OUTLIER-ROBUST KERNEL HIERARCHICAL-OPTIMIZATION RLS ON A BUDGET WITH AFFINE CONSTRAINTS

Konstantinos Slavakis [1] and Masahiro Yukawa [1]

[1]Affiliation not available

November 8, 2023

Abstract

This paper introduces a non-parametric learning framework to combat outliers in online, multi-output, and nonlinear regression tasks. A hierarchical-optimization problem underpins the learning task: Search in a reproducing kernel Hilbert space (RKHS) for a function that minimizes a sample average ℓ_p-norm ($1 \leq p \leq 2$) error loss on data contaminated by noise and outliers, subject to side information that takes the form of affine constraints defined as the set of minimizers of a quadratic loss on a finite number of faithful data devoid of noise and outliers. To surmount the computational obstacles inflicted by the choice of loss and the potentially infinite dimensional RKHS, approximations of the ℓ_p-norm loss, as well as a novel twist of the criterion of approximate linear dependency are devised to keep the computational-complexity footprint of the proposed algorithm bounded over time. Numerical tests on datasets showcase the robust behavior of the advocated framework against different types of outliers, under a low computational load, while satisfying at the same time the affine constraints, in contrast to the state-of-the-art methods which are constraint agnostic.

———-

OUTLIER-ROBUST KERNEL HIERARCHICAL-OPTIMIZATION RLS ON A BUDGET WITH AFFINE CONSTRAINTS

Konstantinos Slavakis

University at Buffalo, SUNY, USA
Department of Electrical Engineering
Email: kslavaki@buffalo.edu

Masahiro Yukawa

Keio University, Japan
Department of Electronics and Electrical Engineering
Email: yukawa@elec.keio.ac.jp

ABSTRACT

This paper introduces a non-parametric learning framework to combat outliers in online, multi-output, and nonlinear regression tasks. A hierarchical-optimization problem underpins the learning task: Search in a reproducing kernel Hilbert space (RKHS) for a function that minimizes a sample average ℓ_p-norm ($1 \leq p \leq 2$) error loss defined on data contaminated by noise and outliers, under affine constraints defined as the set of minimizers of a quadratic loss on a finite number of faithful data devoid of noise and outliers (side information). To surmount the computational obstacles inflicted by the choice of loss and the potentially infinite dimensional RKHS, approximations of the ℓ_p-norm loss, as well as a novel twist of the criterion of approximate linear dependency are devised to keep the computational-complexity footprint of the proposed algorithm bounded over time. Numerical tests on datasets showcase the robust behavior of the advocated framework against different types of outliers, under a low computational load, while satisfying at the same time the affine constraints, in contrast to the state-of-the-art methods which are constraint agnostic.

Index Terms— Adaptive filtering, kernel, RLS, online learning, outliers.

1. INTRODUCTION

Kernel adaptive filtering (KAF), e.g., [1], has been successful in bringing arguments of classical adaptive filtering [2], reproducing kernel Hilbert spaces (RKHSs) and approximation theory [3, 4], as well as non-parametric methods [5] into online learning [6]. A central role in KAF is played by the kernel recursive least squares (KRLS) [7], by analogy with the pivotal role of RLS in adaptive filtering [2].

Recent KAF efforts revolve around cleansing data from outliers to address their deteriorating effects in a wide variety of learning tasks [8], where outliers are defined as (sparsely appearing) contaminating data that do not adhere to a nominal data-generation model, and are often modeled as random variables (RVs) with non-Gaussian heavy tailed distributions, e.g., α-stable ones [9, 10]. Refraining from using the quadratic error loss, which is notoriously sensitive to non-Gaussian outliers [10, 11], robust KAF approaches include methods which are based mainly on **(i)** the ℓ_p-norm error loss [12, 13], motivated by its beneficial role [14, 15] in classical adaptive filtering [16–18]; and **(ii)** the correntropic loss [19–21], based on the concept of correntropy [22]. Alternative loss functions, designed in a similar way to that of the correntropic loss, can be also found in [23, 24]. Among the previous methods, approaches that employ KRLS-type of iterations can be found in [12, 20, 21].

K. Slavakis was supported by the NSF CIF award 1718796, and M. Yukawa by JSPS KAKENHI grant number JP18H01446.

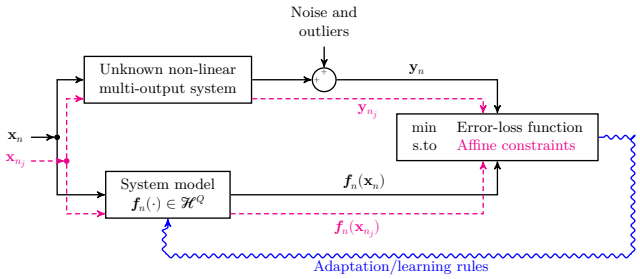


Figure 1: The learning task: With the positive integer $n \in \mathbb{Z}_{>0}$ denoting discrete time, and with the input-output data pair $(\mathbf{x}_n, \mathbf{y}_n)$ becoming available to the user at time n, the goal is to devise an online non-parametric algorithm to learn the unknown non-linear and multi-output system via a reproducing kernel Hilbert space $\mathscr{H}$, where the $Q \times 1$ vector $\mathbf{y}_n$ may carry also noise and outliers. Among all data, $(\mathbf{x}_{n_j}, \mathbf{y}_{n_j})_{j=1}^{J}$ are "clean," with $\mathbf{y}_{n_j}$ being devoid of outliers and noise. Learning the system while rejecting noise and outliers is achieved via the minimization of an error-loss function [see (1)] subject to constraints formed by the faithful data $(\mathbf{x}_{n_j}, \mathbf{y}_{n_j})_{j=1}^{J}$.

