Power Prediction for Electric Vehicles using Online Machine Learning

Stephan Rhode^{☆a,*}, Steven Van Vaerenbergh^{☆b}, Matthias Pfriem^a

^aInstitute of Vehicle System Technology, Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany ^bDepartment of Mathematics, Statistics and Computing, University of Cantabria, Santander 39005, Spain

Abstract

Accurate range prediction of an electric vehicle is an important open problem, affecting among others the adoption and market penetration of electric vehicles. Current range prediction systems suffer several practical limitations. Most critically these systems employ models in which all vehicle-specific parameters are required to be known. In this paper, we propose a methodology for predicting power and mission energy of electric vehicles that does not require knowledge of vehicle-specific parameters nor a drive-train model. The proposed method uses a data-driven approach grounded entirely on available vehicle sensor data. In particular, the predictive model is obtained by applying machine learning techniques, and in order to adapt to changing conditions in real-time, the specific class of kernel adaptive filtering algorithms is employed. Kernel adaptive filtering extends the theory of linear filters with concepts from kernel methods in order to construct nonlinear adaptive filtering algorithms that exhibit properties such as universal approximation capabilities and convexity in training, requiring only modest computational complexity. After providing an overview of the most relevant properties of kernel adaptive filters, we evaluate the proposed prediction methodology on data obtained in nine vehicle trial runs, comparing the performance of one linear adaptive filter, one online trained neural network and two state-of-the-art kernel adaptive filters.

Keywords: Electric vehicle, power prediction, kernel adaptive filters, fixed-budget

1. Introduction

Electric vehicles are regarded as a promising opportunity to reduce local carbon dioxide emissions in transport and to increase the overall energy efficiency, since the drive-train components of electric vehicles operate more efficiently than those of conventionally propelled vehicles. Despite the political will and purchase incentives in many countries, recent studies show that the registration of new electric vehicles remains at a negligible level. In most countries, registration is lower than 1 % of passenger cars Culver (2015). In particular, it is estimated as 0.7% in the United States, 0.4% in Canada, 1% in China and 0.6% in Japan International Energy Agency (2016). In the European Union, only 2 out of 26 member states exceed 2 % electric vehicles in passenger car sales Fergusson (2016).

Current batteries of electric vehicles show limited energy storage capacity with high costs, and the ratio of energy storage capacity to weight is poor. In addition, users fear limitation in their operation due to a comparably low maximum range per charge (\approx 110 km to 160 km per charge Hayes et al. (2011)). Therefore, price and range are seen as the main disadvantages of electric vehicles compared to conventionally propelled vehicles Carley et al. (2013); Adepetu & Keshav (2017)

In particular, drivers worry that they will not reach their desired destination, which is known as "range anxiety" Rauh et al. (2017). Nevertheless, the average range demand, which is reported as 30 km to 40 km per day for European drivers De la Fuente Layos (2007), is easily handled by current electric vehicles. Because of range anxiety, even experienced users do not exceed 75 % to 80 % of the electric vehicle's range Franke et al. (2012); Franke & Krems (2013). Furthermore, current systems for range prediction bear too much uncertainty Dütschke et al. (2012), causing the users' trust in electric vehicle range to be below the overall satisfaction with the vehicle. While there are several ongoing research efforts to reduce this uncertainty Tannahill et al. (2014); Faraj & Basir (2016); Sarrafan et al. (2017), car manufacturers accordingly tend to add substantial range buffers to treat range anxiety. However, these additional range buffers require extra costs, which, once again, lead to increased customer dissatisfaction. As analyzed in Wietschel et al. (2013), customers are only willing to pay little extra money for electric propulsion systems. Improving the precision of range prediction systems would result in a substantial increase of realworld range by using capacity that is already installed. Moreover, the increased weight and price of additional range buffers could be avoided. As summarized in Yu et al. (2012), improvements in range and range prediction systems are crucial for further adoption of electric vehicles and their market penetration.

The lack of precision in current range prediction systems can be mainly attributed to two factors, in particular, imprecise prediction of vehicle operation (predictive vehicle dynamics and usage of auxiliary power), and imprecise vehicle models

^AThe first and second authors contributed equally to this work. *Corresponding author

Email addresses: stephan.rhode@kit.edu (Stephan Rhode $^{\hat{\alpha}}$), steven.vanvaerenbergh@unican.es (Steven Van Vaerenbergh $^{\hat{\alpha}}$), matthias.pfriem@kit.edu (Matthias Pfriem)

(drive-train model, vehicle dynamics model). Recent advances in vehicle dynamics forecasting are discussed in Fünfgeld et al. (2017) and its references and fall out of scope for this study.

The present study contributes to improve the precision of vehicle dynamics models, because current vehicle dynamics models require knowledge of many vehicle-specific parameters, which limits their practical applicability. Current range prediction systems utilize white-box vehicle dynamics models Hayes et al. (2011); Wu et al. (2015); Grunditz & Thiringer (2016); Asamer et al. (2016), in which all the vehicle-specific parameters of the adopted model must be known. However, most of the involved parameters are time-varying, and their estimation has been the subject of a large body of literature on gray-box approaches Rhode et al. (2016); Vahidi et al. (2005); Fathy et al. (2008); McIntyre et al. (2009); De Bruyne et al. (2011); Rhode & Gauterin (2013), which estimate part of the model based on empirical data. A few preliminary studies on black-box modeling for range prediction have been carried out as well, for instance Chen et al. (2012b); Yu et al. (2012). Nevertheless, as will be discussed in the next section, these approaches are limited in scope and present several important issues.

This work introduces kernel adaptive filtering (KAF) algorithms Liu et al. (2011) as black-box modeling techniques for predicting power and mission energy of electric vehicles. KAFs are machine learning techniques that present several favorable properties for the problem at hand. In essence, KAFs are nonlinear extensions of linear adaptive filters, which makes them particularly suitable for online operation and tracking of timevarying models. The online learning capability marks a contrast with neural networks, whose training is generally formulated in a mini-batch fashion and unsuitable for most online applications Goodfellow et al. (2016). The capability of adapting to changing environments is a property that is currently being explored in several machine learning contexts, for instance in the nascent field of lifelong machine learning Zhiyuan & Bing (2018). Finally, KAFs exhibit universal approximation capabilities, and, in contrast to other black-box non-linear methods such as neural networks Goodfellow et al. (2016), KAFs are obtained by solving a convex optimization problem.

To the best of our knowledge, this work represents the first application of KAFs in electric vehicles, and for this reason, we include an extensive background on linear filters and the state-of-the-art in kernel adaptive filtering in the discussion.

The remainder of this paper is organized as follows: Section 2 provides an overview of related work and background on electric range prediction systems, followed by the motivation to employ kernel adaptive filters. Section 3 states the power prediction problem as a black-blox model and discusses the adopted assumptions. Section 4 reviews well-known linear filtering theory to highlight some similarities between linear filters and kernel adaptive filters, which are discussed in Section 5. A set of experiments with data from an electric vehicle are conducted and discussed in Section 6. Finally, Section 7 provides concluding remarks and a discussion of future research questions.

Throughout the discussion, the following notation will be used: Scalar variables are denoted as lowercase letters, x, and vectors as boldface lowercase letters, x, defined as column

vectors. Matrices are indicated by boldface uppercase letters, such as X.

2. Related work and background

A standard principle in design of range prediction systems is conservation of energy ($\sum E_i = 0$). In electric vehicles, this conservation can be described by an equilibrium between battery residual energy and mission energy. The battery residual energy is given by battery models and state-of-charge estimation approaches, which is an emerging research topic that falls outside the scope of this paper. The reader is kindly referred to the extensive overview about state-of-charge estimation in (Cuma & Koroglu, 2015, Table 1).

Accordingly, the following discussion of related work focuses on estimates of the mission energy of electric vehicles. In addition, we discuss the literature that deals with residual range estimation of electric vehicles, and we provide related work for vehicles with combustion engines.

