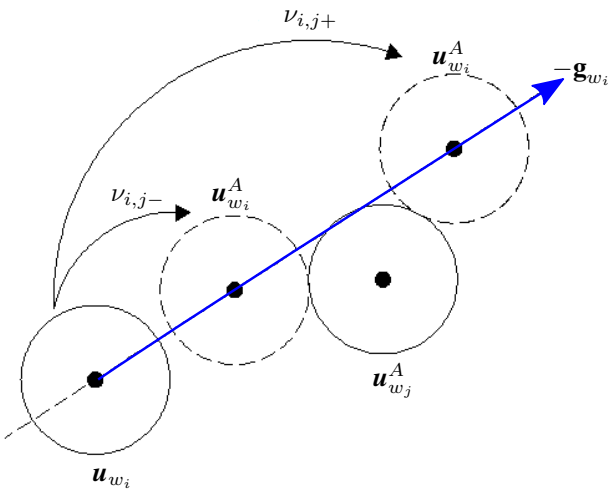


Figure 3. A 2D illustration showing the constraint of choosing $\nu_{ij\,n} \leq \nu_{i,j-}$ or $\nu_{ij\,n} \geq \nu_{i,j+}$ to avoid the overlap of the influence regions of $\boldsymbol{u}^A_{w_i}$ and $\boldsymbol{u}^A_{w_j}$.

We initially select a reference step size $\nu_0 > 0$, similarly as for all adaptive algorithms with a fixed step size. After calculating the $m(m-1)/2$ intervals $[\nu_{i,j-}, \nu_{i,j+}]$, the single adaptation step ν_n used for all dictionary elements is selected as illustrated in Figure 2:

- if $\max\limits_{i,j} \nu_{i,j+} \leq 0 \;\Rightarrow \nu_n = \nu_0$;
- if $0 \leq \min\limits_{i,j} \nu_{i,j-} \leq \nu_0 \;\Rightarrow \nu_n = \min\limits_{i,j} \nu_{i,j-}$;
- if $0 \leq \nu_0 \leq \min\limits_{i,j} \nu_{i,j-} \;\Rightarrow \nu_n = \nu_0$;
- if $0 \leq \min\limits_{i,j} (\nu_{i,j-})^+ \leq \nu_0 \;\Rightarrow \nu_n = \min\limits_{i,j} (\nu_{i,j-})^+$;
- if $0 \leq \nu_0 \leq \min\limits_{i,j} (\nu_{i,j-})^+ \;\Rightarrow \nu_n = \nu_0$.

In these expressions, $(\nu_{i,j-})^+$ indicates the positive part of $\nu_{i,j-}$. Note that ν_0 must be selected relatively small. If ν_0 is too large, the elements of the dictionary can be spread over a non useful region of the input space, inducing an increase of the size of the dictionary without reducing the approximation error. Other heuristics for dictionary adaptation could be considered, but our main objective is only to illustrate the potential efficiency of dictionary adaptation.

3.2. Case of a Polynomial Kernel

3.2.1. Gradient and coherence criterion

The polynomial kernel function is of the form:

$$k(\boldsymbol{u}_i, \boldsymbol{u}_j) = f(\boldsymbol{u}_i^t \boldsymbol{u}_j) = (1 + \boldsymbol{u}_i^t \boldsymbol{u}_j)^\beta, \tag{16}$$

Prepared using acsauth.cls
DOI: 10.1002/acs

with $\beta > 0$. This kernel function is not unit-norm, hence, because of the use of the coherence criterion, we need to normalize it to become a unit-norm kernel [20]. This is achieved by substituting $k(\boldsymbol{u}_i, \boldsymbol{u}_j)$ with:

$$\frac{k(\boldsymbol{u}_i, \boldsymbol{u}_j)}{\sqrt{k(\boldsymbol{u}_i, \boldsymbol{u}_i)}\sqrt{k(\boldsymbol{u}_j, \boldsymbol{u}_j)}}.$$

Therefore, equation (5) becomes:

$$e_n = d_n - \sum_{i=1}^{m} \alpha_{n,i} \frac{k(\boldsymbol{u}_n, \boldsymbol{u}_{w_i})}{\sqrt{k(\boldsymbol{u}_n, \boldsymbol{u}_n)}\sqrt{k(\boldsymbol{u}_{w_i}, \boldsymbol{u}_{w_i})}}. \tag{17}$$

The gradient $\mathbf{g}_{w_i}$ of the instantaneous quadratic error with respect to $\boldsymbol{u}_{w_i}$ is:

$$\mathbf{g}_{w_i} = -2\,\beta\, e_n\, \alpha_{n,i} \frac{k(\boldsymbol{u}_n, \boldsymbol{u}_{w_i})}{\sqrt{k(\boldsymbol{u}_n, \boldsymbol{u}_n)}\sqrt{k(\boldsymbol{u}_{w_i}, \boldsymbol{u}_{w_i})}} \left(\frac{\boldsymbol{u}_n}{k^*(\boldsymbol{u}_n, \boldsymbol{u}_{w_i})} - \frac{1}{2} \frac{\boldsymbol{u}_{w_i}}{k^*(\boldsymbol{u}_{w_i}, \boldsymbol{u}_{w_i})} \right), \tag{18}$$

where $k^*(\boldsymbol{u}_i, \boldsymbol{u}_j) = (1 + \boldsymbol{u}_i^t \boldsymbol{u}_j)$.

The coherence criterion (7), when applied to the polynomial kernel, yields:

$$\frac{|(1 + \boldsymbol{u}_{w_i}^t \boldsymbol{u}_{w_j})^\beta|}{\sqrt{(1 + \boldsymbol{u}_{w_i}^t \boldsymbol{u}_{w_i})^\beta}\sqrt{(1 + \boldsymbol{u}_{w_j}^t \boldsymbol{u}_{w_j})^\beta}} \leq \mu_0, \tag{19}$$

which can be written:

$$\frac{(1 + \boldsymbol{u}_{w_i}^t \boldsymbol{u}_{w_j})^2}{(1 + \|\boldsymbol{u}_{w_i}\|^2)\,(1 + \|\boldsymbol{u}_{w_j}\|^2)} \leq (\mu_0)^{2/\beta}. \tag{20}$$

Considering (20), it seems to be challenging to adjust simultaneously all the dictionary elements while fulfilling the coherence criterion (3). This explains our choice of selecting, for the polynomial kernel, only one element of the dictionary to be adapted at each time step. Heuristics for selecting this element are presented in section 3.2.4.

3.2.2. Search for possible values of the gradient step size

We must now be sure that, after adaptation of the selected element, the new dictionary still fulfills the coherence criterion. Let $\boldsymbol{u}_{w_i}$ denote the dictionary element to be adapted. Introducing (6) in (20) gives:

$$\frac{\left(1 + (\boldsymbol{u}_{w_i}^A)^t\, \boldsymbol{u}_{w_j}\right)^2}{(1 + \|\boldsymbol{u}_{w_i}^A\|^2)\,(1 + \|\boldsymbol{u}_{w_j}\|^2)} \leq (\mu_0)^{2/\beta} \quad \forall j \neq i,\ j = 1 \cdots m. \tag{21}$$

Since

$$(\boldsymbol{u}_{w_i}^A)^t\, \boldsymbol{u}_{w_j} = (\boldsymbol{u}_{w_i} - \nu_n \boldsymbol{g}_{w_i})^t\, \boldsymbol{u}_{w_j} = \boldsymbol{u}_{w_i}^t \boldsymbol{u}_{w_j} - \nu_n\, \boldsymbol{u}_{w_j}^t \boldsymbol{g}_{w_i},$$

and

$$\|\boldsymbol{u}_{w_i}^A\|^2 = \|\boldsymbol{u}_{w_i}\|^2 - 2\nu_n \boldsymbol{u}_{w_i}^t \boldsymbol{g}_{w_i} + \nu_n^2 \|\boldsymbol{g}_{w_i}\|^2,$$

inequality (21) becomes:

$$\frac{k_{ij}^{*2} - 2\,\nu_n\, \boldsymbol{u}_{w_j}^t \boldsymbol{g}_{w_i} k_{ij}^* + \nu_n^2 \left(\boldsymbol{u}_{w_j}^t \boldsymbol{g}_{w_i}\right)^2}{k_{jj}^* \left(k_{ii}^* - 2\nu_n \boldsymbol{u}_{w_i}^t \boldsymbol{g}_{w_i} + \nu_n^2 \|\boldsymbol{g}_{w_i}\|^2\right)} \leq (\mu_0)^{2/\beta}, \tag{22}$$