This paper introduces a KRLS-type framework for outlier rejection with novel contributions in relation with the state-of-the-art methodologies that are highlighted as follows. A non-linear, multi-output (a.k.a. multi-target or multi-response) regression task [25] is considered (Figure 1). Rather than adapting the classical RLS iterations [2] into KAF, as in [7, 12, 20, 21], the proposed Algorithm 1 stems from the stochastic-approximation framework [26]. Similarly to [12], a sample average ℓ_p-norm error loss is used also here to define the objective function in the optimization problem of the learning task. Nevertheless, apart from the outlier contaminated data and in contrast to [12, 20, 21], the present framework allows also the use of faithful data, devoid of noise and outliers, as side information. Compliance to the faithful data is achieved via an affine constraint set, defined by the set of minimizers of a quadratic loss on the clean data. The set of minimizers of a quadratic loss offers flexibility in the way that the constraint set is incorporated in the proposed Algorithm 1 (Section 3.2). Affinely constrained RLS schemes have already appeared in adaptive filtering [27, 28], but it seems that this is the first time that affine constraints are considered in KRLS-type methods to accommodate side information. The adoption of the ℓ_p-norm error loss and the potential infinite dimensionality of the RKHS yield computational bottlenecks. To surmount those bottlenecks, approximations of the ℓ_p-norm error loss are proposed, and in contrast to [12, 20] where *all* of the incoming data are incorporated in computations and the computational complexity grows unbounded with time, this study proposes also a novel twist of the approximate linear dependency criterion [7] to ensure that the computational complexity of Algorithm 1

stays bounded over time. Numerical tests on datasets show the robust behavior of Algorithm 1 against different types of outliers, under a low computational load, while satisfying at the same time the affine constraints, as opposed to the state-of-the-art methods which are constraint agnostic.

Due to space limitations, the proofs of the subsequent propositions, results on the convergence of the estimates of Algorithm 1 to a solution of the ensemble average of the optimization problem, formulae for the recursive updates in Algorithm 1, and more numerical tests than the ones presented here will be included in the journal version of the paper.

2. THE LEARNING TASK, COMPUTATIONAL BOTTLENECKS, AND APPROXIMATIONS

Available to the user are the real-valued input-output data $(\mathbf{x}_\nu, \mathbf{y}_\nu)_{\nu=1}^n$, $n \in \mathbb{Z}_{>0}$, where n denotes discrete time, the $D \times 1$ vector $\mathbf{x}_n$ stands for the input of an unknown non-linear system (see Figure 1) and the $Q \times 1$ vector $\mathbf{y}_n =: [y_n^{(1)}, \ldots, y_n^{(Q)}]^\top$ for the corresponding output, possibly carrying noise and outliers, with $\top$ denoting vector/matrix transposition. Among all data, $(\mathbf{x}_{n_j}, \mathbf{y}_{n_j})_{j=1}^J$ are clean ($\mathbf{y}_{n_j} =: [y_{n_j}^{(1)}, \ldots, y_{n_j}^{(Q)}]^\top$ is devoid of noise and outliers) and known to the user (w.l.o.g. indices $(n_j)_{j=1}^J$ are assumed known). In other words, $(\mathbf{x}_{n_j}, \mathbf{y}_{n_j})_{j=1}^J$ collect all of the faithful side information about the system which is available to the user. Upon the observation of $(\mathbf{x}_n, \mathbf{y}_n)$, learning/adaptation rules are applied to the modeling system via a feedback path (Figure 1), which operates in a *time-adaptive* or *online* mode and where the learning-rules iteration index coincides with n.

Learning the unknown non-linear system is viewed here as a *non-parametric regression* problem [5], where a vector-valued function $\boldsymbol{f} := (f^{(1)}, \ldots, f^{(Q)}) \in \mathscr{H}^Q$ is sought such that (s.t.) $y_n^{(q)} \approx f^{(q)}(\mathbf{x}_n)$, and where each $f^{(q)}(\cdot)$, $\forall q \in \{1, \ldots, Q\}$, is taken from a user-defined RKHS $\mathscr{H}$ with inner product $\langle \cdot \mid \cdot \rangle_{\mathscr{H}}$, norm $\|\cdot\|_{\mathscr{H}}$, and reproducing kernel $\kappa(\cdot,\cdot) : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ [3]. Space $\mathscr{H}$ may be infinite dimensional; e.g., $\mathscr{H}$ with a Gaussian kernel [4]. Hence, it is desirable that the learning algorithm operates on a computational budget to address the unpleasant computational complexity implications which may be inflicted by the potentially infinite dimensionality of $\mathscr{H}$. Motivated by the sample average of quadratic error losses in RLS [2], the hierarchical-optimization problem which underpins the learning task is

$$\min_{(f^{(1)},\ldots,f^{(Q)})\in\mathscr{H}^Q} \sum_{q=1}^{Q} \frac{1}{\Gamma_n} \sum_{\nu=1}^{n} \gamma^{n-\nu} |y_\nu^{(q)} - f^{(q)}(\mathbf{x}_\nu)|^p + \lambda_{\mathscr{H}} \frac{1}{2} \sum_{q=1}^{Q} \|f^{(q)}\|_{\mathscr{H}}^2$$
$$\text{s.to } (f^{(1)}, \ldots, f^{(Q)}) \in \arg\min_{(f'^{(1)},\ldots,f'^{(Q)})\in\mathscr{H}^Q} \sum_{q=1}^{Q} \sum_{j=1}^{J} [y_{n_j}^{(q)} - f'^{(q)}(\mathbf{x}_{n_j})]^2 , \quad (1)$$

where the $|\cdot|^p$ loss, with $1 \le p \le 2$, is adopted here to combat noise and outliers, since values of $p < 2$ give less importance to large errors $y_\nu^{(q)} - f^{(q)}(\mathbf{x}_\nu)$ than the classical case of $p = 2$. To promote compact notations in the following discussion, let $\varphi_n := \kappa(\mathbf{x}_n, \cdot) \in \mathscr{H}$. The regularizer $(\lambda_{\mathscr{H}}/2)\|\cdot\|_{\mathscr{H}}^2$, with $\lambda_{\mathscr{H}} \ge 0$, is a classical way to avoid overfitting and to constrain $f^{(q)}$ into the linear subspace $\mathcal{S}_n := \operatorname{span} \boldsymbol{\Phi}_n$ of $\mathscr{H}$, spanned by the columns of the $\dim \mathscr{H} \times n$ "matrix" (better, linear operator) $\boldsymbol{\Phi}_n := [\varphi_1, \ldots, \varphi_n]$ [29]. Moreover, $\gamma \in (0, 1)$ is a forgetting factor that is often used in adaptive filtering [2] to penalize the errors of the currently and recently received data heavier than those of the remote past, and $\Gamma_n := \sum_{\nu=1}^n \gamma^{n-\nu}$.