2.1. Related work

Wu et al. Wu et al. (2015) used a well-known vehicle longitudinal dynamics model for the instantaneous power (p) and mission energy (E) of an electric vehicle. The power follows

$$p_t = F_t v_t, \tag{1}$$

where *F* denotes tractive force, *v* denotes velocity, and the index *t* denotes time. The integral of p_t yields the mission energy

$$E = \int_{0}^{t} p_t \mathrm{d}t.$$
 (2)

The instantaneous tractive force reads

$$F_t = m_V \dot{v_t} + f_r m_V g \cos \theta_t + m_V g \sin \theta_t + \frac{\rho_a}{2} c_w A_V v_t^2.$$
(3)

The terms on the right-hand side of (3) model the acceleration force, rolling resistance, climbing force, and aerodynamic resistance, respectively. Model (3) requires knowledge of several vehicle-specific parameters, in particular vehicle mass (m_V) , coefficient of rolling resistance (f_r) , drag coefficient (c_w) , and vehicle cross-sectional area (A_V) , as well as knowledge about the air density (ρ_a) and the instantaneous road angle (θ). A parameter sensitivity analysis was conducted in Asamer et al. (2016). In conclusion, the uncertainty in drive train efficiency, coefficient of rolling resistance, and auxiliary power demand should be reduced to improve mission energy modeling of electric vehicles. The study in Wu et al. (2015) treated model (3) as a white-box model, in which all vehicle-specific parameters were assumed to be known. However, these parameters are time-varying and usually only approximately known, which gave motivation for the vast literature on parameter estimators for model (3) Vahidi et al. (2005); Fathy et al. (2008); McIntyre et al. (2009); De Bruyne et al. (2011); Rhode & Gauterin (2013).

In addition to model (3), the authors of Wu et al. (2015) used a vehicle-specific drive-train model which considers losses of the electric motor and converter to compute the instantaneous power (p_t) of their electric vehicle. Hence, more vehicle-specific parameters were required, such as the electric motor resistance, flux, and transmission efficiency. The reported mean absolute error of the mission energy for 41 trips was 15.6 %.

A black-box model was applied in Chen et al. (2012b) to predict the residual range of a vehicle with combustion engine. A neural network (NN) with 21 neurons in a single hidden layer was trained using fuel volume, engine speed, velocity, vehicle mass, and road angle as the input data. The output data consisted of the residual range, and a large training data set was used, covering more than two months of vehicle data. The largest range prediction error was ≈ 4 km. Nevertheless, the NN-based technique proposed in Chen et al. (2012b) showed several issues, the major one being that the prediction quality on test data (unseen data for the NN) was not reported, although it is well known that NNs easily suffer from overfitting. Furthermore, the vehicle mass was treated as a known input even though this quantity is usually unknown. In addition to these issues, the training of NNs requires non-convex optimization, which is commonly conducted in batch processing of large data sets, and for this reason NNs are not recommended for online adaptation or online implementations on electronic control units.

Based on published vehicle parameters from manufacturers, Hayes et al. Hayes et al. (2011) derived drive-train models for two electric vehicles, specifically the Nissan Leaf and the Tesla Roadster. Some additional parameters such as the vehicle cross-sectional area or the coefficient of rolling resistance were estimated to model the range of the electric vehicles. However, the authors did not elaborate on how these parameters were estimated. Presumably, the additional parameters were empirically tuned by comparing the predicted range from the model with the given range from the manufacturers. The model of Hayes et al. (2011) was further validated with real-world data from trial runs, where the velocity and road angle from GPS served as model inputs. Therefore, the model in Hayes et al. (2011) is a white-box model with additional tuned parameters.

Yu et al. Yu et al. (2012) treat the problem of electric-vehicle range estimation by performing clustering of driving patterns, which belongs to black-box modeling. A trial run record was segmented into clusters with similar mission energy demand. The velocity, acceleration, road angle, and road curvature served as features (inputs to the black-box model) in the following machine learning procedure. The modeled mission energy was compared to measured mission energy from a trial run. The mean estimation error was 0.26 kW h.

Different Kalman filter algorithms were compared in Rhode et al. (2016) to estimate parameters of model (3) aiming to predict the vehicle tractive force of a vehicle with combustion engine. Note that the predicted tractive force could be easily converted into power through (1). Due to the involved parameter estimator, the approach of Rhode et al. (2016) belongs to graybox models. However, the method presented in Rhode et al. (2016) still requires a vehicle-specific drive-train model, in which knowledge about several vehicle-specific drive-train parameters is essential. Table 1: Overview of related work for electric vehicles and vehicles with combustion engine.

Reference	model type	predicted output
Asamer et al. (2016)	white box (3)	power, mission energy
Wu et al. (2015)	white box (3)	power, mission energy
Chen et al. (2012b)	black box (NN)	residual range
Hayes et al. (2011)	white box	residual range
Yu et al. (2012)	black box (clusters)	mission energy
Rhode et al. (2016)	gray box (3)	tractive force

to the used model type and the predicted output.

2.2. Motivation to apply kernel adaptive filters

A high-level comparison of kernel adaptive filters and other modeling techniques is provided in (Liu et al., 2011, p. 18). Kernel adaptive filters combine universal approximation capability with modest computational complexity. The concept of "universal approximation capability" indicates that any non-linear function f can be approximated with measured inputs (x) and measured outputs (y) such that $y \approx f(x)$ holds. Hence, kernel adaptive filters belong to the black-box model type. In contrast to white-box or gray-box approaches (see examples in Table 1), full physical understanding of the system is not required. However, some insight is still needed to select meaningful measured inputs and measured outputs, and to chose a suitable kernel. Preliminarily, a kernel is a function with specific properties. More details will be given in Section 5.1.

An important advantage of kernel adaptive filters is that they are obtained by solving a convex optimization problem, which is contrary to other non-linear machine learning methods such as NNs. A convex solution is always preferable, because only convex solutions allow for the design of algorithms that operate on fixed evaluation time, which is required for real-time applications. Also, kernel adaptive filters can easily be made robust against overfitting by incorporating assumptions about the covariance of the unknown non-linear function.

To be more specific on the instantaneous power prediction studied herein, KAFs overcome some obstacles which usually arise when modeling the power or mission energy of electric vehicles. Kernel adaptive filters ground entirely on available vehicle sensor data and the model adapts itself online during trial runs. Hence, the proposed method avoids the need for vehiclespecific parameters and drive-train models, which usually require high design effort due to costly drive-train test bench experiments. Compared to white- or gray-box methods, there is no need to gather information about vehicle-specific data such as tire radius, gear-ratio, efficiency maps, vehicle mass and so forth, which are usually required to design drive train models or to raise the tractive force model in (3). Kernel adaptive filters avoid the need for a drive train or tractive force model. Instead, the kernel adaptive filter allows to include an unknown drive train and tractive force model as part of the non linear mapping $y \approx f(\mathbf{x})$.

Table 1 categorizes all discussed related literature, according



Figure 1: Non-linear black box vehicle model, shown as block diagram. The aim is to estimate the unknown non-linear mapping $p_t \approx f(v_t, a_{xt})$, where f depicts the vehicle. *Note that the longitudinal acceleration (a_x) from the body-fixed acceleration sensor considers the road angle. See (6).

3. Problem statement

Figure 1 illustrates the adopted non-linear black-box vehicle model,

$$p_t \approx f(v_t, a_{\mathrm{x}t}) \tag{4}$$

which includes two measured inputs, velocity (v) and longitudinal acceleration (a_x) , and the measured output power (p), being the product of the measured electric current and voltage. Our objective is to approximate the unknown non-linear function $f(\cdot)$ of (4) given the measurements v_t, a_{xt} , and p_t .

The signals from the body-fixed longitudinal acceleration sensor are corrupted by acceleration due to gravity, cornering, and pitch. The vehicle acceleration, which is the derivative of the velocity, reads

$$\dot{v}_t = a_{\mathrm{x}t} \cos\beta + a_{\mathrm{y}t} \sin\beta - g\sin(\theta_t - \Theta_t) \tag{5}$$

(Kiencke & Nielsen, 2005, p. 353). Therefore, the measured longitudinal acceleration (a_x) is a function of the side slip angle (β) , lateral acceleration (a_y) , pitch angle (Θ) , and road angle (θ) . Consequently, model (4) considers longitudinal and lateral dynamics. Rearranging (5) gives

$$a_{\mathrm{x}t} = \frac{1}{\cos\beta} (\dot{v}_t - a_{\mathrm{y}_t} \sin\beta + g\sin(\theta_t - \Theta_t))$$

which can be simplified by assuming β and Θ being small to

$$a_{\mathrm{x}t} = \dot{v}_t + g\sin\theta_t. \tag{6}$$

The measured velocity is given by a sophisticated velocity observer of the electronic stability system. See (Kiencke & Nielsen, 2005, pp. 351–363) for a detailed review of vehicle velocity estimation with Kalman filter or fuzzy logic.

In addition to the vehicle dynamics, model (4) also considers losses that arise inside the drive train. These losses are caused by acceleration of rotational parts (shafts, wheels), friction, and losses inside the electric motor and gear-box. Conversely to conventional approaches, model (4) considers drive-train losses but does not require efficiency maps of the electric motor and the gear-box.

Model (4) omits additional inputs for accessory power of lightning, air conditioning, or heating and assumes that these accessory power fractions were kept constant or change slowly during operation. Accordingly, the non-linear function $f(\cdot)$ in (4)

accounts additional losses of accessory power. Estimation of the non-linear function $f(\cdot)$ in (4) is therefore a challenging problem, taking into account that no prior knowledge about vehicle specific parameters such as vehicle mass, drag coefficient, tire radius, gear ratio, and efficiency maps, is given.

