Pattern Recognition Letters 105 (2018) 200–206

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Improving battery voltage prediction in an electric bicycle using altitude measurements and kernel adaptive filters

Felipe Tobar[a,*], Iván Castro[b], Jorge Silva[b], Marcos Orchard[b]

[a] Center for Mathematical Modeling, Universidad de Chile, Beauchef 851, Santiago 8370456, Chile
[b] Electrical Engineering Department, Universidad de Chile, Tupper 2007, Santiago 8370451, Chile

ARTICLE INFO

Article history:
Available online 6 September 2017

JEL classification:
68T05
68T10

Keywords:
Learning and adaptive systems
Kernel adaptive filter
Kernel methods
Data-driven prediction
Signal processing
Electric transportation

ABSTRACT

The time-varying nature of consumption patterns is critical in the development of reliable electric vehicles and real-time schemes for assessing energy autonomy. Most of these schemes use battery voltage observations as a primary source of information and neglect variables external to the vehicle that affect its autonomy and help to characterise the behaviour of the battery as main energy storage device. Using an electric bicycle as case study, we show that the incorporation of external variables (e.g., altitude measurements) improves predictions associated with evolution of the battery voltage in time. We achieve this by proposing a novel kernel adaptive filter for multiple inputs and with a data-dependent dictionary construction. This allows us to model the dependency between battery voltage and altitude variations in a sequential manner. The proposed methodology combines automatic discovery of the relationship between voltage and altitude from data, and a kernel-based voltage predictor to address an important issue in reliability of electric vehicles. The proposed method is validated against a standard kernel adaptive filter, fixed linear filters and adaptive linear filters as baselines on the short- and long-term prediction of real-world battery voltage data.

1. Introduction

The development of rechargeable lithium-ion batteries has enabled the use of electrical equipment such as smartphones, cordless power tools and electric vehicles, without the negative externalities associated with petrol, coal, and disposable batteries [23]. To evaluate the autonomy and response of a battery, the estimation of its remaining amount of energy (i.e., the state-of-charge) is crucial but challenging, since the state-of-charge cannot be measured directly but can only be estimated through related variables such as voltage, discharge current and temperature [14]. Online estimation of the state-of-charge also allows us to evaluate if the battery is able to accomplish a task before executing it under variable user profiles and in a safe manner, for instance, a monitoring module for a battery in an electric car should inform the user whether it is safe to overtake another vehicle at the necessary speed.

The physics behind lithium-ion batteries are well understood [23], therefore, it is possible to construct a mathematical model for the inter-dependencies of their state variables (e.g., voltage, current, temperature) under idealised conditions. However, when unexpected operating conditions come into place and the battery needs to fulfil the requirements imposed by a human operator, there is no clear mathematical model governing the relationship between the battery state variables and the environment; a fact that rules out the use of model-based estimation techniques such as the (extended) Kalman filter or the particle filter [1]. For example, an electric bicycle behaves differently depending on the rider's weight, the slope of the ground, whether she or he is navigating in a traffic jam or freely in an open road, or during summer or winter. These conditions certainly change the dynamics of the battery, and although as humans we can give a qualitative assessment as to how the battery might behave, deriving a sound mathematical model describing these scenarios is beyond our reach. In order to circumvent the construction of a theory-based deductive mathematical model for the battery and its environment, we (i) use sensing technologies to measure variables that are observable, both from the environment and the battery itself, then (ii) consider an adaptive algorithm to *filter out* the noise from the measurements, to finally (iii) adopt a kernel-based approach to *learn* hidden and complex structure in the data and *forecast* the voltage signal.

Finding hidden relationships among variables by using observed data for forecasting purposes is at the heart of Signal Processing and Machine Learning—see [12] and [[15], pp. 118–122]. In particular, we will consider kernel adaptive filters [10], a class of nonlinear adaptive filters [5] that operate by transforming noisy observa-

* Corresponding author.
E-mail address: ftobar@dim.uchile.cl (F. Tobar).

http://dx.doi.org/10.1016/j.patrec.2017.09.009
0167-8655/

tions into high-dimensional features, all to perform predictions by linearly combining those features. The predictive ability of kernel adaptive filters (KAFs) stems from the expressiveness of the nonlinear transformation considered and the fact that the combination of features is *fully learnt from the data*. In the last decade, KAFs have proven successful on online data-processing tasks due to both its reduced computational complexity and ability to identify complex relationships between multivariate data in an adaptive manner as shown by [9,16,22,24] and [21]. KAF applications in energy storage are still few and far between, in fact, machine learning approaches to battery management are novel and have not addressed the relationship between the battery state variables and the environment [7,17].

The contribution of this work is twofold: First, we propose a novel kernel adaptive filter which extends the kernel least mean square filter proposed by [9] and [16] to cater for multiple inputs, together with a signal-dependent dictionary construction. Second, we validate the proposed algorithm in a real-world case study to improve the forecast of an electric bicycle battery voltage using measurements of the altitude of the bicycle; this is an applied contribution with practical impact in the monitoring of energy-storage devices and can also be understood as a proof-of-concept for implementation on more general settings.

For the rest of the paper, we give an overview of KAFs followed by our proposed algorithmic extensions, a description of both the equipment used and experimental settings, and a presentation of the proposed kernel algorithm. Lastly, we compare our method to existing approaches on experiments using real-world data and analyse our findings.

2. Existing kernel adaptive filters and proposed extensions

Let us consider the discrete-time, real-valued, stochastic process $X_{t\in\mathbb{N}}$, where we denote the observation of the random variable X_t by $x_t \in \mathbb{R}$, and the observation sequence $\mathbf{x}_t = [x_t, \dots, x_{t-(d-1)}]^T \in \mathbb{R}^d$ for a given order $d \in \mathbb{N}$—notice that we use lowercase letters for scalars and boldface letters for vectors. The prediction problem involves estimating X_{t+1} by only using the sequence $\mathbf{x}_t$, in an online fashion, where limited knowledge of the statistics of the process $X_{t\in\mathbb{N}}$ is available *a priori*.