Since $f^{(q)} \in \mathscr{H}$, the reproducing property [3] yields $f^{(q)}(\mathbf{x}_n) = \langle f^{(q)} \mid \kappa(\mathbf{x}_n, \cdot)\rangle_{\mathscr{H}} = \langle f^{(q)} \mid \varphi_n\rangle_{\mathscr{H}} =: f^{(q)\top}\varphi_n$. Problem (1) is separable in $\{f^{(q)}\}_{q=1}^Q$, and thus equivalent to Q problems of the form

$$\min_{f\in\mathscr{H}} \quad \mathscr{L}_n(f) := \frac{1}{\Gamma_n} \sum_{\nu=1}^{n} \gamma^{n-\nu} |y_\nu - \langle f \mid \varphi_\nu\rangle_{\mathscr{H}}|^p + \lambda_{\mathscr{H}} \frac{1}{2} \|f\|_{\mathscr{H}}^2$$
$$\text{s.to} \quad f \in \mathcal{A} := \arg\min_{f'\in\mathscr{H}} \sum_{j=1}^{J} [y_{n_j} - \langle f' \mid \varphi_{n_j}\rangle_{\mathscr{H}}]^2 , \quad (2)$$

where superscript (q) is omitted to avoid clutter in notations.

With $(\mathbf{x}_n, \mathbf{y}_n)$ becoming available to the user at every time instance n, the dimension $\dim \mathcal{S}_n$ of the linear subspace $\mathcal{S}_n$ may become unbounded as time n advances due to the potential infinite dimensionality of $\mathscr{H}$. Proposition 1 below demonstrates that this "curse of dimensionality" inflicts major burdens upon (2), as in the computation of the popular proximal mapping, defined for $\mathscr{L}_n$ as [30]: $\forall f \in \mathscr{H}$ and $\lambda \in \mathbb{R}_{>0}$, $\operatorname{Prox}_{\lambda\mathscr{L}_n}(f) := \arg\min_{f'\in\mathscr{H}} \lambda\mathscr{L}_n(f') + (1/2)\|f - f'\|_{\mathscr{H}}^2$.

Proposition 1. Let h be any vector in $\mathcal{S}_n$ and $\mathbf{h}$ the unique $\dim \mathcal{S}_n \times 1$ vector s.t. $h = \boldsymbol{\mathfrak{B}}_n \mathbf{h}$, where the columns of the $\dim \mathscr{H} \times \dim \mathcal{S}_n$ matrix $\boldsymbol{\mathfrak{B}}_n := [\mathfrak{b}_1^{(n)}, \ldots, \mathfrak{b}_{\dim \mathcal{S}_n}^{(n)}]$ are the basis vectors $\{\mathfrak{b}_j^{(n)}\}_{j=1}^{\dim \mathcal{S}_n}$ of $\mathcal{S}_n$. Let also the $n \times 1$ vector $\boldsymbol{\mathcal{Y}}_n := [y_1, \ldots, y_n]^\top$. Then,

$$\operatorname{Prox}_{\lambda\mathscr{L}_n}(h) = \boldsymbol{\mathfrak{B}}_n \boldsymbol{\mathcal{K}}_n^{-1} [\boldsymbol{\mathcal{C}}_n \boldsymbol{\mathcal{W}}_n(\operatorname{Prox}_{\lambda\mathscr{L}_n}(h)) \boldsymbol{\mathcal{C}}_n^\top + (\lambda^{-1} + \lambda_{\mathscr{H}}) \boldsymbol{\mathcal{K}}_n^{-1}]^{-1} \cdot \left[\boldsymbol{\mathcal{C}}_n \boldsymbol{\mathcal{W}}_n(\operatorname{Prox}_{\lambda\mathscr{L}_n}(h)) \boldsymbol{\mathcal{Y}}_n + \lambda^{-1} \boldsymbol{\mathcal{C}}_n \boldsymbol{\mathcal{D}}_n \mathbf{h}\right] , \quad (3)$$

where the $\dim \mathcal{S}_n \times \dim \mathcal{S}_n$ "kernel matrix" $\boldsymbol{\mathcal{K}}_n := \boldsymbol{\mathfrak{B}}_n^\top \boldsymbol{\mathfrak{B}}_n$, and the $\dim \mathcal{S}_n \times n$ matrix $\boldsymbol{\mathcal{C}}_n$ as well as the $n \times \dim \mathcal{S}_n$ matrix $\boldsymbol{\mathcal{D}}_n$ are s.t. $\boldsymbol{\Phi}_n = \boldsymbol{\mathfrak{B}}_n \boldsymbol{\mathcal{C}}_n$ and $\boldsymbol{\mathfrak{B}}_n = \boldsymbol{\Phi}_n \boldsymbol{\mathcal{D}}_n$. Moreover, the $\nu\nu$th entry of the diagonal $n \times n$ matrix $\boldsymbol{\mathcal{W}}_n(\operatorname{Prox}_{\lambda\mathscr{L}_n}(h))$ is $[\boldsymbol{\mathcal{W}}_n(\operatorname{Prox}_{\lambda\mathscr{L}_n}(h))]_{\nu\nu} := \gamma^{n-\nu}(p/\Gamma_n)|y_\nu - \langle \operatorname{Prox}_{\lambda\mathscr{L}_n}(h) \mid \varphi_\nu\rangle_{\mathscr{H}}|^{p-2}$.

It is evident from (3) that the computation of the $\dim \mathcal{S}_n \times \dim \mathcal{S}_n$ kernel matrix $\boldsymbol{\mathcal{K}}_n$ poses serious problems in terms of complexity and memory requirements, since $\dim \mathcal{S}_n$ may become unbounded as time n advances. Moreover, since $\operatorname{Prox}_{\lambda\mathscr{L}_n}(h)$ appears at both sides of (3), solving (3) may become an arduous large-scale computational task.

To surmount the bottleneck of solving (3), this work uses a linear subspace $\tilde{\mathcal{S}}_n \subset \mathcal{S}_n$ to serve as an approximation of $\mathcal{S}_n$, under the requirement that the dimension of $\tilde{\mathcal{S}}_n$ stays bounded over time: $\dim \tilde{\mathcal{S}}_n \le L_{\tilde{s}}$, $\forall n$, for a user-defined buffer length $L_{\tilde{s}} \in \mathbb{Z}_{>0}$. To this end, vector $\tilde{\varphi}_\nu^{(n)} \in \tilde{\mathcal{S}}_n$ is introduced to serve as an approximation of φ_ν (the precise definition to be given in Section 3.1). Let also the $\dim \mathscr{H} \times n$ matrix $\tilde{\boldsymbol{\Phi}}_n := [\tilde{\varphi}_1^{(n)}, \ldots, \tilde{\varphi}_n^{(n)}]$. The following proposition introduces two approximations of $\mathscr{L}_n$: The weighted quadratic (4a), and (4b) which capitalizes on the first-order information of (4a).