In order to predict the instantaneous power, future information of velocity and longitudinal acceleration are required to feed the right hand side of model (4). While the question of how to obtain this future information falls outside the scope of this paper, vehicle dynamics forecasting methods were extensively discussed in Fünfgeld et al. (2017). The main approaches are based on time series models, road data, or stochastic models. A sensible approach based on road data is described by Ziegler et al. (2014), in which recent (semi) autonomous vehicles use map data (road angle, road curvature) and plan a velocity trajectory for a forthcoming routst e. The planned velocity trajectory is used together with map data and relation (5) (or its simplified version (6)) to feed the predictive inputs of model (4).

4. Linear regression and adaptive filtering

We now provide a short overview of some basic concepts of linear regression and adaptive filtering theory, before introducing the conceptual design of kernel adaptive filters in the next section.

Given a set of *m* measured inputs $x_t \in \mathbb{R}^{n \times 1}$ and *m* corresponding measured outputs y_t , with $t = 1, \ldots, m$, linear regression assumes a linear data model

$$y_t = \boldsymbol{w}^\top \boldsymbol{x}_t + \boldsymbol{e}_t, \tag{7}$$

in which $w \in \mathbb{R}^{n \times 1}$ contains the regression coefficients and e_t is the residual for the *t*-th data pair $\{x_t, y_t\}$.

Linear regression is concerned with estimating the coefficients w that minimize the residual errors e_t of (7) according to a suitable cost function. The most prominent approach consists in minimizing the least-squares cost, which calculates the sum of the squared errors

$$\min_{\mathbf{w}} J = \frac{1}{2} \sum_{t=1}^{m} |y_t - \mathbf{w}^{\top} \mathbf{x}_t|^2.$$
(8)

The gradient of J with respect to w is given by

$$\mathcal{J} = -\sum_{t=1}^{m} (y_t - \boldsymbol{w}^\top \boldsymbol{x}_t) \boldsymbol{x}_t = -\sum_{t=1}^{m} e_t \boldsymbol{x}_t.$$
(9)

4.1. Least mean squares

Least mean squares (LMS) is a low-complexity algorithm that computes the weights (w) that minimize the least squares cost function (8) (Sayed, 2003, p. 212). Instead of making any statistical assumptions about the data, LMS uses an instantaneous approximation of the gradient (9) at instant t,

$$\mathcal{J}_t \approx -(y_t - \boldsymbol{w}^\top \boldsymbol{x}_t) \boldsymbol{x}_t = -e_t \boldsymbol{x}_t.$$
(10)

Each time a new data pair $\{x_t, y_t\}$ is measured, the LMS algorithm performs the following operations in order to update its solution \widehat{w} :

$$e_t = y_t - \widehat{\boldsymbol{w}}_{t-1}^\top \boldsymbol{x}_t, \tag{11a}$$

$$\widehat{\boldsymbol{w}}_t = \widehat{\boldsymbol{w}}_{t-1} + \eta \boldsymbol{e}_t \boldsymbol{x}_t, \tag{11b}$$

where η represents a learning rate.

Since its invention more than half a century ago, the LMS algorithm has been the workhorse of adaptive filtering, mainly due to the following interesting properties:

- Complexity: LMS is a simple algorithm with computational complexity of O(n) per iteration.
- Adaptive filtering: By adapting its estimate of the optimal weights w at each time step, LMS belongs to the family of adaptive filtering algorithms. These algorithms are especially suitable for real-time scenarios such as the current one, in which the solution must be improved as new data becomes available.
- Tracking: Though LMS is designed to solve (7), which is a fixed model, its instantaneous gradient update allows it to track changes that may occur in the this model.

Under the stationarity assumption, the LMS algorithm converges to the Wiener solution in mean (Sayed, 2003, Chapter 2), but the weight vector \hat{w} shows a variance that converges to a value that is a function of η . Therefore, low variances are only achieved at low adaptation speed. A more sophisticated approach with faster convergence is found in the the Recursive Least-Squares (RLS) algorithm.

4.2. Recursive least squares

The RLS algorithm was first introduced by Plackett in 1950 Plackett (1950). In a stationary scenario, it converges to the Wiener solution in mean and variance, improving also the slow rate of adaptation of the LMS algorithm. Nevertheless, this gain in convergence speed comes at the price of a higher complexity, as we will see below.

A closed-form solution for the least-squares problem (8) can be obtained by setting its gradient (9) to zero, yielding

$$\widehat{\boldsymbol{w}} = \left(\sum_{t=1}^{m} \boldsymbol{x}_t \boldsymbol{x}_t^{\mathsf{T}}\right)^{-1} \sum_{t=1}^{m} \boldsymbol{x}_t y_t, \qquad (12)$$

given that the matrix $\sum_{t=1}^{m} x_t x_t^{\top}$ is nonsingular. The recursive least squares (RLS) algorithm is a well-known recursive estimator for (12) and gives (contrary to LMS) the optimal solution for wunder the assumption that e_t is a sequence of zero-mean white Gaussian noise. In each iteration, the RLS algorithm performs the following operations:

$$e_t = y_t - \widehat{\boldsymbol{w}}_{t-1}^\top \boldsymbol{x}_t, \tag{13a}$$

$$\boldsymbol{g} = \boldsymbol{P}_{t-1}\boldsymbol{x}_t \left(1 + \boldsymbol{x}_t^{\top} \boldsymbol{P}_{t-1} \boldsymbol{x}_t \right)^{-1}, \quad (13b)$$

$$\widehat{\boldsymbol{w}}_t = \widehat{\boldsymbol{w}}_{t-1} + \boldsymbol{g}\boldsymbol{e}, \qquad (13c)$$
$$\boldsymbol{P}_t = (\boldsymbol{I} - \boldsymbol{g}\boldsymbol{x}_t^\top)\boldsymbol{P}_{t-1}, \qquad (13d)$$

$$\boldsymbol{P}_t = \left(\boldsymbol{I} - \boldsymbol{g} \boldsymbol{x}_t^{\top}\right) \boldsymbol{P}_{t-1}, \qquad (13d)$$

where $\boldsymbol{P} \in \mathbb{R}^{n \times n}$ is the covariance matrix, $\boldsymbol{g} \in \mathbb{R}^{n \times 1}$ is the gain, and *I* is the identity matrix, (Sayed, 2003, p. 735).

RLS is an adaptive filtering algorithm, though in its standard form it is not capable of performing tracking. Several related

algorithms have been proposed to deal with this issue, most notably RLS with exponential forgetting factor (Ljung, 1999, p. 365) and extended RLS (Sayed, 2003, Section 12.B), which is related to the Kalman filter Kalman (1960). In terms of performance, RLS converges much faster than LMS, though this gain comes at the expense of a higher computational complexity $O(n^2)$ per iteration.

5. Kernel adaptive filters

A standard procedure to extend the scope of linear filters to non-linear processing consists in mapping the input data $x \in X$ to a high-dimensional *feature space* by means of a non-linear transformation $\Phi(\cdot)$ and then applying the linear filtering algorithm in that space Broomhead & Lowe (1988). By operating in a high-dimensional space, more degrees of freedom are available to formulate a solution for the filtering or regression problem. Nevertheless, the additional dimensions also increase the computational load of the algorithm, which may become excessive.

Kernel methods provide an efficient solution to this problem by avoiding the explicit mapping of the data (Schölkopf & Smola, 2002, p. 15). Kernel methods exploit a property from the theory of reproducing kernel Hilbert spaces (RKHS) that allows to implicitly operate in the high-dimensional feature space by simply replacing inner products by Mercer kernels in the original problem formulation.

5.1. Background

A Mercer kernel is any continuous, symmetric and positive definite function $\kappa : X \times X \to \mathcal{R}$ Aronszajn (1950). The most widely used kernel is the Gaussian or "radial basis function" kernel,

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2\sigma^2}\right),\tag{14}$$

whose parameter is the kernel width (σ). According to Mercer's theorem, any Mercer kernel $\kappa(\mathbf{x}, \mathbf{x}')$ induces a mapping $\Phi(\cdot)$ from the input space to a high-dimensional feature space (\mathbb{F}) (which is an inner product space) such that the following relationship holds Vapnik (1995)

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \Phi(\boldsymbol{x})^{\top} \Phi(\boldsymbol{x}'). \tag{15}$$

In case of the Gaussian kernel, the associated feature space \mathbb{F} is infinitely-dimensional. The property (15), commonly known as the kernel trick, is the building block of kernel methods. It implies that by replacing the inner products by kernels in a linear algorithm, a new algorithm is obtained that is equivalent to performing the original algorithm in the feature space, without the need to perform any explicit calculations in this high-dimensional space.