Prepared using acsauth.cls
DOI: 10.1002/acs

where $k^*_{ij} = k^*(\boldsymbol{u}_{w_i}, \boldsymbol{u}_{w_j}) = (1 + \boldsymbol{u}^t_{w_i}\boldsymbol{u}_{w_j})$. Setting $D = k^*_{jj}\,(\mu_0)^{2/\beta}$, (22) becomes:

$$\left(\left(\boldsymbol{u}^t_{w_j}\boldsymbol{g}_{w_i}\right)^2 - D\|\boldsymbol{g}_{w_i}\|^2\right)\nu_n^2 - 2\left(k^*_{ij}\boldsymbol{u}^t_{w_j}\boldsymbol{g}_{w_i} - D\boldsymbol{u}^t_{w_i}\boldsymbol{g}_{w_i}\right)\nu_n + \left(k^{*^2}_{ij} - Dk^*_{ii}\right) \leq 0, \tag{23}$$

which is of the form:

$$A\,\nu_n^2 + 2B\,\nu_n + C \geq 0, \tag{24}$$

with

$$\begin{aligned} A &= D\|\boldsymbol{g}_{w_i}\|^2 - \left(\boldsymbol{u}^t_{w_j}\boldsymbol{g}_{w_i}\right)^2, \\ B &= k^*_{ij}\boldsymbol{u}^t_{w_j}\boldsymbol{g}_{w_i} - D\boldsymbol{u}^t_{w_i}\boldsymbol{g}_{w_i}, \\ C &= Dk^*_{ii} - k^{*^2}_{ij}. \end{aligned}$$

Here, C is always positive for a coherent dictionary, this results from the fulfillment of the coherence criterion:

$$\frac{k_{ij}}{\sqrt{k_{ii}\,k_{jj}}} \leq \mu_0 \;\Rightarrow\; \frac{k^{*^2}_{ij}}{k^*_{ii}k^*_{jj}} \leq (\mu_0)^{2/\beta}.$$

The reduced discriminant is:

$$\Delta = B^2 - AC = \left(k^*_{ij}\boldsymbol{u}^t_{w_j}\boldsymbol{g}_{w_i} - D\boldsymbol{u}^t_{w_i}\boldsymbol{g}_{w_i}\right)^2 - \left(D\|\boldsymbol{g}_{w_i}\|^2 - (\boldsymbol{u}^t_{w_j}\boldsymbol{g}_{w_i})^2\right)\left(Dk^*_{ii} - k^{*^2}_{ij}\right).$$

As can be seen from (24), selecting a valid value of ν_n depends on the sign of A. The following analysis, based on the study of the positiveness of a second degree polynomial, considers all possible cases.

- Case of $A < 0$. In this case $\Delta \geq 0$
  - if $\Delta > 0$ then the polynomial defined in (23) has two roots with opposite signs:

$$\nu_{i,j}\pm = \frac{-\left(k^*_{ij}\boldsymbol{u}^t_{w_j}\boldsymbol{g}_{w_i} - D\boldsymbol{u}^t_{w_i}\boldsymbol{g}_{w_i}\right) \pm \sqrt{\Delta}}{D\|\boldsymbol{g}_{w_i}\|^2 - \left(\boldsymbol{u}^t_{w_j}\boldsymbol{g}_{w_i}\right)^2},$$

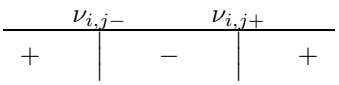
    with:

	$\nu_{i,j-}$		$\nu_{i,j+}$	
$-$	|	$+$	|	$-$

    $\nu_n \in [\nu_{i,j-}, \nu_{i,j+}]$ with $\nu_{i,j-} \leq 0 \leq \nu_{i,j+}$, because $\nu_n = 0$ is always possible since the dictionary is not adapted in this case.
  - if $\Delta = 0$, then $B^2 = AC$ and since $A < 0$ and $C \geq 0$, this implies that $B = 0$ and $\nu_{i,j-} = \nu_{i,j+} = \frac{-B}{A} = 0$.
- Case of $A > 0$. In this case, three possibilities must be considered:

Prepared using acsauth.cls
DOI: 10.1002/acs

– if $\Delta > 0$, the two roots have the same sign since $\frac{C}{A} \geq 0$, with:



and $\nu_n \in \ ]-\infty, \nu_{i,j-}] \cup [\nu_{i,j+}, +\infty[$. Of course $\nu_n = 0$ lies in the interval of possible values.

– if $\Delta < 0$ the polynomial does not have roots but is always positive, meaning that there is no constraint for selecting $\nu_n \in \ ]-\infty, +\infty[$.

– if $\Delta = 0$, the polynomial is positive or null and there is no constraint for selecting ν_n.

- Case of $A = 0$. The unique solution is $\nu_{i,j} = \frac{-C}{2B}$
  - if $B > 0$, this means that $\nu_n \in \ ]\nu_{i,j}, +\infty\,[$.
  - if $B \leq 0$, then $\frac{-C}{2B} \geq 0$ and $\nu_n \in \ ]-\infty, \nu_{i,j}[$.

3.2.3. Selection of the gradient step size

After selecting the element $\boldsymbol{u}_{w_i}$ to adapt, the calculation is made to satisfy the $(m-1)$ constraints with the other dictionary elements. We first select, as usual, a reference step size $\nu_0 > 0$. Second, for $j \neq i$ and $j = 1 \cdots m$, the value of ν_{ij_n} corresponding to the couple $(\boldsymbol{u}_{w_i}, \boldsymbol{u}_{w_j})$ is selected as follows:

1. Case of $A > 0$ and $\Delta > 0$
   - if $\nu_{i,j-} \leq \nu_{i,j+} \leq 0 \Rightarrow \nu_{ij_n} = \nu_0$
   - if $0 \leq \nu_{i,j-} \leq \nu_0 \leq \nu_{i,j+} \Rightarrow \nu_{ij_n} = \nu_{i,j-}$
   - if $0 \leq \nu_{i,j-} \leq \nu_{i,j+} \leq \nu_0 \Rightarrow \nu_{ij_n} = \nu_0$ or $\nu_{ij_n} = \nu_{i,j-}$
   - if $0 \leq \nu_0 \leq \nu_{i,j-} \leq \nu_{i,j+} \Rightarrow \nu_{ij_n} = \nu_0$ or $\nu_{ij_n} = \nu_{i,j-}$
2. Case of $A > 0$ and $\Delta \leq 0$
   - if such is the case, we select $\nu_{ij_n} = \nu_0$
3. Case of $A < 0$ and $\Delta > 0$

   in this case $\nu_{i,j-} \leq 0 \leq \nu_{i,j+}$
   - if $\nu_{i,j+} \leq \nu_0 \Rightarrow \nu_{ij_n} = \nu_{i,j+}$
   - if $\nu_0 \leq \nu_{i,j+} \Rightarrow \nu_{ij_n} = \nu_0$ or $\nu_{ij_n} = \nu_{i,j+}$
4. Case of $A < 0$ and $\Delta = 0$
   - we have $\nu_{ij_n} = 0$

DOI: 10.1002/acs

5. Case of $A = 0$

- if $\frac{-C}{2B} < 0 \Rightarrow \nu_{ij_n} = \nu_0$.
- if $0 \leq \frac{-C}{2B} \leq \nu_0 \Rightarrow \nu_{ij_n} = \frac{-C}{2B}$.
- if $\nu_0 \leq \frac{-C}{2B} \Rightarrow \nu_{ij_n} = \nu_0$.

When the $(m-1)$ values ν_{ij_n} are obtained, we select ν_n as $\nu_n = \min_{j \neq i,\, j=1 \cdots m} \nu_{ij_n}$.

3.2.4. Selection of the element to be adapted

Several approaches can be adopted to select the element that will be adapted. We only investigated two methods. The first one chooses the element that corresponds to the smallest absolute value of the components of the solution vector $\boldsymbol{\alpha}_n = [\alpha_{n,1}, \cdots, \alpha_{n,m}]^t$. The second consists of calculating the partial gradient of the quadratic error with respect to each element of the dictionary. Then we select the one with the highest norm of the gradient. The advantage of the first heuristic is its simplicity, while the second gives better results but requires more computations. Both methods were implemented and are compared in the experimentation section.

4. A BRIEF REVIEW OF SOME ONLINE LEARNING ALGORITHMS

In this paper, and for reasons only related to the possibility to compare the performance obtained with previously published results, we use the Kernel Recursive Least Squares algorithm (KRLS), the Kernel Affine Projection Algorithm (KAPA), and the Kernel Normalized Least Squares algorithm (KNLMS). These algorithms are used with and without dictionary adaptation. The output of the model is:

$$\psi_n(\cdot) = \sum_{j=1}^{m} \alpha_{n,j}\, \kappa(\cdot, \boldsymbol{u}_{w_j}). \tag{25}$$

The optimal solution vector $\boldsymbol{\alpha}_n$ is found using one of the above mentioned online adaptive algorithm.

