Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 53-2 (2020) 316–321

A novel Multiplicative Polynomial Kernel for Volterra series identification

Alberto Dalla Libera, Ruggero Carli, Gianluigi Pillonetto *

* *Authors are with the Department of Information Engineering, University of Padova, Via Gradenigo 6/B, 35131 Padova, Italy*

dallaliber@dei.unipd.it, carlirug@dei.unipd.it, giapi@dei.unipd.it

Abstract: Volterra series is especially useful for nonlinear system identification, also thanks to its capability to approximate a broad range of input-output maps. However, its identification from a finite set of data is hard, due to the curse of dimensionality. Recent approaches have shown how regularization strategies can be useful for this task. In this paper, we propose a new regularization network for Volterra models identification. It relies on a new kernel given by the product of basic building blocks. Each block contains some unknown parameters that can be estimated from data using marginal likelihood optimization or cross-validation. In comparison with other algorithms proposed in the literature, numerical experiments show that our approach allows to better select the monomials that really influence the system output, much increasing the prediction capability of the model. The method immediately extends also to polynomial NARMAX models.

Keywords: Nonlinear system identification; Nonparametric methods; Time series modelling

1. INTRODUCTION

In many real world applications, linear models are not able to adequately describe dynamic systems. This can be due to the presence of saturations, quantizers or static nonlinearities at the input and/or the output Ljung (1999)[Section 5]. Even if some insight on the nonlinearities can be available, the formulation of parametric models from finite data records is a difficult task Haber and Unbehauen (1990); Lind and Ljung (2008); Sjöberg et al. (1995). In particular, nonlinear system identification is often seen as an extended parametric regression where the choice of regressors and basis functions plays a crucial role. In this context, Volterra series is especially useful since it can represent a broad range of nonlinear systems Rugh (1980); Boyd and Chua (1985); Cheng et al. (2017). When working in discrete-time, such models correspond to Taylor expansions of the input-output map. Indeed, a truncated Volterra series describes the system as the sum of all the possible monomials in the past inputs and outputs, up to a certain order. The problem is however the curse of dimensionality: the number of monomials grows quickly w.r.t. the polynomial degree and the system memory (given e.g. by the number of past input values that determine the output). Thus, a careful selection of the relevant components to be included in the model is crucial to control the complexity of the estimator, a problem known as regressors selection.

Suboptimal solutions are often searched through greedy approaches like forward/backward subset selection, see for instance Chen et al. (1989); Billings et al. (1989). These regressor selection methods have however difficulties in handling high-dimensional regression spaces. An interesting option is joint estimation and variable selection. This can be performed using e.g. the ℓ_1-norm regularizer which leads to the famous LASSO Tibshirani (1996).

More recent approaches proposed to deal with the dimensionality of Volterra models can be found in Birpoutsoukis et al. (2017); Stoddard et al. (2017). In particular, in Birpoutsoukis et al. (2017), inspired by ideas developed for linear system identification in Pillonetto and De Nicolao (2010), the authors proposed a regularization strategy suitable for Volterra series with smooth exponential decay.

An alternative route to the approaches mentioned above is the use of kernel-methods, which lead e.g. to the so called regularization networks Poggio and Girosi (1990). Here, an unknown function is determined as the minimizer of an objective that is sum of two terms: a quadratic loss and regularizer defined by a positive definite kernel. The choice of the kernel has a major effect on the quality of the estimate since it encodes the expected properties of the function to reconstruct. Just looking at the function to reconstruct as the unknown system (input-output map), in recent years kernel-based approaches have been widely exploited also for nonlinear system identification and prediction, see for instance Espinoza et al. (2005); Hall et al. (2012).

Another popular model is the polynomial kernel, which has a deep connection with Volterra series. In fact, it implicitly encodes all the monomials up to the desired degree r, a kernel parameter tunable by the user. Regularization networks for efficient Volterra identification that exploit this kernel can be found in Franz and Schölkopf (2006).

The approach described in this paper is based on a new polynomial kernel, named *Multiplicative Polynomial kernel* (MPK). Similarly to the polynomial kernel, the MPK

⋆ This paper has been supported by project PRIN 2015 2015PJ28EP, PI and National Coordinator Gianluigi Pillonetto.