Proposition 2. Consider the approximations of $\mathscr{L}_n$:

$$\tilde{\mathcal{L}}_n^{①}(f) := \frac{1}{2} \sum_{\nu=1}^{n} w_\nu^{(n)} (y_\nu - \langle f \mid \tilde{\varphi}_\nu^{(n)}\rangle_{\mathscr{H}})^2 + \lambda_{\mathscr{H}}^{①} \frac{1}{2} \|f\|_{\mathscr{H}}^2 , \quad (4a)$$

$$\tilde{\mathcal{L}}_n^{②}(f) := \langle f - f_n \mid \sum_{\nu=1}^{n} w_\nu^{(n)} (\langle \tilde{\varphi}_\nu^{(n)} \mid f_n\rangle_{\mathscr{H}} - y_\nu) \cdot \tilde{\varphi}_\nu^{(n)} \rangle_{\mathscr{H}} + \lambda_{\mathscr{H}}^{②} \frac{1}{2} \|f - f_n\|_{\mathscr{H}}^2 + \tilde{\mathcal{L}}_n^{①}(f_n) , \quad (4b)$$

where $w_\nu^{(n)} := \gamma^{n-\nu}(p/\Gamma_n)|y_\nu - \langle f_\nu \mid \varphi_\nu\rangle_{\mathscr{H}}|^{p-2}$, $\mathbf{W}_n$ is the $n \times n$ diagonal matrix whose $\nu\nu$th diagonal entry is $[\mathbf{W}_n]_{\nu\nu} :=$

$w_\nu^{(n)}$, and f_n is the current estimate of the unknown non-linear system. If $\{b_i^{(n)}\}_{i=1}^{\tilde{\mathcal{S}}_n}$ denotes a basis of $\tilde{\mathcal{S}}_n$, with the $\dim \mathscr{H} \times \dim \tilde{\mathcal{S}}_n$ matrix $\mathbf{B}_n := [b_1^{(n)}, \ldots, b_{\dim \tilde{\mathcal{S}}_n}^{(n)}]$, then for any $h \in \tilde{\mathcal{S}}_n$, i.e., $h = \mathbf{B}_n \mathbf{h}$, where $\mathbf{h} \in \mathbb{R}^{\dim \tilde{\mathcal{S}}_n}$,

$$\mathrm{Prox}_{\lambda \tilde{\mathcal{L}}_n^{①}}(h) = \mathbf{B}_n \mathbf{K}_n^{-1} \left[\mathbf{C}_n \mathbf{W}_n \mathbf{C}_n^\top + (\lambda^{-1} + \lambda_{\mathscr{H}}^{①}) \mathbf{K}_n^{-1} \right]^{-1} \cdot [\mathbf{C}_n \mathbf{W}_n \boldsymbol{\mathcal{Y}}_n + \lambda^{-1} \mathbf{C}_n \mathbf{D}_n \mathbf{h}] \,, \tag{5a}$$

$$\mathrm{Prox}_{\lambda \tilde{\mathcal{L}}_n^{②}}(h) = \mathbf{B}_n (\mathbf{h} - \lambda \mathbf{C}_n \mathbf{W}_n \mathbf{C}_n^\top \mathbf{B}_n^\top f_n + \lambda \mathbf{C}_n \mathbf{W}_n \boldsymbol{\mathcal{Y}}_n)/(1 + \lambda \lambda_{\mathscr{H}}^{②}) + \lambda \lambda_{\mathscr{H}}^{②} f_n/(1 + \lambda \lambda_{\mathscr{H}}^{②}) \,, \tag{5b}$$

where $\mathbf{K}_n := \mathbf{B}_n^\top \mathbf{B}_n$, the $n \times \dim \tilde{\mathcal{S}}_n$ matrix $\mathbf{D}_n$ and the $\dim \tilde{\mathcal{S}}_n \times n$ matrix $\mathbf{C}_n$ are s.t. $\mathbf{B}_n = \tilde{\boldsymbol{\Phi}}_n \mathbf{D}_n$ and $\tilde{\boldsymbol{\Phi}}_n = \mathbf{B}_n \mathbf{C}_n$.

Mapping (5b) realizes actually the "steepest-descent" direction of (4a), and it requires no matrix inversion, unlike (5a). However, the computational savings offered by (5b) may render the proposed Algorithm 1 ineffective, especially against large-variance outliers; cf. Section 4.

3. THE ALGORITHM

Algorithm 1 stems from [26]. It is also equipped with the novel module of Section 3.1 to ensure that its computational complexity stays bounded over time in the context of the potentially infinite dimensional $\mathscr{H}$.

3.1. Approximate Linear Dependency on a Budget

To define subspace $\tilde{\mathcal{S}}_n$, the approximate linear dependency (ALD) criterion of [7] is considered, but with a novel twist which ensures the boundedness of the dimension of $\tilde{\mathcal{S}}_n$: $\dim \tilde{\mathcal{S}}_n \leq L_{\tilde{\mathcal{S}}}$, $\forall n$. This condition is satisfied by monitoring the amplitudes of the coefficients of the estimates $f_n^{(q)}$. This feature was not available in the original ALD [7], where the user cannot fix upper bounds on $\dim \tilde{\mathcal{S}}_n$. The definition of $\tilde{\mathcal{S}}_n$ is provided next via induction. To this end, it is assumed that knowledge of $\tilde{\mathcal{S}}_{n-1}$ is available to the user at time n.

The arrival of $\varphi_n = \kappa(\mathbf{x}_n, \cdot) \in \mathcal{S}_n = \mathrm{span}\{\varphi_\nu\}_{\nu=1}^n$ triggers the computation of its metric projection $P_{\tilde{\mathcal{S}}_{n-1}}(\varphi_n)$ onto $\tilde{\mathcal{S}}_{n-1}$, through the basis vectors $\mathbf{B}_{n-1} = [b_1^{(n-1)}, \ldots, b_{\dim \tilde{\mathcal{S}}_{n-1}}^{(n-1)}]$ as $P_{\tilde{\mathcal{S}}_{n-1}}(\varphi_n) = \mathbf{B}_{n-1} \mathbf{K}_{n-1}^{-1} \mathbf{k}_n^{(n-1)}$, where $\mathbf{k}_n^{(n-1)} := \mathbf{B}_{n-1}^\top \varphi_n$ and $\mathbf{K}_{n-1}^{-1} \mathbf{k}_n^{(n-1)} = \arg\min_{\mathbf{c} \in \mathbb{R}^{\dim \tilde{\mathcal{S}}_{n-1}}} \|\varphi_n - \mathbf{B}_{n-1} \mathbf{c}\|_{\mathscr{H}}^2$. The norm of the residual $\varphi_n - P_{\tilde{\mathcal{S}}_{n-1}}(\varphi_n)$ turns out to be equal to $\varpi_n := [\kappa(\mathbf{x}_n, \mathbf{x}_n) - \mathbf{k}_n^{(n-1)\top} \mathbf{K}_{n-1}^{-1} \mathbf{k}_n^{(n-1)}]^{1/2}$. If $\varpi_n \leq \epsilon_{\mathrm{ALD}}$ (Line 9 of Algorithm 1), for a user-defined $\epsilon_{\mathrm{ALD}} \in \mathbb{R}_{>0}$, vector φ_n is not considered to carry sufficient novel information, and $\tilde{\mathcal{S}}_{n-1}$, its basis $\mathbf{B}_{n-1}$, and kernel matrix $\mathbf{K}_{n-1}$ do not change during their updates (Line 10 of Algorithm 1). Moreover, vector $\tilde{\varphi}_n^{(n)} := P_{\tilde{\mathcal{S}}_{n-1}}(\varphi_n)$.