4.1. Kernel Recursive Least Squares algorithm (KRLS)

In the Kernel Recursive Least Squares algorithm, the optimization problem (1) can be written as follows [19]:

$$\min_{\boldsymbol{\alpha}} \|\boldsymbol{d}_n - \boldsymbol{H}_n \boldsymbol{\alpha}\|^2 + \xi \boldsymbol{\alpha}^t \boldsymbol{K}_n \boldsymbol{\alpha}, \tag{26}$$

where $\boldsymbol{K}_n$ is the $m \times m$ Gram matrix of the dictionary at time n, ξ is a regularization coefficient, and $\boldsymbol{H}_n$ is an $(n \times m)$ matrix with (i,j)-th element is $\kappa(\boldsymbol{u}_i, \boldsymbol{u}_{w_j})$. At time step n the solution of the

Prepared using acsauth.cls
DOI: 10.1002/acs

problem (26) will be [19]:

$$\boldsymbol{\alpha}_n = \boldsymbol{P}_n \boldsymbol{H}_n^t \boldsymbol{d}_n, \tag{27}$$

provided that $\boldsymbol{P}_n = (\boldsymbol{H}_n \boldsymbol{H}_n^t + \xi \boldsymbol{K}_n)^{-1}$ exists. At time step n+1, there is a new input $\boldsymbol{u}_{n+1}$ to the model and one of the following cases can occur:

1. *First case*: $\max_{j=1\cdots m} |\kappa(\boldsymbol{u}_{n+1}, \boldsymbol{u}_{w_j})| > \mu_0$

   The new input can be represented by the actual elements of the dictionary. It is not introduced in the dictionary. The matrix $\boldsymbol{H}_n$ is updated by adding a $(n+1)-th$ row $\boldsymbol{h}_{n+1} = [\kappa(\boldsymbol{u}_{n+1}, \boldsymbol{u}_{w_1}), \cdots, \kappa(\boldsymbol{u}_{n+1}, \boldsymbol{u}_{w_m})]^t$ and the new desired output d_{n+1} is added to the output vector $\boldsymbol{d}_n$. Update of the model is performed as follows:

   $$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}_n + \frac{\boldsymbol{P}_n \boldsymbol{h}_{n+1}}{1 + \boldsymbol{h}_{n+1}^t \boldsymbol{P}_n \boldsymbol{h}_{n+1}} \Big(d_{n+1} - \boldsymbol{h}_{n+1}^t \boldsymbol{\alpha}_n \Big), \tag{28}$$

   and

   $$\boldsymbol{P}_{n+1} = \boldsymbol{P}_n - \frac{\boldsymbol{P}_n \boldsymbol{h}_{n+1} \boldsymbol{h}_{n+1}^t \boldsymbol{P}_n}{1 + \boldsymbol{h}_{n+1}^t \boldsymbol{P}_n \boldsymbol{h}_{n+1}}. \tag{29}$$

2. *Second case*: $\max_{j=1\cdots m} |\kappa(\boldsymbol{u}_{n+1}, \boldsymbol{u}_{w_j})| \leq \mu_0$

   The new input cannot be represented by any actual element of the dictionary. Thus, it must be introduced in the dictionary. The order of the dictionary becomes $(m+1)$. First, the vector $\boldsymbol{\alpha}_n$ and $\boldsymbol{P}_n$ are updated according to (28) and (29), leading to $\tilde{\boldsymbol{\alpha}}_{n+1}$ and $\tilde{\boldsymbol{P}}_{n+1}$ respectively. Second, in order to introduce a new coefficient in the model, the new $\boldsymbol{\alpha}_{n+1}$ and $\boldsymbol{P}_{n+1}$ are given by:

   $$\boldsymbol{\alpha}_{n+1} = \begin{bmatrix} \tilde{\boldsymbol{\alpha}}_{n+1} \\ 0 \end{bmatrix} + \frac{d_{n+1} - \boldsymbol{h}_{n+1}^t \boldsymbol{\alpha}_n}{1 - \boldsymbol{h}_{n+1}^t \tilde{\boldsymbol{P}}_{n+1} \boldsymbol{h}_{n+1}} \begin{bmatrix} \tilde{\boldsymbol{P}}_{n+1} \boldsymbol{h}_{n+1} \\ 1/h_0 \end{bmatrix}, \tag{30}$$

   $$\boldsymbol{P}_{n+1} = \begin{bmatrix} \tilde{\boldsymbol{P}}_{n+1} & \boldsymbol{0}_n \\ \boldsymbol{0}_n^t & 0 \end{bmatrix} + \frac{1}{1 - \boldsymbol{h}_{n+1}^t \tilde{\boldsymbol{P}}_{n+1} \boldsymbol{h}_{n+1}} \times \begin{bmatrix} -\tilde{\boldsymbol{P}}_{n+1} \boldsymbol{h}_{n+1} \\ 1/h_0 \end{bmatrix} \begin{bmatrix} -(\tilde{\boldsymbol{P}}_{n+1} \boldsymbol{h}_{n+1})^t & 1/h_0 \end{bmatrix}, \tag{31}$$

   where $h_0 = \kappa(\boldsymbol{u}_{n+1}, \boldsymbol{u}_{n+1})$.

4.2. *Kernel Affine Projection Algorithm*

The Kernel Affine Projection Algorithm, studied in the linear case in [30–32] and in the Kernelized APA in [33, 34], results from the following optimization problem (while neglecting the regularized term in (1)):

$$\min_{\boldsymbol{\alpha}} \|\boldsymbol{d}_n - \boldsymbol{H}_n \boldsymbol{\alpha}\|^2. \tag{32}$$

The optimal solution $\boldsymbol{\alpha}_n$ of the problem (32) is:

$$\boldsymbol{\alpha}_n = (\boldsymbol{H}_n^t \boldsymbol{H}_n)^{-1} \boldsymbol{H}_n^t \boldsymbol{d}_n, \tag{33}$$

Prepared using acsauth.cls
DOI: 10.1002/acs

provided that $(\boldsymbol{H}_n^t \boldsymbol{H}_n)^{-1}$ exists and $\boldsymbol{H}_n$ is a n-by-m matrix whose (i,j)-th entry is $\kappa(\boldsymbol{u}_{n-i+1}, \boldsymbol{u}_{w_j})$, and the i-th element of the column vector $\boldsymbol{d}_n$ is d_{n-i+1}. At each time step n, the affine problem will be:

$$\min_{\boldsymbol{\alpha}} \|\boldsymbol{\alpha} - \boldsymbol{\alpha}_{n-1}\|^2 \text{ subject to } \boldsymbol{d}_{\mathrm{n}} = \boldsymbol{H}_{\mathrm{n}}\boldsymbol{\alpha}, \tag{34}$$

where, at each time step n, only the p most recent inputs $\{\boldsymbol{u}_n, \cdots, \boldsymbol{u}_{n-p+1}\}$ and observations $\{d_n, \cdots, d_{n-p+1}\}$ are used, so the i-th element of the column vector $\boldsymbol{d}_n$ is d_{n-i+1}, $i = 1 \cdots, p$, and $\boldsymbol{H}_n$ is the $p \times m$ matrix whose (i,j)-th entry is $\kappa(\boldsymbol{u}_{n-i+1}, \boldsymbol{u}_{w_j})$. This means that $\boldsymbol{\alpha}_n$ is obtained by projecting $\boldsymbol{\alpha}_{n-1}$ onto the intersection of the p manifolds.

At time step $n+1$, a new input is fed to the model. One of the two following possibilities occurs:

1. *First case*: $\max_{j=1\cdots m} |\kappa(\boldsymbol{u}_{n+1}, \boldsymbol{u}_{w_j})| > \mu_0$

   In this case, $\kappa(\cdot, \boldsymbol{u}_{n+1})$ is not introduced in the dictionary $\mathcal{D}_n$ and the solution vector $\boldsymbol{\alpha}_{n+1}$ is updated according to [20]:

   $$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}_n + \eta \boldsymbol{H}_{n+1}^t (\epsilon \boldsymbol{I} + \boldsymbol{H}_{n+1}\boldsymbol{H}_{n+1}^t)^{-1}(\boldsymbol{d}_{n+1} - \boldsymbol{H}_{n+1}\boldsymbol{\alpha}_n), \tag{35}$$

   where η is called the step-size control parameter and $\epsilon \boldsymbol{I}$ is a regularization factor.

2. *Second case*: $\max_{j=1\cdots m} |\kappa(\boldsymbol{u}_{n+1}, \boldsymbol{u}_{w_j})| \leq \mu_0$

   In this case, $\kappa(\cdot, \boldsymbol{u}_{n+1})$ cannot be represented by the actual kernel functions of the dictionary $\mathcal{D}_n$ and it is introduced as a new element and denoted $\kappa(\cdot, \boldsymbol{u}_{w_{m+1}})$. The resolution of (34) leads to the following recursive solution:

   $$\boldsymbol{\alpha}_{n+1} = \begin{bmatrix} \boldsymbol{\alpha}_n \\ 0 \end{bmatrix} + \eta \boldsymbol{H}_{n+1}^t \left(\epsilon \boldsymbol{I} + \boldsymbol{H}_{n+1}\boldsymbol{H}_{n+1}^t)^{-1}(\boldsymbol{d}_{n+1} - \boldsymbol{H}_{n+1} \begin{bmatrix} \boldsymbol{\alpha}_n \\ 0 \end{bmatrix} \right). \tag{36}$$

The set of recursions (35) and (36) are called the Kernel Affine Projection Algorithm (KAPA).