The output of a linear filtering algorithm in feature space can be written as

$$f(\boldsymbol{x}) = \widetilde{\boldsymbol{w}}^{\top} \boldsymbol{\Phi}(\boldsymbol{x}), \tag{16}$$

where \tilde{w} contains the coefficients of the filter. In order to avoid explicitly calculating the involved high-dimensional vectors, one



Figure 2: A kernel adaptive filter applied to nonlinear system identification.

can rely on the Representer theorem Schölkopf et al. (2001), which guarantees that the solution in the feature space, \tilde{w} , can be expressed as a linear combination of the *m* transformed data points,

$$\widetilde{\boldsymbol{w}} = \sum_{t=1}^m \alpha_t \Phi(\boldsymbol{x}_t)$$

This expression allows to rewrite (16) as

$$f(\boldsymbol{x}) = \sum_{t=1}^{m} \alpha_t \kappa(\boldsymbol{x}_t, \boldsymbol{x}) = \boldsymbol{\alpha}^\top \boldsymbol{k}, \qquad (17)$$

in which α and k are *m*-dimensional vectors that hold the elements α_t and $\kappa(\mathbf{x}_t, \mathbf{x})$, respectively. Equation (17) is called the *kernel expansion*, and the coefficients α_t are referred to as *kernel expansion coefficients*. A diagram representing a generic kernel adaptive filter is represented in Fig. 2.

5.2. Kernel least mean squares

The update equations (11) of the LMS algorithm can be ported directly to the kernel feature space, where they read

$$e_t = y_t - \widetilde{\boldsymbol{w}}_{t-1}^{\top} \Phi(\boldsymbol{x}_t), \qquad (18a)$$

$$\widetilde{w}_t = \widetilde{w}_{t-1} + \eta e_t \Phi(\mathbf{x}_t). \tag{18b}$$

In order to derive a set of update equations for the coefficients α , several alternatives have been proposed. The interested reader is referred to Van Vaerenbergh & Santamaría (2014) for an in-depth discussion. Many of these methods are based on the kernel least mean squares (KLMS) algorithm (Liu et al., 2011, p. 34), which performs the following updates:

$$e_t = y_t - \boldsymbol{\alpha}_{t-1}^\top \boldsymbol{k}_t, \qquad (19a)$$

$$\alpha_t = \begin{bmatrix} \alpha_{t-1} \\ \eta e_t \end{bmatrix}, \tag{19b}$$

in which k_t is a t - 1-dimensional vector whose *i*-th element is $\kappa(\mathbf{x}_i, \mathbf{x}_t)$. In order to calculate k_t , all previous data \mathbf{x}_i , $i = 1, \ldots, t - 1$ need to be stored. The update equations (19) represent a nonlinear version of LMS that corresponds to linear filtering in the high-dimensional feature space induced by the Mercer kernel.

Several interesting observations can be made at this point. First, the update (19b) implies that the vector of filter coefficients α_t grows unboundedly over time, and, as mentioned, all data x_i needs to be stored. Note that this growth stems from the fact that the kernel expansion (17) relies explicitly on all observed data x_t . In Section 5.4 we will outline some approaches that allow to deal with this issue. A second noteworthy observation is that the computational complexity of KLMS for the *m*-th iteration is O(m), which is linear in terms of the number of "bases" stored.

5.3. Kernel recursive least squares

In a similar fashion, a nonlinear "kernelized" version of the RLS algorithm can be obtained. The least-squares cost function (8) is transformed into feature space and extended with a regularization term to avoid overfitting¹,

$$\min_{\widetilde{\boldsymbol{w}}} J = \frac{1}{2} \sum_{t=1}^{m} |y_t - \widetilde{\boldsymbol{w}}^{\top} \Phi(\boldsymbol{x}_t)|^2 + \frac{\lambda}{2} \|\widetilde{\boldsymbol{w}}\|^2,$$

where λ is a regularization constant. By introducing the kernel matrix K with elements $K_{ij} = \Phi(x_i)^{\top} \Phi(x_j) = \kappa(x_i, x_j)$, a matrix-based expression of the cost function is obtained as

$$\min_{\alpha} J = \frac{1}{2} \| \mathbf{y} - \mathbf{K} \boldsymbol{\alpha} \|^2 + \frac{\lambda}{2} \boldsymbol{\alpha}^{\mathsf{T}} \mathbf{K} \boldsymbol{\alpha}.$$
(20)

Equation (20) represents the regularized kernel least-squares problem, also known as *kernel ridge regression (KRR)* Shawe-Taylor & Cristianini (2004). The closed-form solution to this problem is found by setting its gradient w.r.t. α to zero and by solving for α ,

$$\boldsymbol{\alpha} = (\boldsymbol{K} + \lambda \boldsymbol{I})^{-1} \, \boldsymbol{y}. \tag{21}$$

The kernel recursive least squares (KRLS) algorithm Engel et al. (2004a) is concerned with retrieving the solution (21) in a recursive manner: Given the solution α_{t-1} for a set of t - 1 data pairs { x_i , y_i }, KRLS updates this solution in order to incorporate a new data pair { x_t , y_t }. The update equations, which are derived for instance in (Liu et al., 2011, p. 101), read

$$\boldsymbol{k}_t = [\kappa(\boldsymbol{x}_1, \boldsymbol{x}_t), \dots, \kappa(\boldsymbol{x}_{t-1}, \boldsymbol{x}_t)]^\top, \quad (22a)$$

$$e_t = y_t - \alpha_{t-1}^\top k_t, \qquad (22b)$$

$$q_t = K_{t-1}^{-1} k_t,$$
 (22c)

$$r_t = \kappa(\boldsymbol{x}_t, \boldsymbol{x}_t) + \lambda - \boldsymbol{q}_t^\top \boldsymbol{k}_t, \qquad (22d)$$

$$\boldsymbol{K}_{t}^{-1} = \begin{bmatrix} \boldsymbol{K}_{t-1}^{-1} & \boldsymbol{0} \\ \boldsymbol{0}^{\top} & \boldsymbol{0} \end{bmatrix} + \frac{1}{r_{t}} \begin{bmatrix} \boldsymbol{q}_{t} \\ -1 \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_{t} \\ -1 \end{bmatrix}^{\top}, \qquad (22e)$$

$$\boldsymbol{\alpha}_{t} = \begin{bmatrix} \boldsymbol{\alpha}_{t-1} \\ 0 \end{bmatrix} - \frac{\boldsymbol{e}_{t}}{\boldsymbol{r}_{t}} \begin{bmatrix} \boldsymbol{q}_{t} \\ -1 \end{bmatrix}.$$
(22f)

It is immediately observed that the matrices and vectors involved in the KRLS updates represent growing structures, as was the case for KLMS. Furthermore, the computational complexity for the *m*-th iteration of KRLS is even higher, $O(m^2)$, due to the growing matrix K_t^{-1} in (22e). While the algorithm described in (22) may be used to solve modestly-sized nonlinear regression problems in a recursive manner, it is not suitable for continued operating on long data sequences. In the sequel we will deal with this issue.

¹Interestingly, overfitting is typically not an issue in KLMS algorithms.

5.4. Practical machine learning with kernel adaptive filters

The KLMS and KRLS algorithms described in the previous discussion are powerful non-linear adaptive filtering techniques. However, in practice they present several issues, such as unbounded growth, that prevent them from being used. After their original formulations were proposed, several more sophisticated algorithms have been designed that extend the basic KLMS and KRLS algorithms with mechanisms to overcome these different practical issues. Here, we will discuss these issues and their solutions.

5.4.1. Fixing the budget

Adaptive filters should be capable of operating during extended periods of time, processing large amounts of data. Kernel methods rely on the functional representation (17), which grows as the amount of processed observations *m* increases. A naïve implementation of an online kernel adaptive filter will therefore require growing computational resources during operation, leading to performance issues once either the available memory is insufficient to store the training data or once the computations for one update take more time than the interval between incoming data Van Vaerenbergh & Santamaría (2014). The observations stored in memory are referred to as the *dictionary*, and in this case the dictionary will be growing without limit during online operation, as illustrated in Fig. 3 (left).

The standard approach to overcome this issue is to construct an *approximate* kernel expansion that only includes the most relevant terms through a process called *sparsification*. This process leads to smaller dictionary sizes, and, accordingly, a lower computational complexity. Practical kernel adaptive filters start with an empty dictionary and gradually add more observations. One way of limiting the dictionary size, and thus the algorithm's complexity, is by restricting its *growth*: an observation is only added to the dictionary if it fulfills a certain sparsification criterion. The online sparsification procedure process is illustrated in Fig. 3 (center), and it can be found in the KLMS algorithms Richard et al. (2009); Chen et al. (2012a) and the original KRLS algorithm Engel et al. (2004a).