We will address the prediction problem using kernel adaptive filters [10], a class of nonlinear adaptive filters [5]. KAFs proceed by implementing a sequential, supervised-learning, algorithm based on support vector regression machines [3], to learn the relationship between $[X_t, \dots, X_{t-(d-1)}]^T$ and X_{t+1}. Specifically, a KAF builds a set of *support vectors* $\mathcal{D}_t = \{\mathbf{s}_1, \dots, \mathbf{s}_{N_t}\}$ and *weights* $\mathcal{W}_t = \{w_1, \dots, w_{N_t}\}$, encompassing information about the observed trajectories of the process $X_{t\in\mathbb{N}}$; thus, when a new trajectory $\mathbf{x}_t$ is observed, the algorithm assesses the similarity between $\mathbf{x}_t$ and each dictionary member (or support vector) through a kernel function to estimate X_{t+1} as a linear combination of the kernel evaluations.[1]

The KAF prediction can be denoted by

$$\hat{X}_{t+1} = \sum_{\mathbf{s}_j \in \mathcal{D}_t} w_j K(\mathbf{s}_j, \mathbf{x}_t) \tag{1}$$

where

- $\hat{X}_{t+1}$ is the prediction of the random variable X_{t+1},
- the set of support vectors $\mathcal{D}_t = \{\mathbf{s}_1, \dots, \mathbf{s}_{N_t}\}$ is referred to as *Dictionary*,
- w_j is the weight corresponding to the support vector $\mathbf{s}_j$, and
- $K(\mathbf{s}_j, \mathbf{x}_t)$ is the kernel (or similarity) evaluation between support vector $\mathbf{s}_j$ and input $\mathbf{x}_t$.

The adaptivity of the KAF comes from learning both the weights and the dictionary in an online manner. The weight update rule is inherited from classic linear adaptive filters [5], these give rise to a plethora of KAF algorithms including the kernel ridge regression [3], kernel recursive least squares [4], extended kernel recursive least squares [8], and kernel state-space models [21]. We will consider the kernel least mean square (KLMS) variant in our experiments due to its advantages for online implementation: low computational complexity and ability to quickly adapt to nonstationary signals [9,16].

We next propose two extensions of the KAF setting to cater for the multivariate and nonstationary data that will be used in our experiments, that is, a kernel designed for multiple inputs and a data-dependent method for constructing the dictionary.

2.1. A dedicated kernel for multiple inputs

The kernel choice first involves finding a similarity function, we consider the Gaussian kernel [[18], Section 2.3] defined by

$$K_l(\mathbf{s}_j, \mathbf{x}_t) = \exp\left(-\frac{||\mathbf{s}_j - \mathbf{x}_t||^2}{2l^2}\right) \tag{2}$$

where the parameter $l > 0$ is referred to as the *lengthscale* and controls the rate of decay of the similarity between the support vector $\mathbf{s}_j$ and the input sample $\mathbf{x}_t$. The Gaussian kernel reaches its maximum when the support vector $\mathbf{s}_j$ is equal to the input $\mathbf{x}_t$ and decays to zero when they are dissimilar. Furthermore, the Gaussian kernel is preferred since it is *universal*, meaning that the set of estimates produced by the Gaussian kernel is dense in the space of continuous functions; therefore an arbitrary degree of accuracy in regression is guaranteed by increasing the number of support vectors—see [19].

In our experiments, the input signal comprises both altitude $\mathbf{h}_t$ and (battery) voltage $\mathbf{v}_t$ measurements of an electric bicycle, that is, $\mathbf{x}_t = [\mathbf{h}_t^T, \mathbf{v}_t^T]^T$. In this case and in general for multi-input scenarios, a more expressive predictor can be designed by using one Gaussian kernel for each input and then multiply them together, the advantage of this formulation is that each input has a particular lenghtscale parameter. We have chosen this multiplicative kernel rather than a additive one to emphasise that two data samples, say $\mathbf{x}_t$ and $\mathbf{x}_{t'}$, are considered *similar* if both voltage and altitude components are *similar*.

Denoting by l_h and l_v the lengthscales for the altitude and voltage respectively, and the corresponding parts of the support vector $\mathbf{s}_j$ by $\mathbf{s}_j^{\mathbf{h}}$ and $\mathbf{s}_j^{\mathbf{v}}$, a two-input Gaussian kernel can be expressed as:

$$\begin{aligned} K_{l_h,l_v}\left(\begin{bmatrix}\mathbf{s}_j^{\mathbf{h}}\\ \mathbf{s}_j^{\mathbf{v}}\end{bmatrix}, \begin{bmatrix}\mathbf{h}_t\\ \mathbf{v}_t\end{bmatrix}\right) &= K_{l_h}(\mathbf{s}_j^{\mathbf{h}}, \mathbf{h}_t) K_{l_v}(\mathbf{s}_j^{\mathbf{v}}, \mathbf{v}_t) \\ &= \exp\left(-\frac{||\mathbf{s}_j^{\mathbf{h}} - \mathbf{h}_t||^2}{2l_h^2}\right) \exp\left(-\frac{||\mathbf{s}_j^{\mathbf{v}} - \mathbf{v}_t||^2}{2l_v^2}\right) \\ &= \exp\left(-\frac{||\mathbf{s}_j^{\mathbf{h}} - \mathbf{h}_t||^2}{2l_h^2} - \frac{||\mathbf{s}_j^{\mathbf{v}} - \mathbf{v}_t||^2}{2l_v^2}\right). \end{aligned} \tag{3}$$

The lengthscales l_h and l_v reveal how much the altitude and voltage explain the observations respectively, where an input that does not aid the prediction has a vanishing lengthscale—this kernel is therefore referred to as Automatic Relevance Determination Gaussian kernel [11]. A careful determination of the optimal kernel parameters is usually neglected within KAF due to the universal approximation property of the Gaussian kernel (i.e., it works even for a suboptimal choice of parameters), however, in our experiments we will determine the optimal lengthscales via minimisation of the relative mean square error on a training set.