4.3. Kernel Normalized Least Mean Squares algorithm (KNLMS)

This algorithm is similar to the Kernel Affine Projection Algorithm with a number of manifolds p equal to 1. In this case, the constraint appearing in problem (34) becomes $d_n = \boldsymbol{h}_n^t \boldsymbol{\alpha}_n$ where $\boldsymbol{h}_n$ is a column vector with i-th entry equal to $\kappa(\boldsymbol{u}_n, \boldsymbol{u}_{w_i})$. Relations (35) and (36) become:

1. *First case*: $\max_{j=1\cdots m} |\kappa(\boldsymbol{u}_{n+1}, \boldsymbol{u}_{w_j})| > \mu_0$

   $$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}_n + \frac{\eta}{\epsilon + ||\boldsymbol{h}_{n+1}||^2} \left(d_{n+1} - \boldsymbol{h}_{n+1}^t \boldsymbol{\alpha}_n \right) \boldsymbol{h}_{n+1}. \tag{37}$$

2. *Second case*: $\max_{j=1\cdots m} |\kappa(\boldsymbol{u}_{n+1}, \boldsymbol{u}_{w_j})| \leq \mu_0$

   $$\boldsymbol{\alpha}_{n+1} = \begin{bmatrix} \boldsymbol{\alpha}_n \\ 0 \end{bmatrix} + \frac{\eta}{\epsilon + ||\boldsymbol{h}_{n+1}||^2} \left(d_{n+1} - \boldsymbol{h}_{n+1}^t \begin{bmatrix} \boldsymbol{\alpha}_n \\ 0 \end{bmatrix} \right) \boldsymbol{h}_{n+1}, \tag{38}$$

DOI: 10.1002/acs

with $\boldsymbol{h}_{n+1} = [\kappa(\boldsymbol{u}_{n+1}, \boldsymbol{u}_{w_1}), \cdots, \kappa(\boldsymbol{u}_{n+1}, \boldsymbol{u}_{w_{m+1}})]^t$.

5. EXPERIMENTATIONS

In this section, we illustrate the benefits of dictionary adaptation. To compare results, we choose ν_0, the reference step-size for dictionary adaptation, and μ_0, the coherence parameter, in such a way that the final mean quadratic errors are the same, with and without adaptation, and we compare the final sizes of the dictionaries. Another possibility is to select ν_0 and μ_0 to get almost identical sizes of the dictionaries so we can directly compare the residual quadratic error.

5.1. The Kernel Recursive Least Squares Algorithm (KRLS) performance

Our first experiment consists in predicting the Henon time series generated by:

$$d_n = 1 - \gamma_1 d_{n-1}^2 + \gamma_2 d_{n-2},$$

with $d_0 = -0.3$, $d_1 = 0$, $\gamma_1 = 1.4$, and $\gamma_2 = 0.3$. Modeling is performed with a model of the form $\psi_n(d_{n-1}, d_{n-2})$ and makes the use of a series of 2000 samples. The gaussian kernel is used with parameter $\sigma = 0.35$. To compare our results with the ones in [19], we use the same parameters settings and a coherence criterion $\mu_0 = 0.6$. The adaptation step size is $\nu_0 = 0.05$. Modeling is performed with and without dictionary adaptation (see Figure 4 and Figure 5).

With adaptation, the dictionary size decreases from 17 to 16 elements which corresponds to a decrease of 5.88%. The instantaneous quadratic error is averaged over the last 500 samples. Without adaptation, the mean of the quadratic error is 0.01036 versus 0.001628 with adaptation. This corresponds to a decrease of 84.29% of this error.

5.2. Kernel Affine Projection Algorithm (KAPA)

We now present two other experiments using KAPA algorithm. The first consists in predicting the logistic time series using gaussian and exponential kernels. The second is extracted from [20] and makes the use of the polynomial kernel.

5.2.1. Radial basis kernels

In this experiment, the radial basis kernel is used with the Logistic Map which is a deterministic chaotic non linear model of the form:

$$d_{n+1} = a d_n (1 - d_n),$$

Prepared using acsauth.cls
DOI: 10.1002/acs

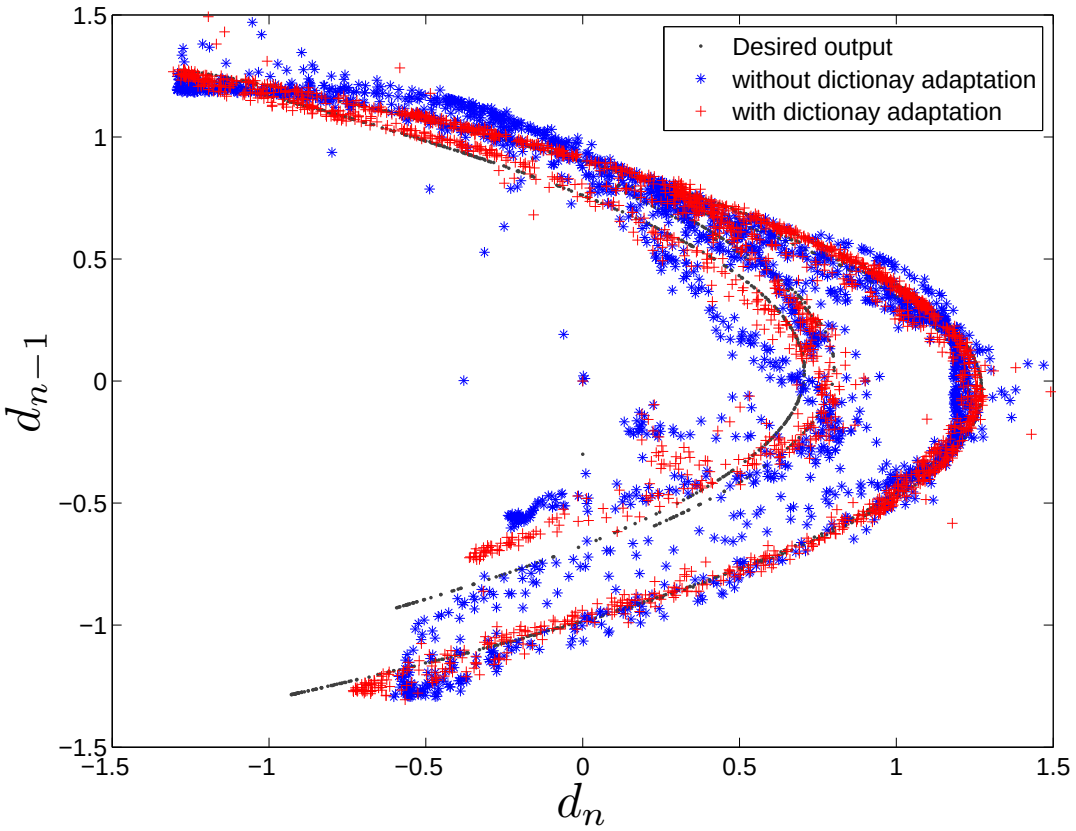


Figure 4. Henon system modeling using KRLS with $\mu_0 = 0.6$ and $\nu_0 = 0.05$.

with initial conditions $a = 4$ and $d_0 = 0.2027$. A zero mean white gaussian noise with standard deviation 0.25 is added to the desired output. Here again the experiment is conducted with and without dictionary adaptation. Note that in the case of radial basis kernels we adapt all the dictionary element using the same gradient step size.

The coherence parameter is $\mu_0 = 0.3$. Results using the gaussian kernel with a bandwidth 0.418, are shown in Figure 6. These results demonstrate that dictionary adaptation (using $\nu_0 = 0.2$) leads to a strong reduction of the mean squared error which is 0.0016778 with adaptation versus 0.0060538 without adaptation. This corresponds to a reduction of 72.285% of the error, with the same final dictionary size $m = 2$.

Results using an exponential kernel with a bandwidth 0.5, are shown in Figure 7. They indicate that dictionary adaptation ($\nu_0 = 0.01$) leads to a mean squared error of 0.0099111 with a dictionary size $m = 2$. Without adaptation, the mean squared error is 0.01601 with the same dictionary size $m = 2$. This means that with the same dictionary size $m = 2$, the quadratic error is reduced with 38.09%.