In many occasions it is also necessary to remove older observations from the dictionary, for instance when the observed model is changing over time. In this case a pruning action is required. Furthermore, by carefully controlling the growth and the pruning it becomes possible to design kernel adaptive filters whose dictionary size remains fixed, called "fixed-budget" algorithms. These are of specific interest in practical scenarios, as they guarantee that their required resources will not surpass a fixed limit. An example of a fixed-budget dictionary construction is provided in the last plot of Fig. 3. Several fixed-budget KRLS algorithms have been proposed, most notably Van Vaerenbergh et al. (2010, 2012a). Optimal pruning is a more complicate manner for KLMS algorithms, since these algorithms' low complexity makes it harder to determine which observations are least relevant. Nevertheless, a fixed-budget KLMS algorithm was recently proposed in Zhao et al. (2013).

5.4.2. Tracking capability

LMS and KLMS algorithms provide tracking capability outof-the-box. This is not the case, however, for neither the standard RLS algorithm nor the original KRLS algorithm Engel et al. (2004b). As a matter of fact, KRLS is based on the assumption of a stationary model (7).

In order to endow a KRLS algorithm with tracking capability, two successful approaches have appeared in the literature. The first approach uses a sliding-window that prunes all data older than a certain threshold age Van Vaerenbergh et al. (2006). The resulting algorithm is called sliding-window KRLS, and it consists of a small set of update equations that are easy to implement. Recently, a more sophisticated algorithm was proposed, called kernel recursive least-squares tracker (KRLS-T) Van Vaerenbergh et al. (2012a). This algorithm achieves tracking capability by modifying the underlying data model into a time-varying probabilistic model using Gaussian process theory.

5.4.3. Hyperparameter selection for kernel adaptive filters

Kernel adaptive filtering algorithms require to determine several parameters, such as Gaussian kernel width, amount of regularization, etc., which are commonly referred to as "hyperparameters" to distinguish them from other parameters such as expansion coefficients.

The optimal values for the hyperparameters are typically determined before running the online experiment, on a separate validation data set. The standard way to determine these optimal values is by performing a grid search using cross-validation. Obviously, this approach is only valid when only a few values are considered for each hyperparameter, as the amount of scenarios to consider grows exponentially with the number of values to consider. A more sensible approach is described in the literature on Gaussian processes (GP) Rasmussen & Williams (2006). In particular, Gaussian processes allow to obtain the optimal hyperparameters by maximizing the marginal likelihood of the hyperparameters given the training data data. Thanks to the equivalence between KRR and GP regression, the values obtained through this approach are valid for KRR and KAF algorithms as well. More details can be found in (Rasmussen & Williams, 2006, Chapter 5) and Van Vaerenbergh et al. (2012b).

5.4.4. Kernel adaptive filters used in the experiments

A decade of research on kernel adaptive filtering has produced several state-of-the-art algorithms. For a detailed overview, the interested reader is kindly referred to Van Vaerenbergh & Santamaría (2014). The experiments of the next section will focus on the two algorithms that show the best empirical performance, one based on KLMS and the other one based on KRLS.

For the family of KLMS algorithms, the best performance is obtained by the fixed-budget quantized kernel least mean squares (QKLMS-FB) algorithm Zhao et al. (2013). QKLMS-FB uses a quantization mechanism based on the *coherence criterion* for deciding which data to add to the dictionary, and a measure of data *significance* to decide which data to prune.

For the family of KRLS algorithms, the best performance is obtained by the KRLS-T algorithm Van Vaerenbergh et al.



Figure 3: Different approaches to online dictionary construction. Each horizontal line marks the presence of a center in the dictionary. Left: The evergrowing dictionary construction, in which the dictionary contains *n* elements in iteration *n*; Center: Online sparsification, which slows down the dictionary growth; Right: Fixed-budget approach, in which the pruning criterion discards one element per iteration, displayed with dictionary size 10.

Table 2: Computational complexity of algorithms. *n* denotes the number of inputs, *m* the samples (which grows over time, $m \rightarrow \infty$), and \overline{m} the user-defined fixed dictionary size.

Algorithm	$O(\cdot)$
LMS	п
RLS	n^2
KLMS	т
KRLS	m^2
QKLMS-FB	\overline{m}
KRLS-T	\overline{m}^2

(2012a), which was devised by constructing standard KRLS theory from a Bayesian probabilistic point of view. KRLS-T is a sequential Gaussian-process based algorithm that allows for variations in the underlying data model. In addition to predictions, it also provides confidence intervals.

Both selected algorithms combine all of the discussed favorable properties, i.e. they are nonlinear adaptive filtering algorithms, capable of tracking time-varying models, and they operate on a fixed budget per iteration. Table 2 summarizes the computational complexity of these algorithms w.r.t. their linear counterparts and the theoretical, evergrowing algorithms KLMS and KRLS.

6. Experiments

Nine trial runs on public roads were conducted with a prototype electric vehicle, and signals of the velocity, longitudinal acceleration, electric current and voltage, and brake pressure were recorded. The vehicle signals were read at 100 Hz on the rapid prototype control unit ETAS ES-910. All signals were smoothed by a Savitzky-Golay filter of third order with a span of 50 to remove noise. In practice, there is no need to predict the power of an electric vehicle at 100 Hz. Accordingly, all signals were down-sampled to 2 Hz prior model estimation. Additionally, driving states where the vehicle brakes through mechanical disc brakes were removed from the training and test data, because model (4) does not consider brake pressure or other inputs that depict the mechanical breaking power. The final number of samples in each trial run is shown in Table 3.

Table 3: Size of the data sets corresponding to the trial runs, after resampling at 2 Hz.

N⁰	т
1	2,204
2	1,916
3	1,166
4	1,325
5	1,559
6	1,053
7	2,673
8	2,886
9	2,417

In the first experiment, each record was divided into a training and a test data set. The latter, which was used for measuring the performance of the compared algorithms, consisted of the last (τ) samples of the trial run. Here, (τ) was fixed at 200 samples (at 2 Hz frequency). Accordingly, the size of the training data varies over the trial runs. Figure 4 shows three vehicle states of the training data from trial run No 1. The electric power (the measured output of model (4)) is shown as the normalized quantity $(p/\max p)$ in Figure 4a. Figure 4b and Figure 4c show the longitudinal acceleration and velocity, respectively. Both signals build the measured inputs in (4).

6.1. Adaptive filter setup

Two fixed-budget kernel adaptive filters were used to estimate the non-linear vehicle model (4). Specifically, we selected KRLS-T Van Vaerenbergh et al. (2012a) and QKLMS-FB Zhao et al. (2013) given their state-of-the-art performance in online and adaptive learning tasks. Both algorithms are implemented in the Kernel adaptive Filtering Toolbox Van Vaerenbergh & Santamaría (2013), which is used to conduct the experiments in this section and which is maintained by one of the authors of this paper. Additionally, two benchmark algorithms were applied to estimate model (4): Firstly, the linear RLS filter from (13) was included to evaluate the benefit of kernel adaptive filters over linear filters. Secondly, a neural network that is trained online was included, motivated by the wide range of machine learning applications in which neural networks currently achieve excellent results. The four compared algorithms (KRLS-T, QKLMS-FB,



Figure 4: Characteristic vehicle states from trial run \mathbb{N} 1. All signals were smoothed and resampled at 2 Hz.

RLS, and NN) were trained online with the training data of trial runs $N \ge 1-9$. The respective kernel expansion coefficients (α), filter weights (w), and NN weights were stored and subsequently used to predict the electric power for the test data.

Both kernel adaptive filters used the Gaussian kernel, defined in (14). For KRLS-T, the optimal kernel width was estimated empirically using the hyperparameter optimization technique described in Section 5.4.3 Van Vaerenbergh et al. (2012b), which is also available within the Kernel Adaptive Filtering Toolbox. For QKLMS-FB, a limited grid search with cross-validation was conducted to select the optimal kernel width.

The specific KRLS-T parameters were: dictionary size $\overline{m} = 50$, kernel width $\sigma = 2.5$, and regularization parameter $\lambda = 2 \times 10^{-4}$. The learning rate of QKLMS-FB was adjusted to $\eta = 0.7$, the dictionary size (\overline{m}) was limited to 500, and the kernel width was $\sigma = 1.118$. Additionally, QKLMS-FB requires a quantization threshold, which was adjusted to 0.1 and its significance forgetting factor was set to $\beta = 0.95$. The dictionary sizes for KRLS-T and QKLMS-FB were chosen such that both algorithms obtained similar execution times. RLS was initialized with $P_{t-1} = 1 \times 10^4 I$ and $\widehat{w}_{t-1} = \begin{bmatrix} 0 & 0 \end{bmatrix}^{\top}$. The neural network contained a single hidden layer with 100 neurons and it was trained through backpropagation using stochastic gradient descent with learning rate 0.02. The weights of the neural network were initialized using Xavier initialization Glorot & Bengio (2010).