[1] We have kept technicalities to a minimum and avoided feature-space references (reproducing kernel Hilbert space) to benefit conciseness and clarity. For a detailed presentation of KAFs, the reader is referred to [10].

2.2. Signal-dependent sparsification (dictionary construction)

The dictionary contains information about the region of the input space where the data have been *seen* and its size controls the trade-off between the algorithm's computational complexity and its predictive ability—the construction of the dictionary is referred to as *sparsification* [6]. Popular sparsification methods are the novelty criterion [9,13] and the coherence criterion [16], both of these strategies add the observed sequence $\mathbf{x}_t$ to the dictionary based on its proximity to the dictionary and whether it improves the estimation or not. Despite the adaptive nature of KAF, the sparsification criteria are always fixed which is unsuitable for nonstationary signals; e.g., for signals with a monotonically decreasing trend as the battery voltage in our experiments.

We propose a novel sparsification criterion that takes concepts from both methods mentioned above but with a further modification that allows it to adapt to changes in the magnitude of the target signal unlike the standard methods. We first take the proximity-to-dictionary rule from the coherence method measured by the kernel distance, see Eq. (4), then, we consider a prediction-oriented rule inspired on the novelty criterion, however, the novelty criterion considers the absolute error whereas we consider the relative error with respect to the current value of the signal, see Eq. (5), thus allowing larger absolute errors when the signal is large and vice versa. The proposed rules are:

$$\text{distance to dictionary} := \max_{\mathbf{s}_j \in \mathcal{D}} K(\mathbf{s}_j, \mathbf{x}_t) \geq \delta_{\text{dict}} \tag{4}$$

$$\text{prediction error} := ||x_{t+1} - \hat{x}_{t+1}|| \leq \delta_{\text{pred}} x_{t+1} \tag{5}$$

The pseudocode for the kernel least mean square with the proposed multi-input kernel and adaptive sparsification criterion is presented in Algorithm 1.

Algorithm 1 Proposed kernel least mean square for multiple inputs with adaptive sparsification rule.

```
# Initialisation
Construct input signal: x_t = [h_t, v_t]^T,  t = 1, ...
Dictionary D = {x_0}, number of delays d, weights w_1 = 0
Learning step: μ_w > 0
Sparsification parameters: δ_dict, δ_pred > 0
Kernel parameteres l_h, l_v > 0

for observation sequence x_t, x_{t-1}, ..., x_{t-(d-1)} do
    Denote regressor: x_t = [x_t, ..., x_{t-(d-1)}]^T
    Similarity distance to dictionary:
    d_dict = max_{s_j ∈ D} K_{l_h,l_v}(s_j, x_t)
    Prediction: x̂_{t+1} = Σ_{s_j ∈ D} w_j K_{l_h,l_v}(s_j, x_t)
    Error: e_pred = x_{t+1} - x̂_{t+1}
    if d_dict ≥ δ_dict and ||e_pred|| ≤ δ_pred x_{t+1} then
        Add new support vector: D ← D ∪ {x_t}
    end if
    w_j ← w_j + μ_w e_pred K_{l_h,l_v}(s_j, x_t)
end for
```

3. Case study: the electric bicycle

3.1. Experimental setting

We used an electric bicycle powered by a 250 W RMS electric motor fitted onto an urban-style frame,[2] the battery that energises the motor had the following specifications:

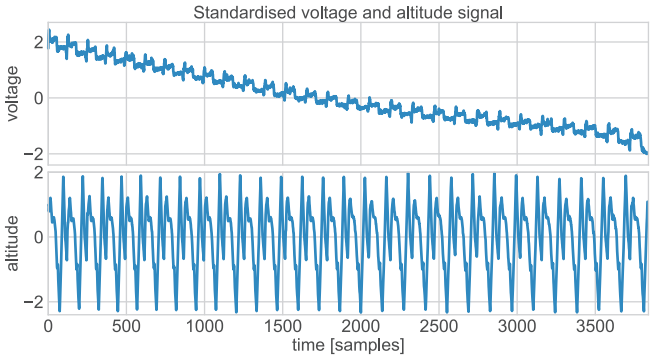

Fig. 1. Battery voltage and altitude signals recorded from electric bicycle (standardised). Notice that the periodic components of both signals are aligned but the voltage signal also features a decreasing trend and a waveform that changes in time.

- The battery package was comprised of 30 lithium-ion cells (4000 mAh, 3.7 V),
- the cells were connected in a 10×3 configuration to give a nominal output of 37 V and 12 A, thus having a total output power of 444 W, and
- the battery energy capacity Joules was 37 V $\times$ 12 A $\times$ 3600 s $=$1,598,400 J.

The voltage signal was acquired using the analog-to-digital (A/D) converter of an Arduino board at a sampling frequency of 0.5 Hz. The bicycle was further equipped with a (smartphone) GPS that provided geolocalisation measurements, these were validated through Strava's database in order to compute the altitude with a precision of up to 11 mm [20]. The motivation to include altitude measurements (an exogenous input to the system) instead of only considering inner variables of the battery, such as current and temperature, was to learn how the battery system reacts to the environment changes. Our hypothesis is that the voltage depends on the slope (differential altitude) rather than the altitude; we aim that this dependency is automatically discovered by the proposed algorithm by considering past observations. We denote the voltage signal coming from the battery (Arduino) by $v_{t\in\mathbb{N}}$ and the altitude signal by $h_{t\in\mathbb{N}}$, these were synchronised and combined into a vector to construct the observation signal $\{x_t = [v_t, h_t]^T,\ t \in \mathbb{N}\}$, also having a sampling rate of 0.5 Hz.