Prepared using acsauth.cls
DOI: 10.1002/acs

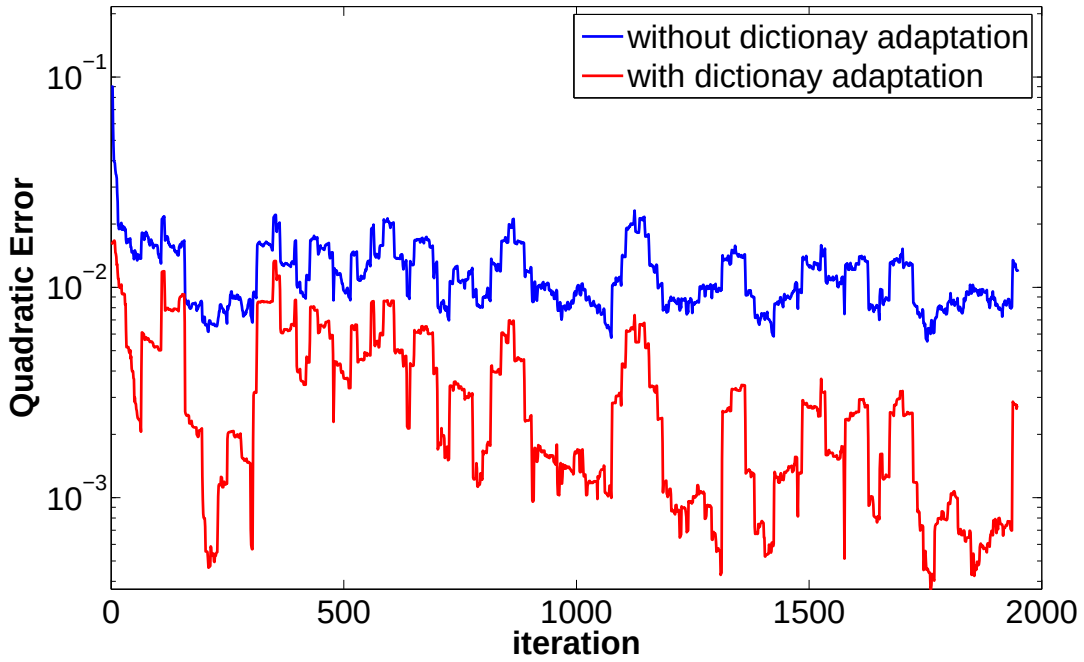


Figure 5. Quadratic error of Henon system modeling using KRLS with $\mu_0 = 0.6$ and $\nu_0 = 0.05$.

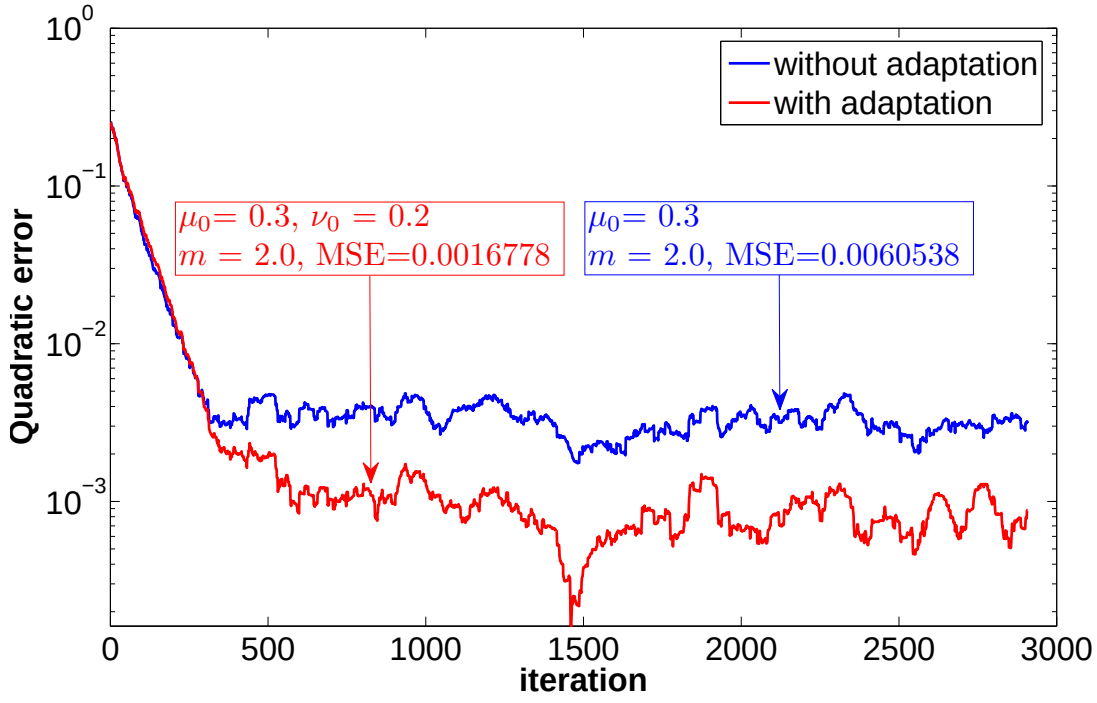


Figure 6. Logistic Map: Quadratic Error - KAPA with gaussian kernel

5.2.2. *Polynomial kernel*

In this experiment,we use the same benchmark parameters setting as in [20]. It consists of a one

Prepared using acsauth.cls
DOI: 10.1002/acs

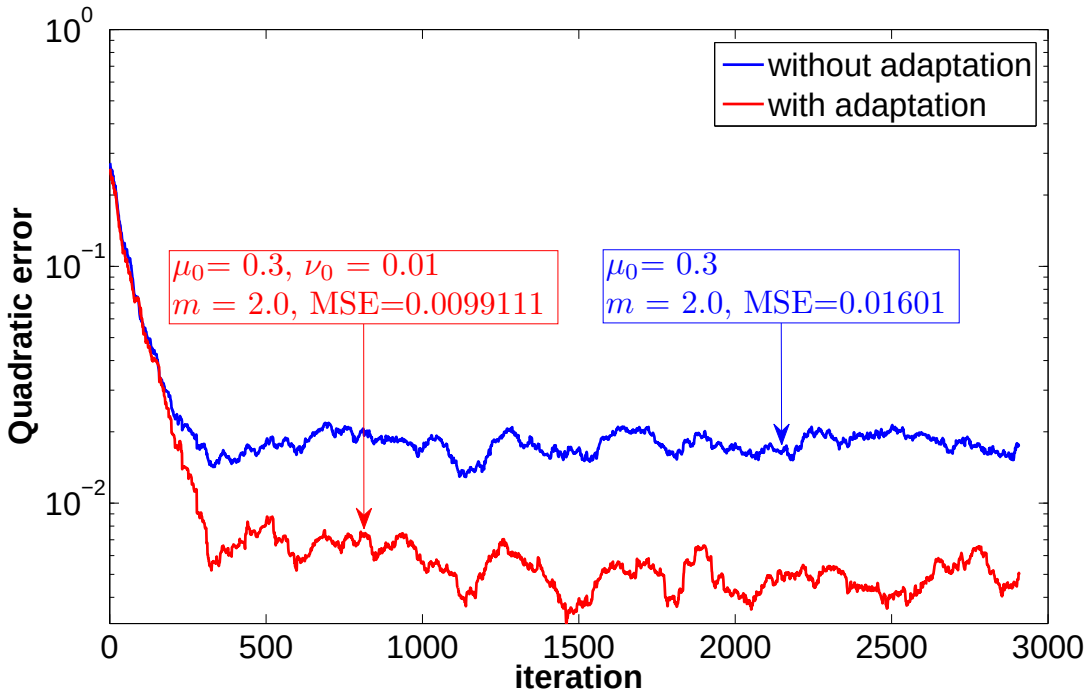


Figure 7. Logistic Map: Quadratic Error - KAPA with exponential kernel

step prediction of the following discrete-time nonlinear dynamical system:

$$\begin{cases} s_n = 1.1\exp(-|s_{n-1}|) + u_n \\ d_n = s_n^2 \end{cases}$$

where u_n and d_n are the input and the desired output, respectively. The data were generated from the initial condition s_0 =0.5. The input was sampled from a zero-mean gaussian distribution with standard deviation 0.25. The system output was corrupted by an additive zero-mean white gaussian noise with standard deviation equal to 1. The model is of the form $\psi_n(u_n, u_{n-1})$.

The KAPA algorithm is used with the following parameters setting: number of manifolds p = 3, step-size control parameter η = 0.01, regularization factor ϵ = 0.07. A set of 200 time series of 3000 samples each is used to compare KAPA and AKAPA [35] using the Normalized Mean Squared Error (NMSE) which is computed over the last 500 samples according to:

$$\text{NMSE} = E\left\{ \frac{\sum_{i=2501}^{3000} (d_n - \psi_n(u_n, u_{n-1}))^2}{\sum_{i=2501}^{3000} d_n^2} \right\}.$$

The expected value is estimated over the 200 realizations of the time series.