6.2. Performance measure

The normalized root mean squared error (NRMSE)

NRMSE =
$$100 \left(1 - \frac{\|e\|}{\|y - \mu(y)\|} \right)$$

gives a meaningful scalar performance index that allows to compare the prediction quality of KRLS-T, QKLMS-FB, RLS, and NN. $\mu(\cdot)$ denotes the mean and *e* denotes the error vector. A NRMSE of 100 % indicates that the model explains the data perfectly. The lower the NRMSE, the poorer the model quality. Note that in extreme cases NRMSE becomes negative.

The error vectors for the compared algorithms were computed with the stored kernel expansion coefficients and estimated weights from the last time step of the training data $(t = m - \tau)$.

Specifically, the *i*-th element of *e* follows

$$e_{t=m-\tau+i} = y_{t=m-\tau+i} - \alpha_{t=m-\tau}^{\top} k_{t=m-\tau+i},$$

for KRLS-T and QKLMS-FB, whereas the errors of RLS follow

$$e_{t=m-\tau+i} = y_{t=m-\tau+i} - \widehat{w}_{t=m-\tau}^{\top} x_{t=m-\tau+i}.$$

6.3. Power prediction results

Figure 5 shows the estimated normalized electric power (denoted by $\hat{y}/\max y$) of KRLS-T, QKLMS-FB, RLS, and NN together with the measured output of the test data (y), for the first three trial runs. Most strikingly, RLS shows the poorest performance in all panels. This result was expected because RLS is a linear filter, which is not designed to map the non-linear inputoutput relation of model (4). The NN performs only slightly better compared to RLS: While the NN learns a nonlinear model, its weak performance is due to the fact that convergence in neural networks is very slow when trained on a single datum at a time. In practice, neural networks are trained using batches of data in each iteration to speed up convergence, and they are rarely used for online machine learning as in the present scenario. Comparing the two kernel adaptive filters, KRLS-T outperforms QKLMS-FB in all three trial runs (Figure 5a-Figure 5c). However, the precision of the predicted electric power of QKLMS-FB improves in Figure 5c.

A similar observation can be drawn by inspecting the values of NRMSE, which are represented for all trial runs in Table 4. The performance of KRLS-T is superior, followed by QKLMS-FB and NN. Again, RLS gave the poorest results.

Figure 6 shows scatter plots of the measured and predicted electric power for trial run № 3. A perfect model would show markers on a straight diagonal line that crosses the origin. The result of KRLS-T in Figure 6a shows high precision, whereas the non-linear shape of the markers of QKLMS-FB in Figure 6b indicate that some non-linearities of model (4) were not properly mapped by QKLMS-FB. Again, RLS and NN fail to find a proper solution for model (4), which is visible in the non-linear and somewhat random dispersed markers in Figure 6c and Figure 6d.



Figure 5: Online electric-power prediction on test data recorded at 2 Hz. The dictionary size of QKLMS-FB and KRLS-T was adjusted to 500 and 50, respectively.

6.4. Predicted mission energy

Table 5 shows the measured and predicted mission energy (2) of the test data for all trial runs. Predominantly, RLS gives the poorest estimate for the predicted mission energy. The results of NN are not stable: The predicted mission energy of NN distributes from very poor in trial run № 6, to good precision in trial run № 3, which indicates that NNs are not robust enough to be applied in practice for the problem herein. Comparing both kernel adaptive filters, KRLS-T is superior in all trial runs, closely followed by QKLMS-FB. Since the mission energy is the integral of the instantaneous power (2), the symmetrical dispersed modeling error of QKLMS-FB vanishes and therefore the gap in terms of precision to KRLS-T shrinks. In conclusion, QKLMS-FB may be preferred over KRLS-T in mission-energy prediction when the available computational power is very limited.

The algorithm complexity depends on the dictionary size \overline{m} (as shown in Table 2) and can be reduced largely by setting the dictionary size small enough. In this experiment, the dictionary size of QKLMS-FB was adjusted to 500. This setting is a compromise between speed and precision, though, as analyzed in the sequel.

6.5. Dependence of the dictionary size

The computational complexity of KRLS-T and QKLMS-FB grows with the dictionary size (\overline{m}) . Hence, the dictionary size

Table 4: Normalized root mean squared error for test data of each trial run. The first column shows the trial run number, while columns 2–5 point normalized root mean squared error of KRLS-T, QKLMS-FB, RLS, and NN.

	Normalized root mean squared error in %					
№	KRLS-T	QKLMS-FB	RLS	NN		
1	93.42	69.49	32.81	45.08		
2	89.39	81.23	43.88	33.07		
3	95.01	88.15	51.35	71.30		
4	94.90	73.98	60.18	81.25		
5	92.29	40.00	28.51	35.46		
6	89.88	82.35	52.52	65.08		
7	92.24	84.06	49.02	68.57		
8	88.71	33.36	43.99	47.36		
9	94.35	63.93	32.32	41.17		



Figure 6: Scatter plots of electric power prediction from trial run N° 3. The NRMSE of KRLS-T, QKLMS-FB, and NN were 95 %, 88.2 %, and 71.3 %, respectively. RLS delived the poorest result with NRMSE = 51.3 %. The dictionary size of KRLS-T and QKLMS-FB was adjusted to 50 and 500, respectively.

Table 5: Mission energy for test data of each trial run. The first column shows the trial run number. Column 2 provides the measured mission energy, and columns 3–6 show the predicted mission energy of KRLS-T, QKLMS-FB, RLS, and NN.

	Mission energy (E) in J				
№	measured	KRLS-T	QKLMS-FB	RLS	NN
1	$5.17\cdot 10^{6}$	$5.17\cdot 10^{6}$	$5.85\cdot 10^{6}$	$2.65\cdot 10^6$	$4.35\cdot 10^6$
2	$2.78 \cdot 10^6$	$3.02\cdot 10^6$	$2.46 \cdot 10^{6}$	$1.53 \cdot 10^{6}$	$3.30\cdot 10^6$
3	$2.49 \cdot 10^{6}$	$2.43 \cdot 10^6$	$2.64 \cdot 10^{6}$	$5.03 \cdot 10^5$	$2.58\cdot 10^6$
4	$-3.70 \cdot 10^{5}$	$-3.89 \cdot 10^5$	$-2.04 \cdot 10^{5}$	$-7.01 \cdot 10^{5}$	$-4.45 \cdot 10^{5}$
5	$4.60 \cdot 10^6$	$4.73 \cdot 10^6$	$6.58\cdot 10^6$	$2.80 \cdot 10^6$	$3.94\cdot 10^6$
6	$-4.44 \cdot 10^{5}$	$-5.22 \cdot 10^5$	$-6.43 \cdot 10^{5}$	$-1.06 \cdot 10^{6}$	$-9.85\cdot10^5$
7	$2.41 \cdot 10^{6}$	$2.25\cdot 10^6$	$2.28\cdot 10^6$	$3.92\cdot 10^5$	$1.80\cdot 10^6$
8	$3.85 \cdot 10^6$	$3.92\cdot 10^6$	$3.40 \cdot 10^{6}$	$1.63 \cdot 10^6$	$4.56 \cdot 10^{6}$
9	$4.62\cdot 10^6$	$4.63\cdot 10^6$	$3.25\cdot 10^6$	$2.45\cdot 10^6$	$3.83\cdot 10^6$

Table 6: Normalized root mean squared error for power prediction with 450 prediction steps of KRLS-T and QKLMS-FB on different dictionary size (\overline{m}). The computational complexity of KRLS-T and QKLMS-FB is $O(\overline{m}^2)$ and $O(\overline{m})$, respectively.

\overline{m}	$\frac{\text{KRLS-T}}{\overline{m}} \text{NRMSE in \%}$		QKLMS-FB \overline{m} NRMSE in %		
10	79.04	74.07	200	57.45	61.66
15	90.40	77.87	300	42.60	61.46
20	91.68	87.25	400	77.11	75.24
50	92.49	89.59	500	63.35	76.02
70	92.66	89.55	700	63.35	76.02
100	92.45	89.47	1,000	63.35	76.02
150	92.55	89.53	1,500	63.35	76.02
200	92.53	89.54	2,000	63.35	76.02
	№ 1	№ 2		№ 1	№2

should be chosen as small as possible. Table 6 lists the NRMSE for KRLS-T and QKLMS-FB as a function of the dictionary size, for trial run \mathbb{N} 1 and \mathbb{N} 2. The test data size (τ) was fixed at 450 samples in this experiment.