10.1016/j.ifacol.2020.12.179

encodes all the monomials up to degree r, but it is defined by the product of r linear kernels, each one equipped with a distinct set of hyperparameters. The MPK has some important features w.r.t. the polynomial kernel used in Franz and Schölkopf (2006). As already said, the polynomial kernel depends only on the polynomial degree r and it encodes a number of monomials rapidly increasing with r and the system memory. When plugged in a regularization network, it induces a penalty that cannot promote any sparsity in the solution. On the contrary, the MPK is equipped with an augmented set of hyperparameters, which allows promoting the monomials w.r.t. their maximum relative degree, improving regularization performance. Tests performed both in simulation and with data coming from a real system show that the MPK hyperparameters can be tuned via marginal likelihood optimization or cross-validation.

The paper is organized as follows. In Section 2 we provide a brief overview on Volterra series and the main identification approaches adopted. In Section 3 we highlight some critical aspects of the standard polynomial kernel, and we introduce our kernel function, the Multiplicative Polynomial Kernel, highlighting its regularization capabilities. Finally, in Section 4 we report numerical results, in which we compare performance of the proposed kernel and the standard polynomial kernel.

2. BACKGROUND

2.1 Volterra series

Let u_k and z_k be the one dimensional input and output signals at time k. When modeling the system response with a discrete time Volterra series of order r the noisy output y_k is assumed to be the sum of measurement noise and $r+1$ contributions acting on the lagged inputs u_k, u_{k-1}, u_{k-2} .... Assume that the system has finite memory m, and define the input vector $\boldsymbol{u}_k = [u_k \ \dots \ u_{k-m}]$. Then we have

$$y_k = z_k + e_k = h_0 + \sum_{i=1}^{r} H_i(\boldsymbol{u}_k) + e_k \ , \tag{1}$$

where $e_k \sim N\left(0, \sigma_n^2\right)$ is the measurement noise, h_0 is a constant accounting for the zero-order Volterra contribution, while the H_i are the i-th Volterra contributions. In particular, each H_i is defined as the convolution between $\boldsymbol{u}_k$ and a Volterra map h_i, namely,

$$H_i(\boldsymbol{u}_k) = \sum_{\tau_1=0}^{m} \cdots \sum_{\tau_i=0}^{m} h_i\left(\tau_1, \dots, \tau_i\right) \prod_{\tau=\tau_1}^{\tau_i} u_{k-\tau}. \tag{2}$$

In this paper, we consider symmetric Volterra series, i.e., given a set of lags $\tau_1, \dots, \tau_i$, the value of h_i is independent on the lags order. For instance, with $i = 2$ we have $h_2\left(\tau_1, \tau_2\right) = h_2\left(\tau_2, \tau_1\right)$.
Alternatively, each H_i term in (1) can be rewritten more compactly as an inner product. Let $\boldsymbol{\phi}_i(\boldsymbol{u}_k)$ be a vector collecting all the distinct monomials with degree i in the components of $\boldsymbol{u}_k$. We have

$$H_i(\boldsymbol{u}_k) = \boldsymbol{\phi}_i^T\left(\boldsymbol{u}_k\right) \boldsymbol{w}_i,$$

where $\boldsymbol{w}_i$ is the vector collecting the distinct h_i coefficients ordered in accordance with $\boldsymbol{\phi}_i$, and opportunely scaled to account for repetitions due to symmetry. More precisely, the $\boldsymbol{w}_i$ entry associated to the monomial $\prod_{j=0}^{m} u_{k-j}^{d_j}$ is the product between the correspondent h_i coefficient scaled by the multinomial coefficient $\binom{i}{d_0,\dots,d_m}$. Based on the above definitions, (1) can be rewritten as

$$y_k = \boldsymbol{\phi}\left(\boldsymbol{u}_k\right)^T \boldsymbol{w} + e_k, \tag{3}$$

where

$$\boldsymbol{\phi}\left(\boldsymbol{u}_k\right) = \left[h_0 \, \boldsymbol{\phi}_1^T(\boldsymbol{u}_k) \ \dots \ \boldsymbol{\phi}_r^T(\boldsymbol{u}_k)\right]^T, \tag{4}$$

$$\boldsymbol{w}\left(\boldsymbol{u}_k\right) = \left[w_0 \, \boldsymbol{w}_1^T \ \dots \ \boldsymbol{w}_r^T\right]^T. \tag{5}$$