We also compute the final size of the dictionary $\overline{m}$ averaged over the 200 realizations. The dictionary adaptation is performed for the element giving the highest value of the norm of the

Prepared using acsauth.cls
DOI: 10.1002/acs

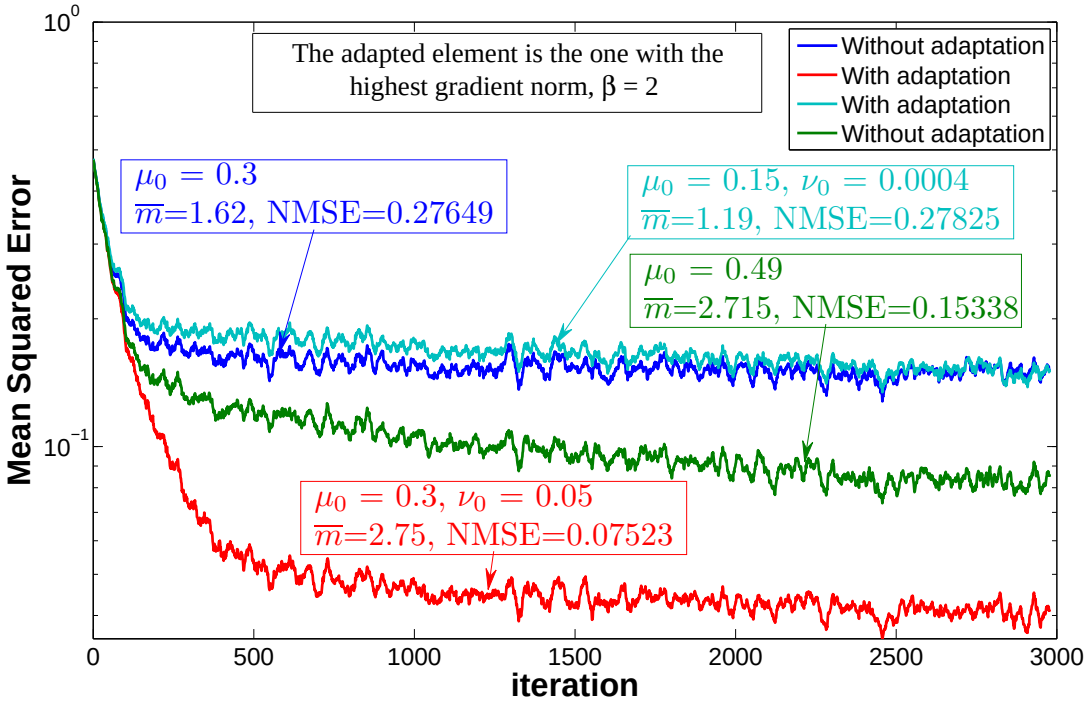


Figure 8. KAPA: Mean Squared Error with and without adaptation - Polynomial kernel ($\beta = 2$)

Table I. KAPA Experimental Setup and Performance, with $\eta = 0.01$, and $\epsilon = 0.07$ - Polynomial kernel

	Third order polynomial kernel $\beta = 2$ Adaptation of the element with the highest norm of the gradient			Second order polynomial kernel $\beta = 3$ Adaptation of the element with the smallest value of $\lvert\alpha_{n,i}\rvert$		
Adaptation	Parameters Settings	$\overline{m}$	NMSE	Parameters Settings	$\overline{m}$	NMSE
Without	μ_0=0.3	1.62	0.27649	μ_0=0.3	2.315	0.21308
Without	μ_0=0.49	2.715	0.15338	μ_0=0.3662	2.915	0.15078
With	μ_0=0.3, ν_0=0.05	2.75	0.07523	μ_0=0.3, ν_0=0.05	2.91	0.12448
With	μ_0=0.15, ν_0=0.0004	1.19	0.27825	μ_0=0.15, ν_0=0.00055	1.735	0.21577

gradient along with a polynomial kernel of degree $\beta = 2$ (see Figure 8). Also, the other strategy to choose the element to adapt which is the one giving the smallest value of $|\alpha_{n,i}|$ was test with a polynomial kernel of degree $\beta = 3$ (see Figure 9).

Using Table I, the following observations can be extracted:

Prepared using acsauth.cls
DOI: 10.1002/acs

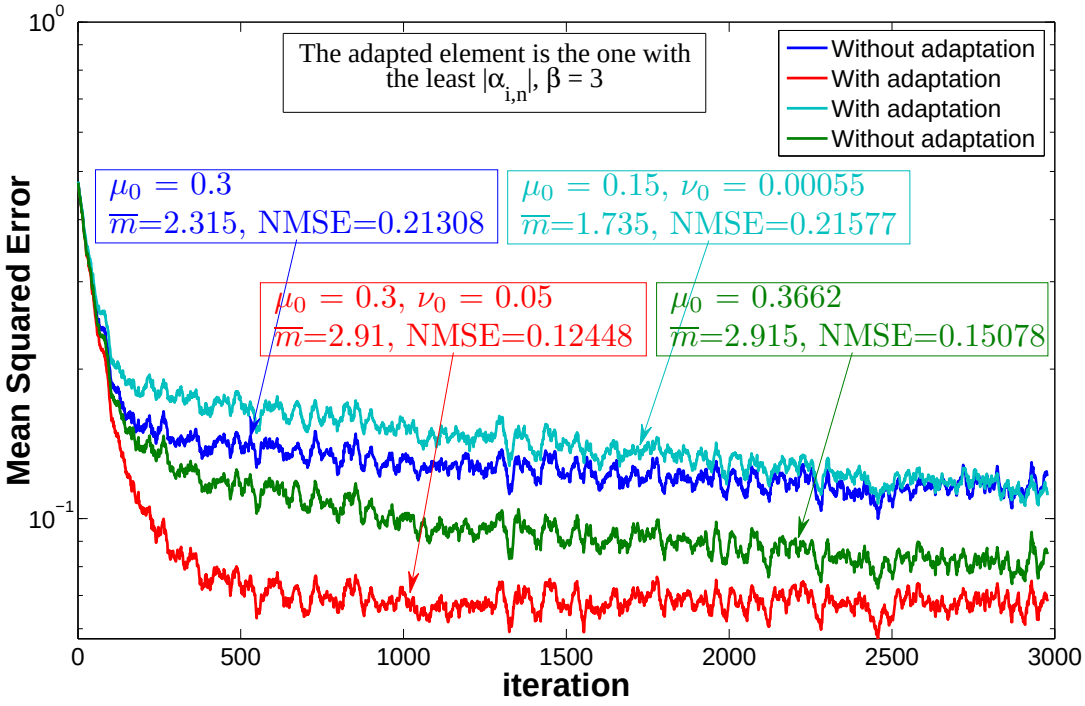


Figure 9. KAPA: Mean Squared Error with and without adaptation - Polynomial kernel ($\beta = 3$)

- for the same average final dictionary size, the NMSE decreases of 50.95% if the adjusted element is the one with the highest gradient norm ($\beta = 2$), and of 17.44% if the adjusted element is the one with the the smallest value of $|\alpha_{n,i}|$ ($\beta = 2$).
- for approximately the same NMSE, the average dictionary size has decreased of 26.54% if the adjusted element is the one with the highest gradient norm ($\beta = 2$), and of 25.05% if the adjusted element is the one with the the smallest value of $|\alpha_{n,i}|$ ($\beta = 2$).

5.3. Kernel Normalized Least Mean Squares algorithm (KNLMS)

In this section we consider the same nonlinear benchmark as proposed in [14]. The desired system output d_n is given by:

$$d_n = (0.8 - 0.5e^{-d_{n-1}^2})d_{n-1} - (0.3 - 0.9e^{-d_{n-1}^2})d_{n-2} + 0.1\sin(d_{n-1}\pi).$$

The initial conditions are (0.1;0.1). The output is corrupted with an additive zero mean white noise with standard deviation equal to 0.1. The model is of the form $\psi_n(d_{n-1}, d_{n-2})$.

The same parameters as in [20] are used, and 200 time series with 3000 samples each are generated. The parameters used for comparison between different cases are the average final size of the dictionary $\overline{m}$ and the Normalized Mean Squared Error (NMSE) calculated over the last 500 samples of each time series and averaged over the 200 realizations. Both exponential and gaussian

Prepared using acsauth.cls
DOI: 10.1002/acs

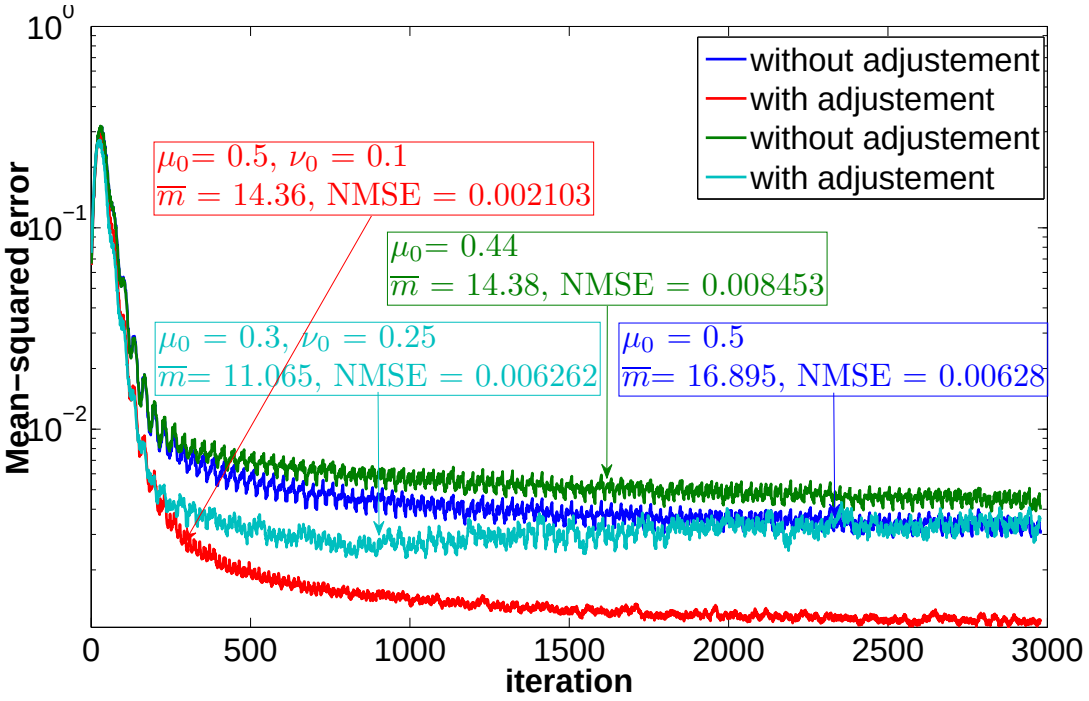


Figure 10. KNLMS: Mean Squared Error with and without adaptation - gaussian kernel