The experiment consisted in riding the bicycle at a constant (controlled) speed in a 440 m racetrack in O'Higgins Park, Santiago, Chile, starting from a fully-charged battery to a complete discharge. Notice that the altitude measurements were corrupted by terrain irregularities and sensor noise. The top speed was 16.6 km/h, the average speed 13.5 km/h, the travelled distance 29.4 km, and the altitude difference between the lowest and highest points of the route was 10 m as confirmed by the GPS and the geolocalisation database from Strava. The duration of the experiment was 2 h and 10 min (3900 samples) and the standardised voltage and altitude signals are shown in Fig. 1.

4. Data preprocessing and models considered

4.1. Assessing (non-)linearity of the voltage signal

Kernel adaptive filters (e.g., KLMS) are more computationally-expensive than linear ones (e.g., LMS). Therefore, we first motivate the use of KAFs by showing that if only a linear estimator is considered, the number of delayed samples required for a sound prediction becomes excessively large.

We studied the linear dependency between the altitude and voltage by computing the autocorrelation function (ACF) of the voltage signal, and the crosscorrelation function (CCF) between

[2] Equipment provided by the Chilean company EliBatt SPA.

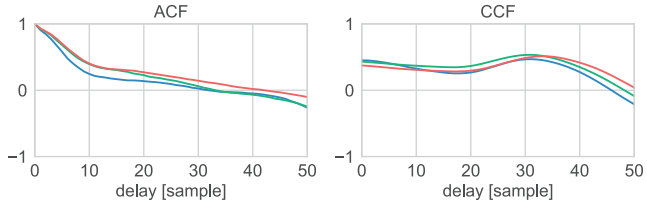

Fig. 2. Autocorrelation function (left, ACF) of the voltage signal, and cross-correlation function (right, CCF) between voltage and altitude signals. The three colours denote three non-overlapping thirds of the detrended data.

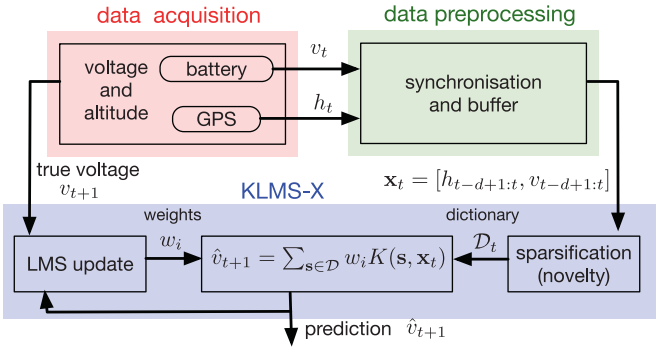

Fig. 3. Data acquisition from bicycle, lithium-ion battery and GPS (red), composition of the input data (green) and kernel adaptive filter (blue). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the voltage and altitude signals. As these signals are known to be nonstationary, the ACF and CCF were computed on three non-overlapping sections of the (detrended) experimental data and are shown in Fig. 2. Recall that the magnitude of the ACF (cf. CCF) is directly related to the magnitude of the coefficients of the linear estimator of the voltage given its own (cf. the altitude's) delayed samples. The fast decay of the ACF and the low values of the CCF functions do not allow us to confirm the existence of a linear relationship in the data either temporally or across channels. Furthermore, as it will be seen in Section 5, accounting only for linear relationships on the data is misleading and requires a large number of delays, where the proposed method will suggest the existence of short-term nonlinear relationship in the data.

4.2. KLMS-X: a voltage predictor that incorporates altitude observations

To assess whether the inclusion of altitude measurements improves the forecasting of the battery voltage, we considered the KLMS algorithm proposed in Section 2, which incorporates multi-input and adaptive-sparsification capabilities as seen in Algorithm 1, and implemented two variants: the first one incorporates altitude and voltage measurements (referred to as KLMS-X due to the exogenous input) and the second one is the standard autoregressive-only approach (referred to as KLMS). Note that for the single-input case, the multi-input kernel proposed in Eq. (3) collapses to the classic Gaussian kernel in Eq. (2). The diagram in Fig. 3 illustrates (i) the data acquisition module comprising the battery and the GPS on the bicycle in red; (ii) the data preprocessing module, that is, the synchronisation and composition of the input to the predictor in green; and (iii) the KLMS-X algorithm composed by the sparsification strategy that builds the dictionary, the least-mean-square (LMS) weight update and the computation of the prediction using a sum of weighted kernel evaluations in blue. Recall that the KLMS implementation does not incorporate the altitude information from the GPS module.

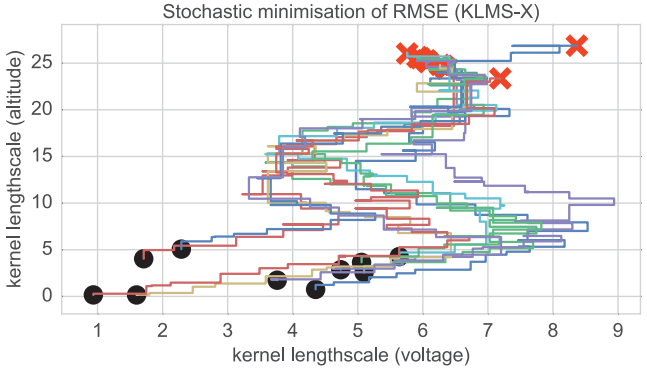

Fig. 4. Search of KLMS-X lengthscales using stochastic minimisation of RMSE (over the first 500 observations). The black dots are the initial values, the red crosses are the found minima and the trajectories are shown in colours. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4.3. Hyperparameter learning for KLMS and KLMS-X

Although both KLMS-X and KLMS algorithms are guaranteed to learn the hidden relationships on the data (recall the universality of the Gaussian kernel) for any choice of lengthscales, we found the optimal lengthscales in terms of prediction error by minimising the relative mean square error (RMSE) using the first $T = 500$ samples

$$\mathrm{RMSE} = 100\frac{\sum_{t=1}^{T}(v_t - \hat{v}_t)^2}{\sum_{t=1}^{T} v_t^2} \tag{6}$$

where v_t is the true voltage signal at time t, and $\hat{v}_t$ is the prediction computed at time $t-1$. Intuitively, the RMSE is a percentage ratio between the power of the prediction error and the power of the signal to predict.