If, however, $\varpi_n > \epsilon_{\mathrm{ALD}}$ (Line 11 of Algorithm 1), then φ_n is considered to carry sufficient novel information to be included in $\tilde{\mathcal{S}}_n$. In the case where there is still space in the buffer, i.e., $\dim \tilde{\mathcal{S}}_{n-1} < L_{\tilde{\mathcal{S}}}$, then $\tilde{\boldsymbol{\Phi}}_n := [\tilde{\boldsymbol{\Phi}}_{n-1}, \varphi_n]$, $\tilde{\mathcal{S}}_n := \mathrm{span}\, \tilde{\boldsymbol{\Phi}}_n$, and $\mathbf{B}_n := [\mathbf{B}_{n-1}, \varphi_n]$. If there is no space in the buffer (Line 14 of Algorithm 1), then φ_n takes the place of $\tilde{\varphi}_{i_*}^{(n-1)}$ and $b_{i_*}^{(n-1)}$ in $\tilde{\boldsymbol{\Phi}}_{n-1}$ and $\mathbf{B}_{n-1}$, respectively (Line 16 of Algorithm 1), where index i_* identifies the smallest contribution of basis vectors in the representation of the current estimates $\{f_n^{(q)}\}_{q=1}^Q$ as

Algorithm 1: Kernel hierarchical-optimization RLS

Data : $(\mathbf{x}_\nu, \mathbf{y}_\nu)_{\nu=1}^n$, $n \in \mathbb{Z}_{>0}$.
Output : $(\boldsymbol{f}_n = (f_n^{(1)}, \ldots, f_n^{(Q)}))_{n \in \mathbb{Z}_{>0}}$.

1 **Initialization**
2 $\quad \tilde{\mathcal{S}}_0 := \mathrm{span}(\tilde{\boldsymbol{\Phi}}_0 := \boldsymbol{\Phi}_{\mathrm{c}})$.
3 $\quad$ **for** $q = 1$ **to** Q **do**
4 $\quad\quad f_{1/2}^{(q)} = T_\alpha^{(q)}(f_0^{(q)})$.
5 $\quad\quad f_1^{(q)} = \mathrm{Prox}_{\lambda \tilde{\mathcal{L}}_0^{l}}(f_{1/2}^{(q)})$, where (5a) is used if $l = ①$, while (5b) if $l = ②$.
6 **for** $n = 1$ **to** $+\infty$ **do**
7 $\quad$ Data $(\mathbf{x}_n, \mathbf{y}_n)$ become available to the user.
8 $\quad$ Compute ϖ_n as in Section 3.1.
9 $\quad$ **if** $\varpi_n \leq \epsilon_{\mathrm{ALD}}$ **then** `// Basis remains unchanged`
10 $\quad\quad$ Update $\tilde{\boldsymbol{\Phi}}_n := [\tilde{\boldsymbol{\Phi}}_{n-1}, P_{\tilde{\mathcal{S}}_{n-1}}(\varphi_n)]$, basis $\mathbf{B}_n = \mathbf{B}_{n-1}$, subspace $\tilde{\mathcal{S}}_n = \tilde{\mathcal{S}}_{n-1}$, $\dim \tilde{\mathcal{S}}_n = \dim \tilde{\mathcal{S}}_{n-1}$, and kernel matrix $\mathbf{K}_n = \mathbf{K}_{n-1}$.
11 $\quad$ **else if** $\varpi_n > \epsilon_{\mathrm{ALD}}$ **then** `// Basis changes`
12 $\quad\quad$ **if** $\dim \tilde{\mathcal{S}}_{n-1} < L_{\tilde{\mathcal{S}}}$ **then** `// There is space in the buffer`
13 $\quad\quad\quad$ Let $\dim \tilde{\mathcal{S}}_n = \dim \tilde{\mathcal{S}}_{n-1} + 1$ and update $\tilde{\boldsymbol{\Phi}}_n := [\tilde{\boldsymbol{\Phi}}_{n-1}, \varphi_n]$, $\mathbf{B}_n := [\mathbf{B}_{n-1}, \varphi_n]$, $\tilde{\mathcal{S}}_n := \mathrm{span}\, \tilde{\boldsymbol{\Phi}}_n$, as well as the kernel matrix and its inverse.
14 $\quad\quad$ **else if** $\dim \tilde{\mathcal{S}}_{n-1} = L_{\tilde{\mathcal{S}}}$ **then** `// Buffer overflows`
15 $\quad\quad\quad$ Identify i_* as in Section 3.1.
16 $\quad\quad\quad$ Let $\dim \tilde{\mathcal{S}}_n = L_{\tilde{\mathcal{S}}}$ and update $\tilde{\boldsymbol{\Phi}}_n := [\tilde{\varphi}_1^{(n-1)}, \ldots, \tilde{\varphi}_{i_*-1}^{(n-1)}, \varphi_n, \tilde{\varphi}_{i_*+1}^{(n-1)}, \ldots, \tilde{\varphi}_{L_{\tilde{\mathcal{S}}}}^{(n-1)}]$, $\mathbf{B}_n := [b_1^{(n-1)}, \ldots, b_{i_*-1}^{(n-1)}, \varphi_n, b_{i_*+1}^{(n-1)}, \ldots, b_{L_{\tilde{\mathcal{S}}}}^{(n-1)}]$, $\tilde{\mathcal{S}}_n := \mathrm{span}\, \tilde{\boldsymbol{\Phi}}_n$, as well as the kernel matrix and its inverse.
17 $\quad\quad\quad$ Set $\mathbf{f}_n^{(q)}[i_*] := 0$ in $f_n^{(q)} = \mathbf{B}_n \mathbf{f}_n^{(q)}$, $\forall q \in \{1, \ldots, Q\}$.
18 $\quad$ **for** $q = 1$ **to** Q **do**
19 $\quad\quad f_{n+1/2}^{(q)} = P_{\tilde{\mathcal{S}}_n}\left[f_{n-1/2}^{(q)} - T_\alpha^{(q)}(f_{n-1}^{(q)}) + T^{(q)}(f_n^{(q)}) \right]$.
20 $\quad\quad f_{n+1}^{(q)} = \mathrm{Prox}_{\lambda \tilde{\mathcal{L}}_n^{l}}(f_{n+1/2}^{(q)})$, where (5a) is used if $l = ①$, while (5b) if $l = ②$.