The authors in Birpoutsoukis et al. (2017) have proposed to learn the input-output relations by directly estimating the elements of $\boldsymbol{w}$. This estimation is performed by solving a least square problem defined by (3), given a data set of input-output measurements $D = \{(\boldsymbol{u}_k, y_k)\,, k = 1, \dots T\}$. It is worth stressing that the applicability of algorithms based on a least-square approach is strongly limited by the high computational and memory requirements related to the dimension of $\boldsymbol{w}$. Indeed, the number of Volterra coefficients grows rapidly with the system memory m and the Volterra order r. More precisely, assuming that the Volterra maps are symmetric, we have that $\boldsymbol{w}_i$ is composed of $N_i = \binom{m+i}{i}$ elements, leading to a total number of $N = 1 + \sum_{i=1}^{r} N_i$ parameters to be estimated.

2.2 Polynomial kernel and Volterra series

An alternative solution to accomplish the Volterra series identification has been proposed in Franz and Schölkopf (2006). Instead of formulating the identification problem directly w.r.t. $\boldsymbol{w}$, the authors rely on kernel based techniques. The input-output map f is assumed to belong to a *reproducing kernel Hilbert space* (RKHS) Scholkopf and Smola (2001), defined by a kernel function $k(\boldsymbol{u}_k, \boldsymbol{u}_j)$. Given an input-output dataset D like that previously introduced, $\hat{f}$, the estimate of f, is obtained solving the following problem,

$$\underset{f \in H}{\arg\min} \sum_{t=1}^{T} \left(y_t - z(\boldsymbol{u}_t)\right)^2 + \gamma^2 ||f||_H^2, \tag{6}$$

where the first term of the loss function accounts for the adherence to experimental data, while the second is the regularization term, given by the squared RKHS norm of f; the balance between these two contributions can be controlled tuning the hyperparameter γ. According to the *representer theorem*, $\hat{f}$ is expressed in closed form as

$$\hat{f}(\boldsymbol{u}_k) = \hat{z}_k = \sum_{t=1}^{T} \alpha_t k(\boldsymbol{u}_k, \boldsymbol{u}_t), \tag{7}$$

where $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_T]^T$ is equal to $(K + \gamma^2 I_T)^{-1}\boldsymbol{y}$, $\boldsymbol{y} = [y_1, \dots, y_T]^T$ denotes the vector containing all the output measurements, and K is the Kernel matrix, i.e. its (k, j) entry is $K_{k,j} = k(\boldsymbol{u}_k, \boldsymbol{u}_j)$.
For our future use, it is also useful recalling the following fundamental facts regarding RKHS theory. Under mild assumptions, a kernel function admits an expansion (possibly infinite) in terms of basis functions ϕ_q, namely,

$$k(\boldsymbol{u}_k, \boldsymbol{u}_j) = \sum_{q} \lambda_q \phi_q(\boldsymbol{u}_k) \phi_q(\boldsymbol{u}_j), \tag{8}$$

where λ_q are positive scalars. It can then be proved that any function in the RKHS induced by the above kernel has the representation

$$f(\boldsymbol{u}_k) = \sum_q c_q \phi_q(\boldsymbol{u}_k), \tag{9}$$

for suitable coefficients c_q. In addition, if all the basis functions ϕ_q are linearly independent, one also has

$$||f||_H^2 = \sum_q \frac{c_q^2}{\lambda_q}. \tag{10}$$

This last relation shows how the λ_q coefficients are related to each ϕ_q in determining the regularization term present in (6). In particular, small values of λ_q entail high penalization of ϕ_q.