The RMSE is highly nonlinear w.r.t. the lengthscale, not only because of the exponential function but also due to the sparsification criterion that links the number of support vectors to the optimal lengthscales. For this reason, the RMSE was minimised w.r.t. the lengthscale parameters by exhaustive search for KLMS (one parameter) and stochastic optimisation for the proposed KLMS-X (two parameters)—in the latter, the solution was found using a random walk which moves where accepted only when they lowered the RMSE. This initialisation of the adaptive filter resembles the approach in [2], where the hyperparameters of the algorithm are fitted by a *maximum-a-posteriori* rationale.

The search of the optimum lengthscales was performed for different delays d for both kernel algorithms over the first 500 observations. Fig. 4 shows the minimisation of RMSE reported by KLMS-X for $d = 30$ (this is the first time the ACF vanishes—see Fig. 2) as a function of its two lengthscale hyperparameters, where 10 different initial conditions in black dots arrived at the local minima shown in red crosses. The chosen hyperparameters were the average of these minima: $l_v = 6.42$, $l_h = 25.18$ for KLMS-X, and $l = 4.4$ for KLMS. For different delays, we scaled the lenghtscales linearly with the square root of the number of delays, this choice was based on the growth of the norm of the input and confirmed by an experimental exploratory analysis.

5. Experiments

We implemented KLMS-X for voltage prediction together with (i) a purely-autoregressive KLMS, (ii) least mean square filters using voltage observations only (LMS), and both voltage and altitude observations (LMS-X), (iii) a linear time-invariant order-one Gaussian state-space model (Kalman) and (iv) a basic estimator that

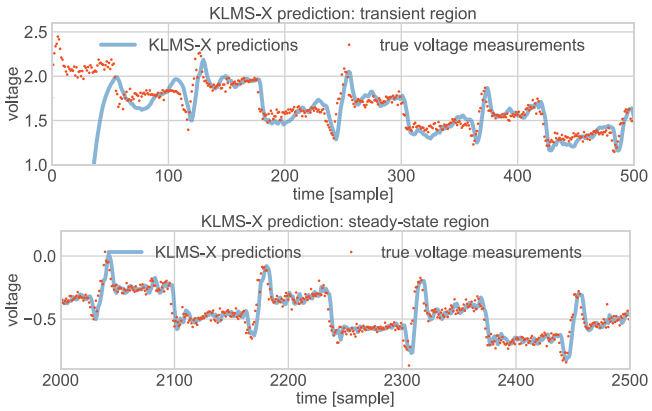

Fig. 5. Electric bicycle voltage recording (red dots) and KLMS-X predictions (blue line): transient region (top) and steady state (bottom). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

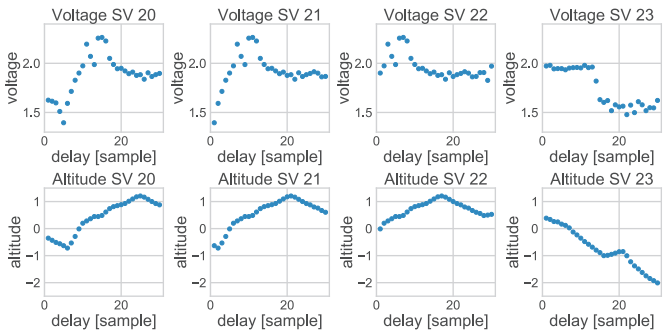

Fig. 6. Four support vectors (SVs) within KLMS-X. The SVs corresponding to the voltage (cf. altitude) signal are shown in the top (cf. bottom) row. Recall that the data is standardised.

Table 1
Performance and complexity of algorithms considered for one-step ahead prediction of voltage signal and 30 delays.

	KLMS-X	KLMS	LMS-X	LMS	Kalman
RMSE	0.329	0.512	0.479	0.35	0.284
Time	0.73 s	0.68 s	0.22 s	0.15 s	0.92 s
#SV	136	107	N/A	N/A	N/A

predicted the next voltage value using the last seen voltage observation (referred to a persistent estimator). All the parameters and hyperparameters were learnt using the first 500 observations, in particular, the parameter of the state-space model was (for all experiments) very close to unity and therefore its performance virtually equal to that of the persistent estimator.

We next show three different experiments: First, we implemented the aforementioned models for one-step-ahead prediction, where we focused on the 30-delay case in detail, and then analysed its performance as a function of the delay; the aim of this experiment was to show that the proposed approach is able to discover data structure in an automatic manner, i.e., without handcrafted terms. Second, we extended the KLMS and the proposed KLMS-X algorithm with a linear part in order to reduce computational complexity; the aim of this part was to show that the kernel algorithms admit prior signal knowledge (autoregressive behaviour). Third, we showed that the proposed method can also be used beyond the one-step-ahead scenario and show a 20-step-ahead prediction experiment.