$i_* := \min\{\arg\min_{i \in \{1, \ldots, L_{\tilde{\mathcal{S}}}\}} \sum_{q=1}^Q (\mathbf{f}_n^{(q)}[i])^2 \cdot \|b_i^{(n-1)}\|_{\mathscr{H}}^2\}$, where $\mathbf{f}_n^{(q)}[i]$ stands for the ith entry of vector $\mathbf{f}_n^{(q)}$ in $f_n^{(q)} = \mathbf{B}_{n-1} \mathbf{f}_n^{(q)}$. Symbol $P_{\tilde{\mathcal{S}}_n}(\cdot)$ stands for the metric projection mapping onto $\tilde{\mathcal{S}}_n$ in Line 19 of Algorithm 1.

3.2. Accommodating the Affine Constraints

Algorithm 1's precursor [26], which was introduced to solve composite and convex stochastic problems s.to (stochastic) affine constraints, as well as its deterministic predecessor [31], accommodates affine constraints via affine and nonexpansive mappings $T : \mathscr{H} \to \mathscr{H}$ [30], whose fixed point set $\mathrm{Fix}\, T := \{f \in \mathscr{H} \,|\, T(f) = f\}$ coincides with the affine constraint.

In the present context, gathering the "clean" data in $\boldsymbol{\Phi}_{\mathrm{c}} := [\varphi_{n_1}, \ldots, \varphi_{n_J}]$ and $\mathbf{y}_{\mathrm{c}}^{(q)} := [y_{n_1}^{(q)}, \ldots, y_{n_J}^{(q)}]^\top$ renders the constraint set in (2) an affine set, as suggested by its equivalent description $\mathcal{A}^{(q)} = \arg\min_{f' \in \mathscr{H}} \|\mathbf{y}_{\mathrm{c}}^{(q)} - \boldsymbol{\Phi}_{\mathrm{c}}^\top f'\|^2$. Consequently, by virtue of the flexibility that affine nonexpansive mappings offer, Proposition 3 below shows several ways via which the constraint $\mathcal{A}^{(q)}$ can be accommodated.

Proposition 3. Let the $J \times J$ kernel matrix $\mathbf{K}_{\Phi_c} := \mathbf{\Phi}_c^\top \mathbf{\Phi}_c$. Then, any of the following $T^{(q)}$ is affine nonexpansive with $\mathrm{Fix}\, T^{(q)} = \mathcal{A}^{(q)}$, and can be thus used in Algorithm 1.

- **(i)** $T^{(q)}(f) = f - (\varrho/\eta)\mathbf{\Phi}_c(\mathbf{\Phi}_c^\top f - \mathbf{y}_c^{(q)})$, $\forall f \in \mathscr{H}$, where $\eta \geq \lambda_{\max}(\mathbf{K}_{\Phi_c})$, with $\lambda_{\max}(\mathbf{K}_{\Phi_c})$ denoting the largest eigenvalue of $\mathbf{K}_{\Phi_c}$, and $\varrho \in (0, 1]$.
- **(ii)** $T^{(q)}(f) = f + \mathbf{\Phi}_c \mathbf{K}_{\Phi_c}^\dagger (\mathbf{y}^{(q)} - \mathbf{\Phi}_c f)$, $\forall f \in \mathscr{H}$, where superscript $\dagger$ denotes the pseudoinverse of a matrix.
- **(iii)** $T^{(q)}(f) = (\mathbf{\Phi}_c \mathbf{\Phi}_c^\top + \mu\, \mathrm{Id})^{-1}(\mu f + \mathbf{\Phi}_c \mathbf{y}^{(q)})$, $\forall f \in \mathscr{H}$ and $\mu \in \mathbb{R}_{>0}$. In the case where $f = \mathbf{\Phi}_c \mathbf{f}$, for some $J \times 1$ vector $\mathbf{f}$, then $T^{(q)}(f) = \mathbf{\Phi}_c(\mathbf{K}_{\Phi_c} + \mu \mathbf{I})^{-1}(\mu \mathbf{f} + \mathbf{y}^{(q)})$.

In Algorithm 1, $T_\alpha^{(q)} := \alpha T^{(q)} + (1-\alpha)\,\mathrm{Id}$, for $\alpha \in [0.5, 1)$. It is also assumed that $\mathrm{span}\, \mathbf{\Phi}_c \subset \tilde{\mathcal{S}}_n$, $\forall n$. Since time instances $(n_j)_{j=1}^J$, where the clean data are observed, are known to the user, the previous condition can be guaranteed, without any loss of generality, by initializing $\tilde{\mathcal{S}}_0 := \mathrm{span}(\tilde{\mathbf{\Phi}}_0 := \mathbf{\Phi}_c)$ (Line 2 in Algorithm 1), and by refraining from removing any of the columns of $\mathbf{\Phi}_c$ via index i_* in Section 3.1.

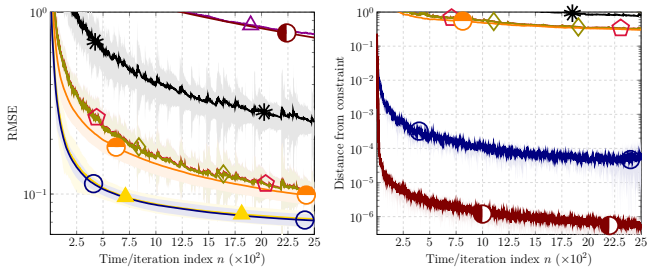
4. NUMERICAL TESTS

Algorithm 1 is compared with KRLS [7], the state-of-the-art KRLP [12] and KRMMC [20], as well as [32–34], whose kernel versions with the criterion of Section 3.1 were crafted specifically for these tests. In [32–34], outlier vectors become parameters to be estimated. The software code was written in Julia [35], while α-stable noise was generated by [36].