As far as the kernel function is concerned, in Franz and Schölkopf (2006) the authors considered the polynomial kernel. In particular, we discuss the inhomogeneous polynomial kernel, defined as

$$k^{(r)}(\boldsymbol{u}_k, \boldsymbol{u}_j) = \left(1 + \boldsymbol{u}_k^T \boldsymbol{u}_j\right)^r, \tag{11}$$

where r is a tunable hyperparameter corresponding to the degree of the polynomial kernel. As showed in Scholkopf and Smola (2001), the polynomial kernel in (11) admits an expansion in the monomials in $\boldsymbol{u}_k$, with degree up to r. Namely, referring to (8) and (10), we have that the ϕ_q corresponds to the elements of the $\boldsymbol{\phi}$ vector defined in (4); accordingly, $1/\lambda_q$ then defines the penalty assigned to the relative monomial.

We conclude this subsection with a computational note. The computation of $\hat{f}$ involves the inversion of a $T \times T$ matrix, see (7). The number of operations so scales with the cube of the data set size, and there is no direct dependence on N, the dimensions of $\boldsymbol{\phi}$, allowing the use of high-order Volterra models.

3. PROPOSED KERNEL

The Volterra series learning strategy we propose is based on a novel polynomial kernel, called *Multiplicative Polynomial Kernel* (MPK). Compared to the standard polynomial kernel reported in (11), our kernel is equipped with a set of parameters which allows to assign suitable priors to the different basis functions of the RKHS, thus leading to better performance in terms of estimation and generalization. Before describing the proposed kernel function we highlight some critical issues of standard inhomogeneous polynomial kernel.

3.1 Penalties induced by (11)

As stated in Rasmussen and Williams (2006) (Chapter 4.2.2), polynomial kernels are not widely used in regression problems, since they are prone to overfitting, in particular, in presence of high dimensional inputs and when the degree is greater than two. Indeed, in the kernel formulation given in (11) there are not parameters that allow to weigh differently the monomials composing the RKHS.

To clarify this concept we consider a simple example, a third order Volterra series with $m = 1$ defined as follow

$$f(\boldsymbol{u}_k) = u_k^3 + u_k^2 u_{k-1} + 0.5. \tag{12}$$

We compute the λ_q obtained with the standard polynomial kernel expanding (11), and comparing the result with (8). For the sake of clarity, we will denote with $\lambda_{d_0,\dots,d_m}$ the penalty coefficient associated to the monomial $\prod_{\tau=0}^{m} u_{k-\tau}^{d_\tau}$. The kernel expansion is

$$\begin{aligned} k^{(3)}(\boldsymbol{u}_i, \boldsymbol{u}_j) = {} & u_i^3 u_j^3 + u_{i-1}^3 u_{j-1}^3 \\ & + 3u_i^2 u_{i-1} u_j^2 u_{j-1} + 3u_i u_{i-1}^2 u_j u_{j-1}^2 \\ & + 3u_i^2 u_j^2 + 3u_{i-1}^2 u_{j-1}^2 + 6u_i u_{i-1} u_j u_{j-1} \\ & + 3u_i u_j + 3u_{i-1} u_{j-1} + 1. \end{aligned}$$

Then, by inspection, we obtain

$$\begin{aligned} &\lambda_{3,0} = \lambda_{0,3} = 1,\ \lambda_{2,1} = \lambda_{1,2} = 3, \\ &\lambda_{2,0} = \lambda_{0,2} = 3,\ \lambda_{1,1} = 6, \\ &\lambda_{1,0} = \lambda_{0,1} = 3,\ \lambda_{0,0} = 1. \end{aligned}$$

These values show that (11) assigns penalties based on the monomial degree, and promoting mixed terms. This trend might not be representative of the Volterra kernel, leading to the need of more training data to obtain accurate estimates. For instance, consider the function reported in (12). It is evident that the λ values obtained with (11) do not describe properly the contributions of the different monomials, since, for example, the higher values of λ are assigned to monomials that are not present.