5.1. One-step-ahead voltage prediction using kernel methods

We first implemented KLMS-X for one-step-ahead prediction of the voltage using 30 delayed samples, this corresponds to predicting 2 seconds ahead using the observations corresponding to the last minute—recall that the sampling frequency is 0.5 Hz. Fig. 5 shows the (standardised) measured voltage signal (red dots) and the prediction using KLMS-X (blue line) for the transient and steady-state regions, illustrating the improvement of KLMS-X as more data are seen in time. Observe the characteristic voltage-discharge curve and its trend, this is aligned with the intrinsic impedance (unknown for the KLMS-X algorithm), which determines the slope in the different operation zones of the battery [14]. Predictions are shown from sample 31, since 30 delayed samples are used.

Fig. 6 shows four consecutive support vectors (SV) encapsulating information about voltage and altitude trajectories from where KLMS-X has learnt; comparing these trajectories can reveal differences in, e.g., slope which affect the behaviour of the battery. One can intuitively understand the SVs and the weights w_j as a database, and then when new measurements are received they are compared to the elements in this database (SVs) in order to calculate the prediction based on similarity to the SVs and the learnt weights w_j. Furthermore, as the considered algorithm is nonparametric [18], the effective number of parameters increases with the number of observations. In this case, the support vectors and weights are the parameters of the predictor, which becomes more expressive as more data are seen.

The proposed KLMS-X algorithm was also compared to the KLMS that only considers past values of the voltage signal **and not the altitude**; the least mean square estimators considering past values of the voltage only (LMS) and both voltage and altitude (LMS-X); and the fixed-parameter Kalman filter. The rationale behind comparing KLMS-X against these linear predictors is to show that the kernel algorithms are effectively learning the non-linear time-varying relationships within the data and therefore their computational complexity is justified.

Table 1 shows the RMSE, the running time, and the dictionary size for all algorithms considered. Notice that the size of the dictionaries at the end of the experiment were 107 and 136 for KLMS and KLMS-X respectively, this is to be expected since the dimension of the input variable for KLMS-X is larger than that of KLMS (60 versus 30, for a delay of 30), and therefore extracting knowledge from larger datasets will result in increased computational complexity. However, the computational complexity is still in the same order of magnitude as revealed by running times of 0.68 s (KLMS) and 0.73 s (KLMS-X) for the duration of the experiment (3900 iterations), this is well suited for online implementation of the experiment considered, where data arrives at 0.5 Hz. Recall that both KLMS and KLMS-X are operating with their optimal hyperparameters as explained in Section 4.3.

Note that LMS performed better than arguably more expressive LMS-X, thus revealing that LMS-X is an incorrect model: it regards the relationship between the voltage and altitude as linear and learns a misleading instantaneous linear relationship. Additionally, the (fixed-parameter) Kalman filter exhibited the lowest RMSE, this suggests that the adaptive algorithms struggle to identify the simple autoregressive structure in the data; we incorporate the Kalman filter terms into the KLMS algorithms in the next experiment.

5.1.1. Comparison for different number of delays

To further validate the proposed KLMS-X as a means to extract expressive relationships between the altitude and voltage using fewer delays than KLMS or linear adaptive filters, we implemented all five algorithms for different number of delays from one

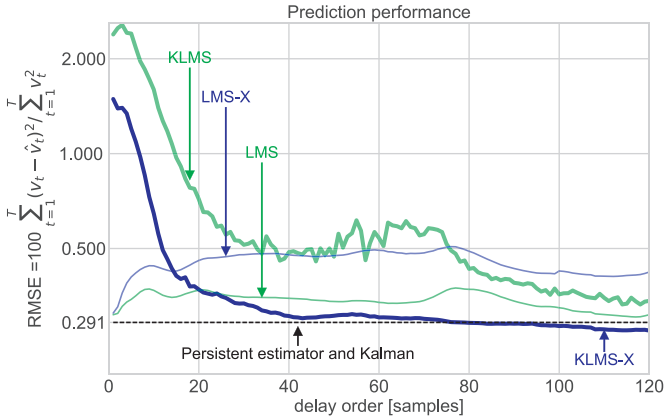

Fig. 7. Performance of kernel and linear algorithms considered for multiple-number-of-delays prediction of the voltage signal (relative mean square error).

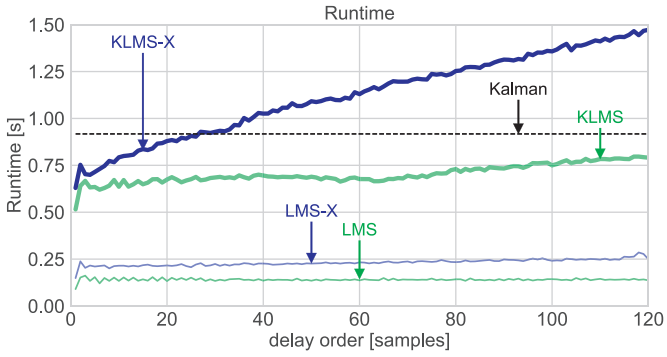

Fig. 8. Runtime (computational complexity) for voltage prediction as a function of the number of delays for all algorithms considered.

to 120 s and compared them in terms of their RMSEs (Fig. 7). The hyperparameters were also set as explained in Section 4.3.

Let us first notice that the performance of the linear adaptive filters improves very slowly, this was expected from the slow decay of the ACF and CCF shown in Fig. 2 and confirms that a large amount of data has to be processed to provide sound estimates if we only rely on linear relationships. On the contrary, observe how the RMSE related to the kernel filters decays sharply for the first few number of delays. One of our main contribution is highlighted here: note that it only required about 20 delayed samples for KLMS-X to surpass the other **adaptive** algorithms considered, thus confirming the ability of KLMS-X to find expressive nonlinear relationships between altitude and voltage using short regression horizons. This figure also reveals that, although in the considered delay horizon KLMS-X finally performs marginally better than the basic Kalman filter, it takes a large number of delays to do so—this will be addressed in Section 5.2.