The number J of clean data is set equal to 1, and the version of Proposition 3(ii) is chosen for $T^{(q)}$ in Algorithm 1. Performance is measured by the following metrics on $N_t = 500$ noise- and outlier-free test data $(\mathbf{x}_{t,k}, \mathbf{y}_{t,k})_{k=1}^{N_t}$: **(i)** Root mean square error $\mathrm{RMSE}(n) := [(1/N_t)\sum_{k=1}^{N_t} \|\mathbf{y}_{t,k} - \boldsymbol{f}_n(\mathbf{x}_{t,k})\|^2]^{1/2}$; **(ii)** distance $(1/Q)\sum_{q=1}^{Q} \|f_n^{(q)} - P_{\mathcal{A}^{(q)}}(f_n^{(q)})\|_{\mathscr{H}}$ from constraint, where $P_{\mathcal{A}^{(q)}}$ is the metric projection mapping onto the affine set $\mathcal{A}^{(q)} := \{f^{(q)} \in \mathscr{H} \,|\, f^{(q)}(\mathbf{x}_{n_1}) = y_{n_1}^{(q)}\}$; and **(iii)** computational time per iteration n. Multiple independent tests were run and average results are reported in Figures 2 and 3.

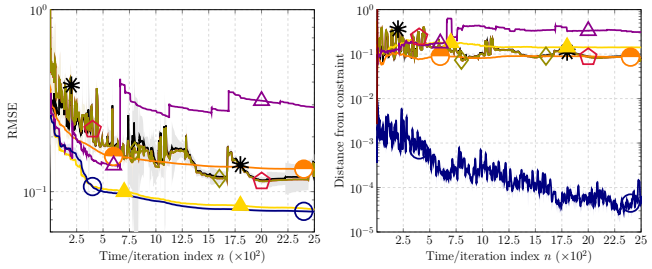
Synthetic data were generated by $\mathbf{y}_n := \mathrm{sinc}(\mathbf{L}\mathbf{x}_n) + \mathbf{o}_n + \mathbf{n}_n$, $\forall n$, with $(D, Q) = (2, 5)$ (Figure 2). The entries of $\mathbf{x}_n$ are realizations of independent and identically distributed (i.i.d.) uniform RVs with values in $[-1, 1]$, zero mean and variance σ_x^2. The entries of the $Q \times D$ matrix $\mathbf{L}$ are generated by i.i.d. normal RVs of zero mean and unit variance, $\mathrm{sinc}(\cdot) := \sin(\cdot)/(\cdot)$ is applied entry-wise, and the entries of noise $\mathbf{n}_n$ are realizations of i.i.d. normal RVs of zero mean and variance σ_n^2. Each entry of the outlier vector $\mathbf{o}_n$ is modeled as the product $o_B \cdot o_U$ of two independent RVs, where $o_B \in \{0, 1\}$ is a Bernoulli(p_B) RV with $p_B = \mathbb{P}(o_B = 1) = 0.2$ (sparse $\mathbf{o}_n$), and o_U is uniformly distributed, with zero mean and variance σ_o^2. In Figure 2, $\sigma_x^2/\sigma_n^2 = 20$dB and $\sigma_o^2/\sigma_n^2 = 30$dB. A Gaussian kernel with variance 0.5 was used and $L_{\tilde{\mathcal{S}}} = 70$. Figure 2a shows that Algorithm 1 with (5a) achieves identical performance with KRLP [12] in much smaller time per n. By virtue of Section 3.1, the computational time per iteration stays fixed for Algorithm 1, in contrast to KRLP and KRMMC where time grows unbounded as n advances. Using only the first-order information in (5b) does not seem to be an effective strategy in the case of large-variance outliers.

Data [37] refer to an inverse dynamics problem for a 7 degrees-of-freedom SARCOS anthropomorphic robot arm.



(a) RMSE (b) Distance from constraint

Figure 2: Synthetic data. Curve markers: KRLS [7]: ✳, [32]: ⬠, [33]: ◇, [34]: ◓, KRLP [12]: ▲, KRMMC [20]: △, Algorithm 1 (5a): ◯, Algorithm 1 (5b): ◐. Average computational times per iteration n in secs are: KRLS: 2.51×10^{-3}, [32]: 2.72×10^{-3}, [33]: 9.51×10^{-3}, [34]: 1.38×10^{-4}, KRLP [12]: 4.30×10^{-1}, KRMMC [20]: 4.29×10^{-1}, Algorithm 1 (5a): 1.62×10^{-2}, Algorithm 1 (5b): 7.34×10^{-4}.



(a) RMSE (b) Distance from constraint

Figure 3: SARCOS data [37]. Curve markers follow those of Figure 2. Average computational times per iteration n in secs are: KRLS: 2.47×10^{-2}, [32]: 2.59×10^{-2}, [33]: 2.85×10^{-2}, [34]: 5.32×10^{-4}, KRLP [12]: 6.41×10^{-1}, KRMMC [20]: 6.39×10^{-1}, Algorithm 1 (5a): 2.50×10^{-1}. Algorithm 1 (5b) diverged, and hence, its performance is not shown here.

The task is to map $(D = 21)$-dimensional vectors (7 positions, 7 velocities, and 7 accelerations) to their corresponding $Q = 7$ joint torques. Outliers are synthetically inserted into the data by adding to each output entry values of an RV with an α-stable distribution [9] and the following parameters: index/stability: 1.5, skewness: 0.5, location: 0, and scale: 10^{-2} [36]. A Gaussian kernel with variance 0.1 was used and $L_{\tilde{\mathcal{S}}} = 150$. Similarly to Figure 2, Figure 3 shows that the RMSE performance of Algorithm 1 with (5a) reaches and even exceeds that of KRLP with smaller computational complexity and memory footprints.

To conclude, KRMMC [20] performed poorly in cases of large-variance outliers, but performed similarly to KRLP and Algorithm 1 with (5a) under small/moderate-variance outliers (due to space limitations, these tests will be detailed elsewhere). Extensive tests have showed that Algorithm 1 with (5b) achieved competitive performance in cases of small-variance outliers, but performed poorly otherwise. In all tests, regardless of the size of the variance of the outliers, Algorithm 1 with (5a) achieved the best RMSE performance among all competing methods, matched, and even exceeded in some cases, the RMSE performance of KRLP [12], with a bounded computational complexity over time, while complying with the side information, in contrast to the constraint-agnostic methods [7, 12, 20, 32–34].

REFERENCES

[1] M. Yukawa, "Multikernel adaptive filtering," *IEEE Trans. Signal Process.*, vol. 60, no. 9, pp. 4672–4682, Sep. 2012.