3.2 Multiplicative Polynomial Kernel

The kernel function we propose to model the r order Volterra series is given by the product of r linear kernels, and it is formally defined as

$$k^{(r)}(\boldsymbol{u}_k, \boldsymbol{u}_j) = \prod_{i=1}^{r} \left(\sigma_{0_i} + (\boldsymbol{u}_k)^T \Sigma_i \boldsymbol{u}_j\right), \tag{13}$$

where the matrices $\Sigma_i \in \mathbb{R}^{(m+1)\times(m+1)}$ are diagonal. In particular, for each i we have $\Sigma_i = diag\left(\left[\sigma_0^{(i)}, \dots, \sigma_m^{(i)}\right]\right)$, with the diagonal elements greater of equal than zero. Exploiting the kernel properties it can be easily shown that the function defined in (13) is a well-defined kernel function, since it is the product of several valid kernel functions, see Rasmussen and Williams (2006).

3.3 Penalties induced by the MPK

In this subsection we analyze the advantages of the proposed kernel function, focusing on the role played by the kernel parameters. To this aim, we consider the example analyzed in the previous subsection, that is, the identification of the input-output behavior of a Volterra series with $r = 3$ and $m = 1$. Starting from the kernel definition given in (13), through standard algebraic computations, we can derive the penalties coefficients as functions of the MPK parameters. In particular, the penalties assigned to monomials of degree three are

$$\begin{aligned} &\lambda_{3,0} = \prod_{j=1}^{3} \sigma_0^{(j)},\ \ \lambda_{0,3} = \prod_{j=1}^{3} \sigma_1^{(j)}, \\ &\lambda_{2,1} = \sum_{j=1}^{3} \sigma_1^{(j)} \prod_{l\neq j} \sigma_0^{(l)},\ \ \lambda_{1,2} = \sum_{j=1}^{3} \sigma_0^{(j)} \prod_{l\neq j} \sigma_1^{(l)}, \end{aligned}$$

the ones assigned to monomials of degree two are

$$\lambda_{2,0} = \sum_{j=1}^{3} \sigma_{0_j} \prod_{l \neq j} \sigma_0^{(l)}, \quad \lambda_{0,2} = \prod_{j=1}^{3} \sigma_{0_j} \prod_{l \neq m} \sigma_1^{(l)},$$
$$\lambda_{1,1} = \sum_{j=1}^{3} \sigma_{0_j} \sum_{l_1 \neq l_2 \neq j} \sigma_0^{(l_1)} \sigma_1^{(l_2)},$$

and, finally, the ones assigned to monomials of degree one and zero are

$$\lambda_{1,0} = \sum_{j=1}^{3} \sigma_0^{(j)} \prod_{l \neq j} \sigma_{0_l}, \qquad \lambda_{0,1} = \sum_{j=1}^{3} \sigma_1^{(j)} \prod_{l \neq j} \sigma_{0_l},$$
$$\lambda_{0,0} = \prod_{j=1}^{3} \sigma_{0_j},$$

Some interesting insights can be obtained from the previous penalties expressions. Notice that the MPK parameters allow penalizing the monomials w.r.t. their relative degree. For instance, consider the penalties assigned to monomials that contains u_k, namely, $\lambda_{3,0}$, $\lambda_{2,1}$, $\lambda_{1,2}$, $\lambda_{2,0}$, $\lambda_{1,1}$, and $\lambda_{1,0}$, as function of $\sigma_0^{(j)}$, with $j = 1, 2, 3$. Analyzing these expression we can appreciate that to promote monomials in which u_k appears with degree i at least i of the $\lambda_0^{(j)}$ parameters need to be significantly greater than zero. On the contrary, to penalize monomials in which u_k appears with relative degree greater than i we need to set to zero $r-i$ of the $\sigma_0^{(j)}$ elements. Thus, referring to the test function in (12), where the maximum relative degree of u_k and u_{k-1} are, respectively, 3 and 1, we can exclude part of the monomials that do not influence the input-output defining

$$\begin{aligned} \sigma_{0_1} = 1 \ , \ \left[\sigma_0^{(1)} \, \sigma_1^{(1)}\right] = [1\,1]\,, \\ \sigma_{0_2} = 1 \ , \ \left[\sigma_0^{(2)} \, \sigma_1^{(2)}\right] = [1\,0]\,, \\ \sigma_{0_3} = 1 \ , \ \left[\sigma_0^{(3)} \, \sigma_1^{(3)}\right] = [1\,0]\,, \end{aligned} \tag{14}$$

which determine the following penalties,

$$\begin{aligned} &\lambda_{3,0} = 1,\ \lambda_{2,1} = 1,\ \lambda_{0,3} = \lambda_{1,2} = 0, \\ &\lambda_{2,0} = 2,\ \lambda_{0,2} = 0,\ \lambda_{1,1} = 2, \\ &\lambda_{1,0} = 2,\ \lambda_{0,1} = 1,\ \lambda_{0,0} = 1. \end{aligned} \tag{15}$$