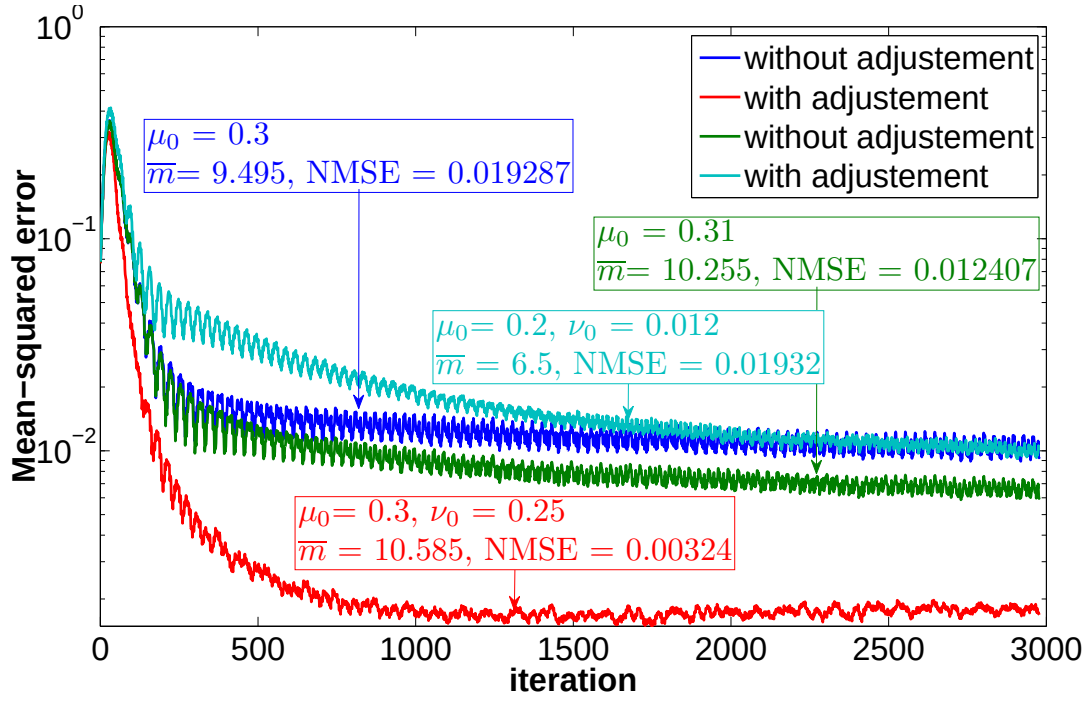


Figure 11. KNLMS: Mean Squared Error with and without adaptation - exponential kernel

kernels are used and the learning curves are shown in Figure 10 and Figure 11. Table II gives a recapitulation for the obtained results and leads to the following observations:

Prepared using acsauth.cls
DOI: 10.1002/acs

Table II. KNLMS Experimental Setup and Performance, with $\eta = 0.09$, and $\epsilon = 0.03$

	Gaussian kernel			Exponential kernel		
Adaptation	Parameters Settings	$\overline{m}$	NMSE	Parameters Settings	$\overline{m}$	NMSE
Without	μ_0=0.5	16.895	0.00628	μ_0=0.3	9.495	0.019287
Without	μ_0=0.44	14.38	0.00845	μ_0=0.31	10.255	0.012407
With	μ_0=0.5, ν_0=0.1	14.36	0.0021	μ_0=0.3, ν_0=0.25	10.585	0.00324
With	μ_0=0.3, ν_0=0.25	11.065	0.006262	μ_0=0.2, ν_0=0.012	6.5	0.01932

- When performing dictionary adaptation, and for the same average final size of the dictionary, NMSE has decreased of 75.148% for the gaussian kernel and of 73.885% for the exponential kernel.
- For the same final average NMSE, the average size of the dictionary has decreased of 34.50% for the gaussian kernel and of 31.54% for the exponential kernel.

Note that for both kernels, algorithm converges faster when performing dictionary adaptation, which is not surprising. KNLMS and KRLS are compared in Figure 12.

5.4. Sun spots prediction using KAPA

We now apply our adaptation algorithm to a real time series which consists of the number of sunspots per month between January 1749 and February 2012. This data set has been obtained from [36]. For this time series, we use the Kernel Affine Projection Algorithm coupled with the gaussian kernel function. The model is of the form $\psi_n(u_{n-1}, u_{n-2}, u_{n-3})$. Using the following parameter settings $\eta = 0.1, \epsilon = 0.07, p = 3,$ the NMSE is calculated over the last 300 samples. Comparing NMSE while obtaining approximately the same final dictionary size $m \approx 536$, without adaptation, setting $\mu_0 = 0.12415$ the NMSE is 0.016839 while with adaptation and setting $\mu_0 = 0.1$ and $\nu_0 = 0.0175$, we observe that the NMSE has decreased to 0.0033112 inducing a reduction of 80.336%. The details are shown in Figure 13.