When given a larger dictionary size, and thus a larger memory, the algorithm's performance should increase. This behavior is correctly observed in the results of KRLS-T for $\overline{m} = 10, 15, 20$ and 50. Note that the performance of KRLS-T converges for $\overline{m} > 50$. However, we did not find a parameter setting for QKLMS-FB that led to the same desirable behavior in all trial runs. In particular, in trial run $N_{\odot} 1$, QKLMS-FB obtains its best results with a dictionary of 400 elements, and increasing the dictionary somewhat decreases its performance.

6.6. Large-scale learning experiment

In the final experiment, we analyze the performance of the described methods on a larger database, which we simulate by concatenating the data of all nine measured trial runs. We split the obtained data set in a training set, containing the first 16199 data, and a test set consisting of the last 1000 data. We then perform online machine learning on the training data set. In particular, each training step receives one input-output data pair and performs a single model update. After every 50 training

steps we test the NRMSE performance of each method on the 1000 samples of the test data set.

Figure 7 illustrates the NRMSE learning curves of each of the algorithms. All methods start with a fairly low NRMSE and then show a certain convergence. The RLS benchmark method converges quickly to a stable regime after 3000 training samples: Nevertheless, this filter yields the poorest results with NRMSE \approx 34%. The NN shows a slow convergence, fluctuating around an NRMSE value of 40 % during the first 8000 training steps. Eventually, the NRMSE of the NN improves to roughly 70% and becomes comparable with QKLMS-FB from 12 000 training samples onwards. However, the precision of NN shows a large dispersion, demonstrating that the algorithm lacks robustness in this application. QKLMS-FB converges faster than NN, but shows the same high fluctuation in NRMSE compared to NN. During the last third of the experiment, QKLMS-FB maintains an average precision level of 70%. The best performance is obtained by KRLS-T, in terms of fast convergence, high precision, and most stable results, making it clearly the superior learning algorithm in this experiment. In particular, KRLS-T converges to NRMSE \approx 90 % after 8000 training steps, and it does so in a robust manner.

7. Conclusions and Discussion

We studied the application of real-time capable machine learning techniques to the problem of power and mission energy prediction of an electric vehicle. First, we posed the problem as a black-box prediction scenario, which allows to operate directly on vehicle sensor data. Compared to conventional white-box or graybox models from the literature, the proposed approach allows us to formulate the problem as a non-linear mapping of the measured inputs (velocity and longitudinal acceleration) to the measured power. Because of this direct non-linear mapping, knowledge of vehicle-specific parameters (vehicle mass, coefficient of rolling resistance, gear ratio, etc.) or efficiency maps is not required, and the costly process of building a drive-train model can be avoided.

Then, we applied two state-of-the-art machine learning methods from the class of kernel adaptive filters to this problem. The prediction accuracy for power and mission energy of both kernel



Figure 7: Learning curves, presented in terms of NRMSE, for KRLS-T, QKLMS-FB, RLS, and NN in the large-scale learning experiment. The NRMSE values are plotted after every 50 training steps and the last 1000 samples of the large-scale data served as the test set.

adaptive filters is superior to the benchmark of a standard linear filter and a neural network. In particular, the kernel recursive least-squares tracker algorithm obtains the best performance for power prediction, whereas the computationally cheaper fixedbudget quantized kernel least mean squares algorithm is worth considering in mission energy prediction applications.

The application of kernel adaptive filters in vehicle control units is novel. In order to apply kernel adaptive filters safely to control units of vehicles that operate in changing environmental conditions, many research questions lie open. In the remainder of this section we briefly point out some of these future research questions.

The applied fixed-budget kernel adaptive filters (KRLS-T and QKLMS-FB) are real-time capable. However, numerical analysis is required to study the behavior of these algorithms in single precision control units. In the field of Kalman filtering such studies have been carried out by comparing the standard Kalman filter implementation (Ljung, 1999, p. 369) and the numerous variants (square root form, Potter's algorithm, regularized versions (Simon, 2006, Chapter 6)) that were specifically designed to apply the Kalman filter in electronic control units. In the present work, KRLS-T and QKLMS-FB were trained in a double precision machine.

Furthermore, in order to guarantee safe operation, any algorithm applied in this context should provide a confidence measure in addition to the estimated output. While KRLS-T already delivers such a confidence measure Van Vaerenbergh et al. (2012a), QKLMS-FB does not and therefore requires further study. In practice, the confidence measure allows to decide if the adapted model is reliable enough or if the decision should switch back to some more traditional method. The latter may occur when there is a substantial mismatch between the estimated model and the reality, which could be detected in real-time by employing changepoint detectors Tartakovsky et al. (2014).

References

References

- Adepetu, A., & Keshav, S. (2017). The relative importance of price and driving range on electric vehicle adoption: Los angeles case study. *Transportation*, 44, 353–373. doi:10.1007/s11116-015-9641-y.
- Aronszajn, N. (1950). Theory of reproducing kernels. Transactions of the American Mathematical Society, 68, 337–404. doi:10.2307/1990404.
- Asamer, J., Graser, A., Heilmann, B., & Ruthmair, M. (2016). Sensitivity analysis for energy demand estimation of electric vehicles. *Transportation Research Part D: Transport and Environment*, 46, 182 – –199. doi:10.1016/ j.trd.2016.03.017.
- Broomhead, D. S., & Lowe, D. (1988). Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2, 321–355.
- Carley, S., Krause, R., Lane, B., & Graham, J. (2013). Intent to purchase a plug-in electric vehicle: A survey of early impressions in large US cites. *Transportation Research Part D: Transport and Environment*, 18, 39–45. doi:10.1016/j.trd.2012.09.007.
- Chen, B., Zhao, S., Zhu, P., & Príncipe, J. C. (2012a). Quantized kernel least mean square algorithm. *IEEE Transactions on Neural Networks and Learning Systems*, 23, 22–32.
- Chen, Y., Huang, C., Kuo, Y., & Wang, S. (2012b). Artificial neural network for predictions of vehicle drivable range and period. In 2012 IEEE International Conference on Vehicular Electronics and Safety (ICVES) (pp. 329–333). doi:10.1109/ICVES.2012.6294324.
- Culver, M. (2015). Norway leads global electric vehicle market, IHS says. URL: http://press.ihs.com/press-release/automotive/ norway-leads-global-electric-vehicle-market-ihs-says accessed 2015/09/28.
- Cuma, M., & Koroglu, T. (2015). A comprehensive review on estimation strategies used in hybrid and battery electric vehicles. *Renewable and Sustainable Energy Reviews*, 42, 51–531. doi:10.1016/j.rser.2014.10. 047.
- De Bruyne, S., Van der Auweraer, H., Diglio, P., & Anthonis, J. (2011). Online estimation of vehicle inertial parameters for improving chassis control systems. In *Proceedings of the 18th World Congress, The International Federation of Automatic Control* (pp. 1814–1819). volume 18. doi:10.3182/ 20110828-6-IT-1002.03379.
- Dütschke, E., Schneider, U., Sauer, A., Wietschel, M., Hoffmann, J., & Domke, S. (2012). Roadmap zur Kundenakzeptanz: Zentrale Ergebnisse der sozialwissenschaftlichen Begleitforschung in den Modellregionen. Technical Report Fraunhofer ISI.
- Engel, Y., Mannor, S., & Meir, R. (2004a). The kernel recursive least-squares algorithm. *IEEE Transactions on Signal Processing*, 52, 2275–2285. doi:10. 1109/TSP.2004.830985.
- Engel, Y., Mannor, S., & Meir, R. (2004b). The kernel recursive least squares algorithm. *IEEE Transactions on Signal Processing*, 52, 2275–2285. doi:10. 1109/TSP.2004.830985.
- Faraj, M., & Basir, O. (2016). Range anxiety reduction in battery-powered vehicles. In 2016 IEEE Transportation Electrification Conference and Expo (ITEC) (pp. 1–6). doi:10.1109/ITEC.2016.7520190.
- Fathy, H., Kang, D., & Stein, J. (2008). Online vehicle mass estimation using recursive least squares and supervisory data extraction. In 2008 American Control Conference (pp. 1842–1848). doi:10.1109/ACC.2008.4586760.
- Fergusson, M. (2016). Electric Vehicles in Europe 2016: Approaching Adolescence. Technical Report European Federation for Transport and Environment AISBL Brussels, Belgium.
- Franke, F., & Krems, J. (2013). What drives range preferences in electric vehicle users? *Transport Policy*, 30, 56–62. doi:10.1016/j.tranpol.2013.07. 005.
- Franke, T., Neumann, I., Bühler, F., Cocron, P., & Krems, J. (2012). Experiencing range in an electric vehicle: Understanding psychological barriers. *Applied Psychology*, 61, 368–391. doi:10.1111/j.1464-0597.2011.00474.x.
- De la Fuente Layos, L. (2007). *Mobilität im Personenverkehr in Europa*. Technical Report Eurostat.