Finally, Fig. 8 shows the running time for all algorithms considered. Despite the large computational complexity of the kernel algorithms, they process 3900 observations in less than 1 s for any number of delays, and are therefore appropriate for online operation, since the data arrives at 0.5 Hz. Recall that the Kalman filter requires a matrix inversion which is computationally expensive compared to LMS.

5.2. One-step-ahead voltage prediction using kernel adaptive filters with order-one linear component

Recall from Fig. 7 that despite the fact that the proposed KLMS-X is able to achieve better results than all algorithms considered, it takes about 90 delays to perform better than the basic persistent

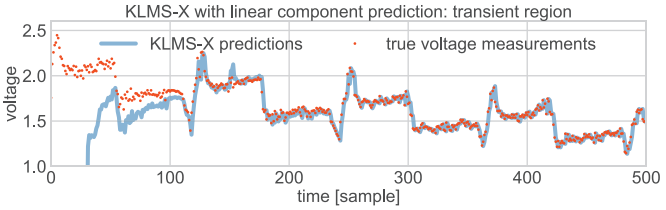

Fig. 9. Electric bicycle voltage recording (red dots) and KLMS-X predictions (blue line): transient region (top) and steady state (bottom). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

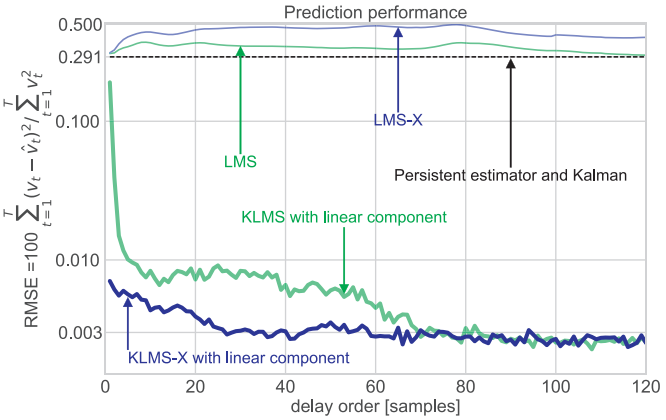

Fig. 10. Performance of kernel algorithms with linear component and linear algorithms considered for multiple-number-of-delays prediction of the voltage signal (relative mean square error).

estimator (or Kalman filter). For this reason, in this section we repeat the above experiment but now enhance both KLMS and the proposed KLMS-X with a linear function of the last seen sample, thus giving explicit form for the autoregressive component. That is, instead of the estimator in Eq. (1), the KAFs considered now have the form

$$\hat{X}_{t+1} = \alpha x_t + \sum_{\mathbf{s}_j \in \mathcal{D}_t} w_j K(\mathbf{s}_j, \mathbf{x}_t) \tag{7}$$

where the parameter α is to be updated in an LMS fashion as well, and the rest of the hyperparameters are learnt as explained in Section 4.3. The adaptation of KLMS-X with the linear component is shown in Fig. 9 for the 30-delay case, where it can be seen that the inclusion of the linear autoregressive part results in a shorter transient time as compared to the kernel-only case as shown in Fig. 5. We also run this experiment for all algorithms and different delays: Fig. 10 shows the RMSE for all algorithms considered where both KLMS and KLMS-X have the additional linear autoregressive terms, notice how now these kernel estimators are always better than the persistent estimator and thus achieve much better performance.

5.3. Long-term voltage prediction using KLMS-X

Finally, we implemented the proposed KLMS-X and compared it to the algorithms mentioned on a long-term prediction, a crucial application in battery health prognosis. We chose 20 steps ahead, 30 delays and implemented the kernel algorithms with the autoregressive linear component as explained in Section 5.2. The parameter of the linear term was trained offline, since for 20 steps the linear relationship decays in 80% and recovering it sequentially from data is challenging—see Fig. 2.

Fig. 11 shows the KLMS-X 20-step ahead prediction starting from sample 50, this is because 30 delays are used to forecast the

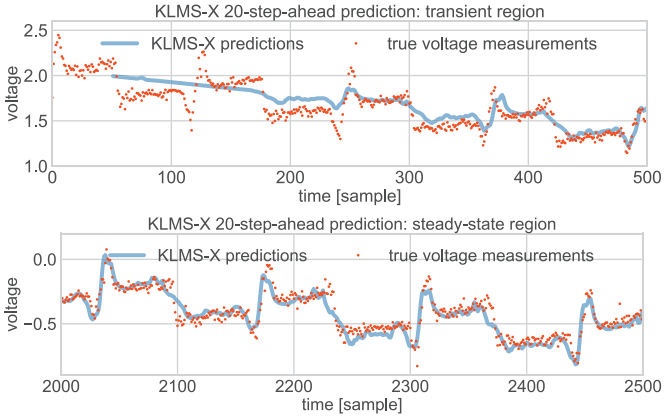

Fig. 11. 20-step-ahead prediction of the voltage signal using KLMS-X with linear autoregressive component.

Table 2
Performance of algorithms considered for 20-step ahead prediction of voltage signal and 30 delays.

	KLMS-X	KLMS	LMS-X	LMS	Kalman
RMSE	0.512	5.433	3.453	5.044	3.223

signal 20 steps into the future. Notice the large transient time compared to the one-step-ahead experiments in Figs. 7 and 10 due to the more challenging task of forecasting against short-term predictions. Table 2 reports the RMSE of all methods considered, where the proposed KLMS-X outperformed all other methods. Recall that the RMSE is a percentage between the power of the prediction error signal and the power of the signal to predict, where the large RMSE values in Table 2 are due to the fact that KLMS-X was the only algorithm able to predict the oscillations in the voltage, as these depended nonlinearly on the altitude.