Prepared using acsauth.cls
DOI: 10.1002/acs

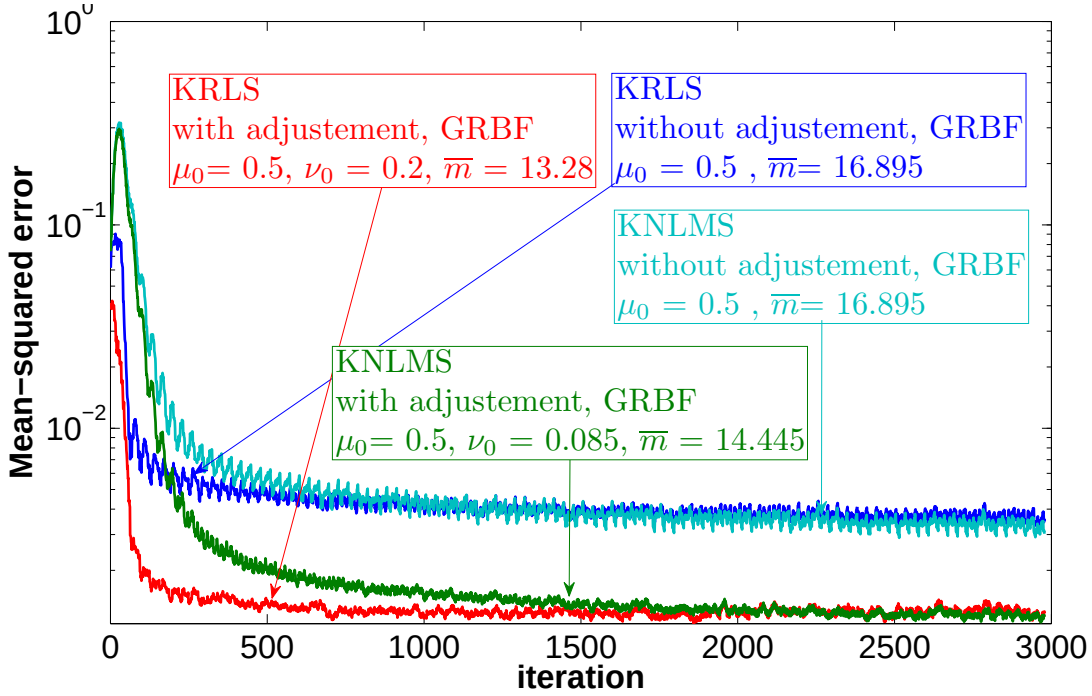


Figure 12. KNLMS versus KRLS - gaussian kernel

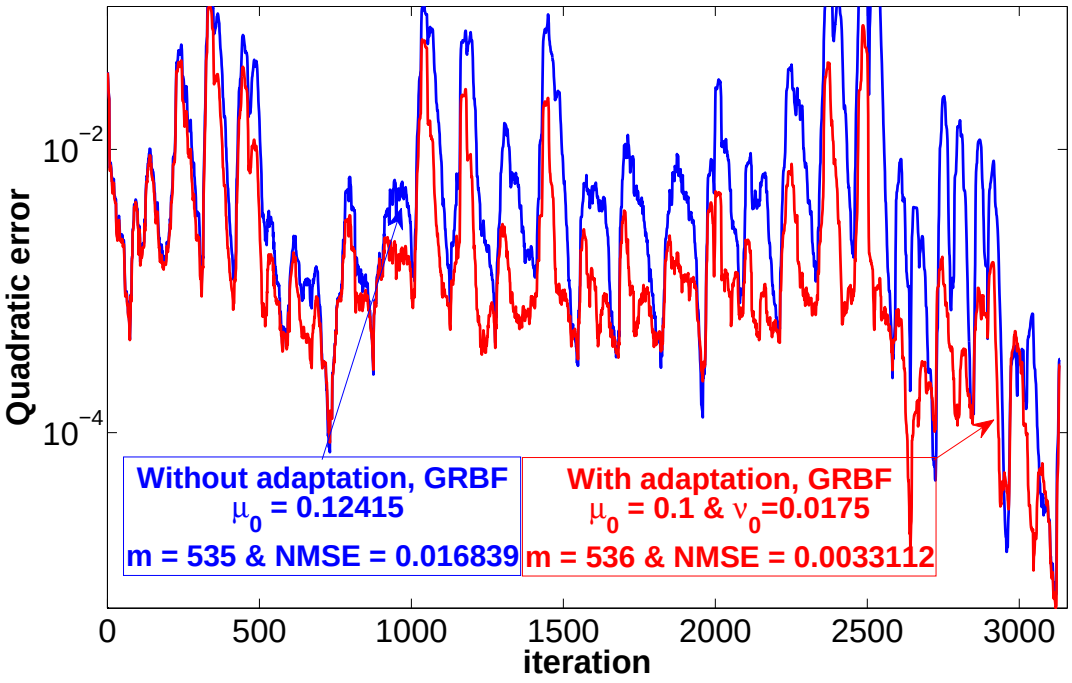


Figure 13. Sunspots prediction using the gaussian kernel

Prepared using acsauth.cls
DOI: 10.1002/acs

6. CONCLUSION

In kernel based adaptive algorithms, model complexity increases with time. Sparsification methods must be used to select some past input vectors (called the dictionary) which appear in the expression of the model. As shown in the literature, several criteria can be used to perform sparsification. However, even in a non stationary environment, previous work never considered the dictionary as a set of parameters that could be adapted in order to further reduce the instantaneous quadratic output error.

In this paper, we have introduced a method for optimizing the dictionary elements on which the model is decomposed, simultaneously with the standard model parameters adaptation. The coherence was the sparsification criterion used. Our gradient-based method allowed iterative fulfillment of the coherence criterion used both for introducing a new element in the dictionary and for adapting the existing dictionary elements. We have presented the complete updating equations in the case of unit-norm gaussian and exponential kernels and the non unit-norm polynomial kernel. We have also shown that there is no universal dictionary updating algorithm, which means that every kernel type requires a specific study. We have compared the results obtained with and without dictionary adaptation, both on simulated and real world data, and we have shown that, using dictionary adaptation, the estimation error can be dramatically reduced (for a similar final size of the dictionary), or the final size of the dictionary can be reduced (for a similar residual estimation error). Obviously, any other compromise can be selected between these two extreme possibilities.

Finally, among the possible extensions of this work, we are working on reducing the computational complexity which is strongly related to the (time varying) number of dictionary elements and we are considering different sparsification criteria such as linear dependence measures.

REFERENCES

1. Aihara S, Bagchi A. Adaptive filtering for stochastic risk premia in bond market. *International journal of innovative computing, information and control* March 2012; **8**(3(B)):2203–2214.
2. Yu Z, Badong C, Jinchun H. Adaptive filtering with adaptive p-power error criterion. *International journal of innovative computing, information and control* April 2011; **7**(4):1725–1737.
3. Schetzen M. *The Volterra and Wiener theories of nonlinear systems*. Wiley, 1980.
4. Wiener N. *Nonlinear Problems in Random Theory (Technology Press Research Monographs)*. The MIT Press, 1966.
5. Ogunfunmi T. *Adaptive Nonlinear System Identification: The Volterra and Wiener Model Approaches*. Springer, 2007.
6. Haykin S. *Neural Networks and Learning Machines (3rd Edition)*. 3rd edn., Prentice Hall, 2008.

Prepared using acsauth.cls
DOI: 10.1002/acs

7. Smola A, Schölkopf B. A tutorial on support vector regression. *Neurocolt technical report*, Royal Holloway College, University of London, UK 1998.
8. Suykens J, van Gestel T, de Brabanter J, deMoor B, Vandewalle J. *Least squares support vector machines*. World Scientific, 2002.
9. Honeine P, Richard C, Bermudez JCM. On-line nonlinear sparse approximation of functions. *Proc. IEEE International Symposium on Information Theory*, Nice, France, 2007; 956–960.
10. Mercer J. Functions of positive and negative type and their connection with the theory of integral equations. *Philos. Trans. Roy. Soc. London* 1909; URL http://en.wikipedia.org/wiki/Mercer's_theorem.
11. Aronszajn N. Theory of reproducing kernels. *Transactions of the American Mathematical Society* 1950; **68**(3):337–404.
12. Aizerman A, Braverman EM, Rozoner LI. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control* 1964; **25**:821–837.
13. Dodd T, Harrison R. A theory of iterative sparse solutions to strict interpolation in reproducing kernel hilbert spaces. *Technical repport.* 2002; **827**.
14. Dodd TJ, Kadirkamanathan V, Harrison RF. Function estimation in hilbert space using sequential projections. *Intell. Control Syst. Signal Process.* 2003; :113–118.
15. Dodd TJ, Nair S, Harrison RF, Kadirkamanathan V, Phonphitakchai S. Sparse,online learning in reproducing kernel hilbert spaces. *IEEE Trans. Signal Processing* 2005; .
16. Phonphitakchai S, Dodd T. Stochastic meta descent in online kernel methods. *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2009. ECTI-CON 2009. 6th International Conference on*, vol. 02, 2009; 690 –693, doi:10.1109/ECTICON.2009.5137142.
17. Engel Y, Mannor S, Meir R. Sparse online greedy support vector regression. *Proceedings of the 13th European Conference on Machine Learning*, ECML '02, Springer-Verlag: London, UK, 2002; 84–96.
18. Engel Y, Mannor S, Meir R. Kernel recursive least squares. *IEEE Transactions on Signal Processing* 2004; **52**:2275–2285.
19. Honeine P, Richard C, Bermudez JCM. Modélisation parcimonieuse non linéaire en ligne par une méthode á noyau reproduisant et un critère de cohérence. *Actes du XXI-ème Colloque GRETSI sur le Traitement du Signal et des Images*, Troyes, France, 2007.
20. Richard C, Bermudez JCM, Honeine P. Online prediction of time series data with kernels. *IEEE trans. on Signal Processing* March 2009; **57**(3):1058–1067.
21. Mercer J. Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society* 1909; **A 209**(441-458):415–446.
22. Schölkopf B, Herbrich R, Smola A. A generalized representer theorem. *COLT/EuroCOLT'01*, 2001; 416–426.
23. Kimeldorf GS, Wahba G. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications* 1971; **33**(1):82–95.
24. Wahba G. *Spline models for observational data*, *CBMS-NSF Regional Conference Series in Applied Mathematics*, vol. 59. Society for Industrial and Applied Mathematics (SIAM): Philadelphia, PA, 1990.
25. Tropp JA. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. Inform. Theory* 2004; **50**:2231–2242.
26. Sun Y, Gomez FJ, Schmidhuber J. On the size of the online kernel sparsification dictionary. *ICML*, icml.cc / Omnipress, 2012.
27. Csat L, Opper M. Sparse representation for gaussian process models. *Advances in Neural Information Processing Systems*, MIT Press, 2001; 444–450.
28. Mallat S, Zhang Z. Matching pursuit with time-frequency dictionaries. *IEEE Transactions on Signal Processing* 1993; **41**:3397–3415.

Prepared using acsauth.cls
DOI: 10.1002/acs

29. Cucker F, Smale S. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society* 2002; **39**:1–49.
30. Ozeki K, Umeda T. An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties. *Electron. Commun. Japan* 1984; **67**(A):1927–1984.
31. Sayed A. *Fundamentals of Adaptive Filtering*. Wiley: New York, 2003.
32. Shams Esfand Abadi M, Mehrdad V, Norouzi M. A family of set-membership affine projection adaptive filter algorithms. *International journal of innovative computing, information and control* February 2012; **8**(2):1313–1326.
33. Liu W, Príncipe J. Kernel affine projection algorithms. *EURASIP Journal on Advances in Signal Processing* 2008; **2008**(1):784 292+, doi:10.1155/2008/784292.
34. Liu W, Principe JC, Haykin S. *Kernel Adaptive Filtering: A Comprehensive Introduction*. 1st edn., Wiley Publishing, 2010.
35. Saide C, Lengelle R, Honeine P, Richard C, Achkar R. Dictionary adaptation for online prediction of time series data with kernels. *Statistical Signal Processing Workshop (SSP), 2012 IEEE*, 2012; 604–607, doi:10.1109/SSP.2012.6319772.
36. Hathaway D. The sunspot cycle, http://solarscience.msfc.nasa.gov/ SunspotCycle.shtml March 2012. URL http://solarscience.msfc.nasa.gov/SunspotCycle.shtml.

DOI: 10.1002/acs