[2] A. H. Sayed, *Adaptive Filters*. Hoboken: New Jersey: John Wiley & Sons, 2008.

[3] N. Aronszajn, "Theory of reproducing kernels," *Trans. American Math. Society*, vol. 68, no. 3, pp. 337–404, 1950.

[4] B. Schölkopf and A. J. Smola, *Learning with Kernels*. MIT Press, 2001.

[5] M. P. Wand and M. C. Jones, *Kernel Smoothing*, ser. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 1994.

[6] S. Shalev-Shwartz, "Online learning and online convex optimization," *Foundations and Trends in Machine Learning*, vol. 4, no. 2, pp. 107–194, Mar. 2012.

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

[8] R. J. Adler, R. E. Feldman, and M. S. Taqqu, Eds., *A Practical Guide to Heavy Tails: Statistical Techniques and Applications*. USA: Birkhauser Boston Inc., 1998.

[9] G. Samoradnitsky and M. S. Taqqu, *Stable Non-Gaussian Random Processes: Stochastic Models with Infinite Variance*. Chapman and Hall/CRC, 1994.

[10] M. Shao and C. L. Nikias, "Signal processing with fractional lower order moments: Stable processes and their applications," *Proceedings IEEE*, vol. 81, no. 7, pp. 986–1010, 1993.

[11] P. J. Rousseeuw and A. M. Leroy, *Robust Regression and Outlier Detection*. New York: John Wiley & Sons, 1987.

[12] W. Ma, J. Duan, W. Man, H. Zhao, and B. Chen, "Robust kernel adaptive filters based on mean p-power error for noisy chaotic time series prediction," *Engineering Applications of Artificial Intelligence*, vol. 58, pp. 101–110, 2017.

[13] B. Chen, L. Xing, X. Wang, J. Qin, and N. Zheng, "Robust learning with kernel mean p-power error loss," *IEEE Trans. Cybernetics*, vol. 48, no. 7, pp. 2101–2113, 2018.

[14] C. Gentile, "The robustness of the p-norm algorithms," *Machine Learning*, vol. 53, pp. 265–299, 2003.

[15] E. E. Kuruoğlu, P. J. W. Rayner, and W. J. Fitzgerald, "Least ℓ_p-norm impulsive noise cancellation with polynomial filters," *Signal Processing*, vol. 69, no. 1, pp. 1–14, 1998.

[16] S.-C. Pei and C.-C. Tseng, "Least mean p-power error criterion for adaptive FIR filter," *IEEE J. Selected Areas in Communications*, vol. 12, no. 9, pp. 1540–1547, 1994.

[17] M. Belge and E. L. Miller, "A sliding window RLS-like adaptive algorithm for filtering alpha-stable noise," *IEEE Signal Process. Letters*, vol. 7, no. 4, pp. 86–89, 2000.

[18] A. Navia-Vázquez and J. Arenas-García, "Combination of recursive least p-norm algorithms for robust adaptive filtering in alpha-stable noise," *IEEE Trans. Signal Process.*, vol. 60, pp. 1478–1482, 2012.

[19] Z. Wu, J. Shi, X. Zhang, W. Ma, and B. Chen, "Kernel recursive maximum correntropy," *Signal Processing*, vol. 117, pp. 11–16, 2015.

[20] B. Chen, X. Wang, N. Lu, S. Wang, J. Cao, and J. Qin, "Mixture correntropy for robust learning," *Pattern Recognition*, vol. 79, pp. 318–327, 2018.

[21] M. S. Duarte and G. A. Barreto, "Online sparse correntropy kernel learning for outlier-robust system identification," *IFAC-PapersOnLine*, vol. 52, no. 1, pp. 430–435, 2019.

[22] I. Santamaría, P. P. Pokharel, and J. Príncipe, "Generalized correlation function: Definition, properties, and application to blind equalization," *IEEE Trans. Signal Process.*, vol. 54, pp. 2187–2197, 2006.

[23] B. Chen, L. Xing, B. Xu, H. Zhao, N. Zheng, and J. C. Príncipe, "Kernel risk-sensitive loss: Definition, properties and application to robust adaptive filtering," *IEEE Trans. Signal Process.*, vol. 65, no. 11, pp. 2888–2901, 2017.

[24] W. Shi, K. Xiong, and S. Wang, "Multikernel adaptive filters under the minimum Cauchy kernel loss criterion," *IEEE Access*, vol. 7, pp. 120 548–120 558, 2019.

[25] H. Borchani, G. Varando, C. Bielza, and P. Larranaga, "A survey on multi-output regression," *WIREs Data Mining and Knowledge Discovery*, vol. 5, pp. 216–233, 2015.

[26] K. Slavakis, "The stochastic Fejér-monotone hybrid steepest descent method and the hierarchical RLS," *IEEE Trans. Signal Process.*, vol. 67, no. 11, pp. 2868–2883, Jun. 2019.

[27] L. S. Resende, J. M. T. Romano, and M. G. Bellanger, "A fast least-squares algorithm for linearly constrained adaptive filtering," *IEEE Trans. Signal Process.*, vol. 44, no. 5, pp. 1168–1174, 1996.

[28] R. Arablouei and K. Doğançay, "Reduced-complexity constrained recursive least-squares adaptive filtering algorithm," *IEEE Trans. Signal Process.*, vol. 60, no. 12, pp. 6687–6692, 2012.

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

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

[31] K. Slavakis and I. Yamada, "Fejér-monotone hybrid steepest descent method for affinely constrained and composite convex minimization tasks," *Optimization*, vol. 67, no. 11, pp. 1963–2001, 2018.

[32] S. Farahmand and G. B. Giannakis, "Robust RLS in the presence of correlated noise using outlier sparsity," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 3308–3313, Jun. 2012.

[33] L. Xiao, M. Wu, J. Yang, and J. Tian, "Robust RLS via the nonconvex sparsity prompting penalties of outlier components," in *IEEE ChinaSIP*, Jul. 2015, pp. 997–1001.

[34] K. Slavakis and S. Banerjee, "Robust hierarchical-optimization RLS against sparse outliers," *IEEE Signal Process. Letters*, vol. 27, pp. 171–175, 2020.

[35] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing," *SIAM Review*, vol. 59, no. 1, pp. 65–98, 2017, https://julialang.org.

[36] [SW], *Pylevy*, URL: https://github.com/josemiotto/pylevy.

[37] *SARCOS data*. [Online]. Available: http://www.gaussianprocess.org/gpml/data/.