- Fünfgeld, S., Holzäpfel, M., Frey, M., & Gauterin, F. (2017). Stochastic forecasting of vehicle dynamics using sequential monte carlo simulation. *IEEE Transactions on Intelligent Vehicles*, 2, 111–122. doi:10.1109/TIV. 2017.2723823.
- Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (pp. 249–256).
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
- Grunditz, E., & Thiringer, T. (2016). Characterizing bev powertrain energy consumption, efficiency, and range during official and drive cycles from gothenburg, sweden. *IEEE Transactions on Vehicular Technology*, 65, 3964– 3980. doi:10.1109/TVT.2015.2492239.
- Hayes, J., de Oliveira, R., Vaughan, S., & Egan, M. (2011). Simplified electric vehicle power train models and range estimation. In 2011 IEEE Vehicle Power and Propulsion Conference (VPPC) (pp. 1–5). doi:10.1109/VPPC.2011. 6043163.
- International Energy Agency (2016). Global EV Outlook 2016: Beyond one million electric cars. Technical Report International Energy Agency Paris, France.
- Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME – Journal of Basic Engineering*, 82 (Series D), 35–45.
- Kiencke, U., & Nielsen, L. (2005). Automotive control systems : for engine, driveline, and vehicle. (2nd ed.). Berlin: Springer. doi:10.1007/b137654.
- Liu, W., Principe, J., & Haykin, S. (2011). Kernel Adaptive Filtering: A Comprehensive Introduction volume 57. Wiley.
- Ljung, L. (1999). *System identification : theory for the user*. Prentice-Hall information and system sciences series. Upper Saddle River, NJ: Prentice Hall.
- McIntyre, M., Ghotikar, T., Vahidi, A., Song, X., & Dawson, D. (2009). A two-stage Lyapunov-based estimator for estimation of vehicle mass and road grade. *IEEE Transactions on Vehicular Technology*, 58, 3177–3185. doi:10.1109/TVT.2009.2014385.

Plackett, R. L. (1950). Some theorems in least squares. Biometrika, 37, 149-157.

Rasmussen, C. E., & Williams, C. K. I. (2006). Gaussian Processes for Machine Learning. MIT Press.

- Rauh, N., Franke, T., & Krems, J. F. (2017). First-time experience of critical range situations in bev use and the positive effect of coping information. *Transportation research part F: traffic psychology and behaviour*, 44, 30–41.
- Rhode, S., & Gauterin, F. (2013). Online estimation of vehicle driving resistance parameters with recursive least squares and recursive total least squares. In 2013 IEEE Intelligent Vehicles Symposium (IV) (pp. 269–276). doi:10.1109/ IVS.2013.6629481.
- Rhode, S., Hong, S., Hedrick, J., & Gauterin, F. (2016). Vehicle tractive force prediction with robust and windup-stable kalman filters. *Control Engineering Practice*, 46, 37–50. doi:10.1016/j.conengprac.2015.10.002.
- Richard, C., Bermudez, J. C. M., & Honeine, P. (2009). Online prediction of time series data with kernels. *IEEE Transactions on Signal Processing*, 57, 1058–1067. doi:10.1109/TSP.2008.2009895.
- Sarrafan, K., Muttaqi, K. M., Sutanto, D., & Town, G. E. (2017). An intelligent driver alerting system for real-time range indicator embedded in electric vehicles. *IEEE Transactions on Industry Applications*, .
- Sayed, A. H. (2003). Fundamentals of Adaptive Filtering. Wiley-IEEE Press.
- Schölkopf, B., Herbrich, R., & Smola, A. J. (2001). A generalized representer theorem. In *Computational learning theory* (pp. 416–426). Springer. doi:10. 1007/3-540-44581-1_27.
- Schölkopf, B., & Smola, A. J. (2002). Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. The MIT Press.
- Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge university press.
- Simon, D. (2006). Optimal state estimation : Kalman, H_{∞} and nonlinear approaches. Hoboken, NJ: Wiley-Interscience.
- Tannahill, V. R., Sutanto, D., Muttaqi, K. M., & Masrur, M. A. (2014). Future vision for reduction of range anxiety by using an improved state of charge estimation algorithm for electric vehicle batteries implemented with low-cost microcontrollers. *IET Electrical Systems in Transportation*, 5, 24–32.
- Tartakovsky, A., Nikiforov, I., & Basseville, M. (2014). Sequential analysis: Hypothesis testing and changepoint detection. Chapman and Hall/CRC.
- Vahidi, A., Stefanopoulou, A., & Peng, H. (2005). Recursive least squares with forgetting for online estimation of vehicle mass and road grade: theory and experiments. *Vehicle System Dynamics*, 43, 31–55. doi:10.1080/

00423110412331290446.

- Van Vaerenbergh, S., Lazaro-Gredilla, M., & Santamaria, I. (2012a). Kernel recursive least-squares tracker for time-varying regression. *Neural Networks* and Learning Systems, IEEE Transactions on, 23, 1313–1326. doi:10.1109/ TNNLS.2012.2200500.
- Van Vaerenbergh, S., & Santamaría, I. (2013). A comparative study of kernel adaptive filtering algorithms. In 2013 IEEE Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE) (pp. 181–186). Software available at https://github.com/steven2358/kafbox/.
- Van Vaerenbergh, S., & Santamaría, I. (2014). Online regression with kernels. In *Regularization, Optimization, Kernels, and Support Vector Machines* Machine Learning & Pattern Recognition Series chapter 21. (pp. 477–501). New York: Chapman and Hall/CRC.
- Van Vaerenbergh, S., Santamaria, I., L., W., & Principe, J. (2010). Fixedbudget kernel recursive least-squares. In 2010 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP) (pp. 1882–1885). doi:10.1109/ICASSP.2010.5495350.
- Van Vaerenbergh, S., Santamaría, I., & Lázaro-Gredilla, M. (2012b). Estimation of the forgetting factor in kernel recursive least squares. In 2012 IEEE International Workshop on Machine Learning for Signal Processing (MLSP). doi:10.1109/MLSP.2012.6349749.
- Van Vaerenbergh, S., Via, J., & Santamaria, I. (2006). A sliding-window kernel rls algorithm and its application to nonlinear channel identification. In 2006 IEEE International Conference on Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings (pp. 789–792). volume 5. doi:10.1109/ICASSP.2006.1661394.
- Vapnik, V. N. (1995). The Nature of Statistical Learning Theory. New York, NY, USA: Springer-Verlag New York, Inc.
- Wietschel, M., Plötz, P., Kühn, A., & Gnann, T. (2013). Market evolution scenarios for electric vehicles. Technical Report Fraunhofer ISI.
- Wu, X., Freese, D., Cabrera, A., & Kitch, W. (2015). Electric vehicles' energy consumption measurement and estimation. *Transportation Research Part D: Transport and Environment*, 34, 52–67. doi:10.1016/j.trd.2014.10.007.
- Yu, H., Tseng, F., & McGee, R. (2012). Driving pattern identification for EV range estimation. In 2012 IEEE International Electric Vehicle Conference (IEVC) (pp. 1–7). doi:10.1109/IEVC.2012.6183207.
- Zhao, S., Chen, B., Zhu, P., & Principe, J. (2013). Fixed budget quantized kernel least-mean-square algorithm. *Signal Processing*, 93, 2759–2770. doi:10.1016/j.sigpro.2013.02.012.
- Zhiyuan, C., & Bing, L. (2018). Lifelong Machine Learning. (2nd ed.). Morgan & Claypool. doi:10.2200/S00832ED1V01Y201802AIM037.
- Ziegler, J., Bender, P., Schreiber, M., Lategahn, H., Strauss, T., Stiller, C., Dang, T., Franke, U., Appenrodt, N., Keller, C., Kaus, E., Herrtwich, R., Rabe, C., Pfeiffer, D., Lindner, F., Stein, F., Erbs, F., Enzweiler, M., Knöppel, C., Hipp, J., Haueis, M., Trepte, M., Brenk, C., Tamke, A., Ghanaat, M., Braun, M., Joos, A., Fritz, H., Mock, H., Hein, M., & Zeeb, E. (2014). Making Bertha drive; an autonomous journey on a historic route. *IEEE Intelligent Transportation Systems Magazine*, 6, 8–20. doi:10.1109/MITS.2014.2306552.