The hyperparameters tuning can be accomplished relying on empirical methods, like cross validation, or optimizing a given loss function, like *Marginal Likelihood* (ML).

3.4 Parametrization of the Σ_i matrices

Notice that the MPK is given by the product of r equal blocks. Consequently, permuting the Σ_i matrices we obtain different configurations of the MPK hyperparameters associated to the same set of penalties. For instance, referring to the test case analyzed in the previous subsection, the configuration reported in (14) and the following configuration are associated to the same set of penalties, i.e. (15),

$$\begin{aligned} \sigma_{0_1} = 1 \ , \ \left[\sigma_0^{(1)} \, \sigma_1^{(1)}\right] = [1\,0]\,, \\ \sigma_{0_2} = 1 \ , \ \left[\sigma_0^{(2)} \, \sigma_1^{(2)}\right] = [1\,1]\,, \\ \sigma_{0_3} = 1 \ , \ \left[\sigma_0^{(3)} \, \sigma_1^{(3)}\right] = [1\,0]\,. \end{aligned}$$

When optimizing the hyperparameters by ML, this fact could lead to undesired behaviors, due to the presence of several local maxima. To avoid such behaviors, we propose an iterative parametrization of the Σ_i diagonal elements. More specifically, the Σ_i, are defined by a backward iteration as follows,

$$\begin{aligned} \Sigma_r &= diag\left(\left[a_0^{(r)}, \dots, a_m^{(r)}\right]\right), \\ \Sigma_i &= \Sigma_{i+1} + diag\left(\left[a_0^{(i)}, \dots, a_m^{(i)}\right]\right), \end{aligned} \tag{16}$$

where the $a_j^{(i)}$ elements are greater of equal than zero. We conclude by emphasizing the relation between the proposed re-parameterization and the relative degree with which each term appears. Notice that increasing $a_j^{(i)}$ we promote simultaneously all the monomials in which u_{k-j} appears with relative degree up to i. Moreover, to penalize monomials in which u_{k-j} appears with relative degree greater than i we need to set to zero the $a_j^{(l)}$ with $l > i$.

4. EXPERIMENTAL RESULTS

4.1 Simulated environment

In this set of experiments we test the performance of the MPK in a simulated environment, the benchmark system introduced in Spinelli et al. (2005), i.e. a third order Volterra series described by the following equation

$$\begin{aligned} z_k = u_k &+ 0.6u_{k-1} + 0.35(u_{k-2} + u_{k-4}) - 0.25u_{k-3}^2 \\ &+ 0.2(u_{k-5} + u_{k-6}) + 0.9u_{k-3} + 0.25u_k u_{k-1} + 0.75u_{k-2}^3 \\ &- u_{k-1}u_{k-2} + 0.5(u_k^2 + u_k u_{k-2} + u_{k-1}u_{k-3}). \end{aligned} \tag{17}$$

The MPK-based estimator is compared with the one based on the polynomial kernel reported in (11), hereafter denoted with PK; both for Pk and MPK we considered $r = 3$. Input signals are 1000 samples obtained from a realization of Gaussian noise. Concerning m_u^{tr}, m_u^{ts}, σ_u^{tr} and σ_u^{ts}, respectively, the input mean and standard deviation of the training and test samples, we consider four different scenarios:

- *Experiment 1*: $m_u^{tr} = m_u^{ts} = 0$, $\sigma_u^{tr} = \sigma_u^{ts} = 4$;
- *Experiment 2*: $m_u^{tr} = m_u^{ts} = 0$, $\sigma_u^{tr} = \sigma_u^{ts} = 2$;
- *Experiment 3*: $m_u^{tr} = -12$, $m_u^{ts} = 12$, $\sigma_u^{tr} = \sigma_u^{ts} = 4$;
- *Experiment 4*: $m_u^{tr} = -12$, $m_u^{ts} = 12$, $\sigma_u^{tr} = \sigma_u^{ts} = 2$.