6. Conclusions

Through a real-world case study, we have shown that it is possible to forecast an internal variable of an electric bicycle (voltage) using both past values of such variable and external measurements (altitude). The key challenge in the case considered was that the relationship between internal and external variables cannot be described by a known mathematical model, therefore, it was characterised by an expressive model purely learnt from data using kernel adaptive methods. This is then our main contribution: the use of a machine learning approach to discover structure from data in a sequential manner to then perform predictions that are fundamental for the application at hand.

In algorithmic terms, we have proposed a novel kernel adaptive filter that extends existing ones by catering for multiple inputs and has a signal-dependent rule for constructing the dictionary. The proposed KLMS-X (enhanced with a linear component) performed better than its purely-autoregressive counterpart (KLMS) and linear filters for all regression horizons in terms of prediction error and the amount of past samples required. Although the computational complexity of the proposed KLMS-X filter is slightly higher than the baselines considered, its complexity is more than acceptable and it can still be used in a real-time.

From an application point of view, our contribution is to show than kernel adaptive filters can be used to forecast the voltage of an electric bicycle using altitude measurements both for short- and long-term scenarios. This is a proof-of-concept that paves the way for implementation on other vehicles (e.g., cars or drones) through the consideration of more expressive environmental variables such as trajectory planning, road conditions or weight load. Furthermore, due to its bounded computational complexity, the proposed algorithm can be implemented online on the Arduino unit in order to develop accurate alerts about the autonomy and state of the vehicle in real time.

Acknowledgements

The authors acknowledge financial support from: CMM UMCNRS (F.T.), CONICYT-PAI #82140061 (F.T. and I.C.), CONICYT-FONDECYT #1170044 (M.O.), CONICYT-FONDECYT #1170854 (J.S.), CONICYT Basal project #FB0008 (M.O., I.C. and J.S.), and CONICYT-PIA ACT1405 (M.O. and J.S.).

References

[1] B. Arulampalam, Beyond the Kalman filter: Particle Filters for Tracking Applications, Artech House, 2004.
[2] I. Castro, C. Silva, F. Tobar, Initialising kernel adaptive filters via probabilistic Inference, in: Proc. of the IEEE International Conference on Digital Signal Processing (DSP), London, 2017, pp. 1–4.
[3] H. Drucker, C. Burges, L. Kaufman, A. Smola, V. Vapnik, Support vector regression machines, in: Proc. Advances in Neural Information Processing Systems 9, 1996, pp. 155–161.
[4] Y. Engel, S. Mannor, R. Meir, The kernel recursive least-squares algorithm, IEEE Trans. Signal Process. 52 (8) (2004) 2275–2285.
[5] S.S. Haykin, Adaptive Filter Theory, Pearson Education India, 2008.
[6] P. Honeine, Analyzing sparse dictionaries for online learning with kernels, IEEE Trans. Signal Process. 63 (23) (2015) 6343–6353.
[7] X. Hu, S.E. Li, Y. Yang, Advanced machine learning approach for lithium-ion battery state estimation in electric vehicles, IEEE Trans. Transport. Electrific. 2 (2) (2016) 140–149.
[8] W. Liu, I. Park, Y. Wang, J. Principe, Extended kernel recursive least squares algorithm, IEEE Trans. Signal Process. 57 (10) (2009) 3801–3814.
[9] W. Liu, P. Pokharel, J. Principe, The kernel least-mean-square algorithm, IEEE Trans. Signal Process. 56 (2) (2008) 543–554.
[10] W. Liu, J.C. Principe, S. Haykin, Kernel Adaptive Filtering: A Comprehensive Introduction, Wiley, 2010.
[11] R.M. Neal, Bayesian Learning for Neural Networks, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1996.
[12] M. Orchard, F. Tobar, G. Vachtsevanos, Outer feedback correction loops in particle filtering-based prognostic algorithms: statistical performance comparison, Stud. Inform. Control 18 (4) (2009) 295–304.
[13] J. Platt, A resource-allocating network for function interpolation, Neural Comput. 3 (2) (1991) 213–225.
[14] D. Pola, H. Navarrete, M. Orchard, R. Rabié, M. Cerda, B. Olivares, J. Silva, P. Espinoza, A. Pérez, Particle-filtering-based discharge time prognosis for lithium-ion batteries with a statistical characterization of use profiles, IEEE Trans. Reliab. 62 (2) (2015) 701–709.
[15] C.E. Rasmussen, C.K.I. Williams, Gaussian Processes for Machine Learning, The MIT Press, 2006.
[16] C. Richard, J. Bermudez, P. Honeine, Online prediction of time series data with kernels, IEEE Trans. Signal Process. 57 (3) (2009) 1058–1067.
[17] R.R. Richardson, M.A. Osborne, D.A. Howey, Gaussian process regression for forecasting battery state of health, J. Power Sources 357 (2017) 209–219.
[18] B. Scholkopf, A.J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, 2001.
[19] I. Steinwart, On the influence of the kernel on the consistency of support vector machines, J. Mach. Learn. Res. 2 (2001) 67–93.
[20] Strava, Strava support: elevation for your activity, 2012, (https://support.strava.com/hc/en-us/articles/%216919447-Elevation-for-Your-Activity).
[21] F. Tobar, P. Djurić, D. Mandic, Unsupervised state-space modelling using reproducing kernels, IEEE Trans. Signal Process. (2015) 5210–5221.
[22] F. Tobar, S.-Y. Kung, D. Mandic, Multikernel least mean square algorithm, IEEE Trans. Neural Netw. Learn. Syst. 25 (2) (2014) 265–277.
[23] N. Watrin, B. Blunier, A. Miraoui, Review of adaptive systems for lithium batteries state-of-charge and state-of-health estimation, in: IEEE Transportation Electrification Conference and Expo (ITEC), 2012, pp. 1–6.
[24] M. Yukawa, Multikernel adaptive filtering, IEEE Trans. Signal Process. 60 (9) (2012) 4672–4682.