In all the experiments the noise standard deviation is $\sigma_n = 4$. The PK and MPK hyperparameters have been trained optimizing the ML of the training samples. As concerns the optimization, we used standard gradient descent algorithm, with adaptive learning rate. The two estimators are implemented in PyTorch Paszke et al. (2017), to exploit automatic differentiation functionalities.

The four experiments can be grouped in two sets. Generalization properties are stressed more in *Experiment 3* and *Experiment 4* since $m_u^{tr} \neq m_u^{ts}$, and hence the training and test input signals are significantly different with each other; in particular, the mean values are such that with high probability the test inputs are outside the 3σ of training inputs distribution. In *Experiment 1* (resp. *Experiment 3*) and *Experiment 2* (resp. *Experiment 4*) we considered different values of the input standard deviation in order to analyze the estimators behaviors with different system excitations.

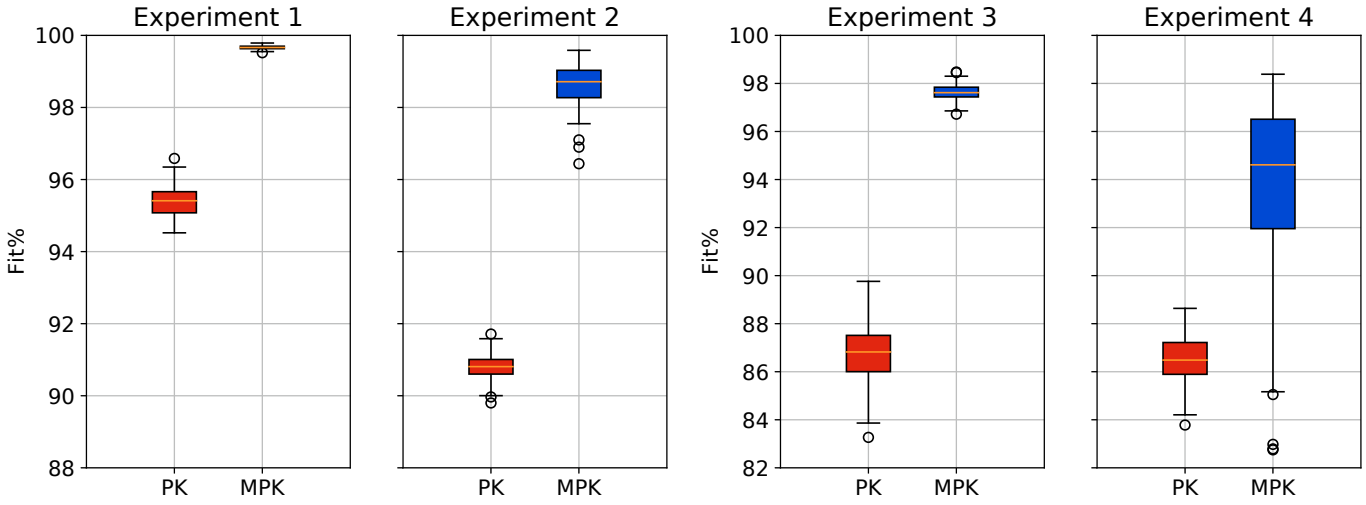

Fig. 1. Boxplots of the 100 test set Fit% obtained in the different scenarios considered.

For each experiment we performed a Monte Carlo of 100 simulations. In each simulation the same training and test data sets have been used to implement and test the MPK and PK based estimators. Results are reported in Figure 1. Performance is measured by the percentage fit (Fit%), defined as

$$100\% \left(1 - \frac{||\boldsymbol{z} - \hat{\boldsymbol{z}}||_1}{||\boldsymbol{z} - \bar{z}||_1}\right),$$

where $\boldsymbol{z}$ and $\hat{\boldsymbol{z}}$ are the vectors collecting the real and estimated system output, while $\bar{z}$ is the mean of $\boldsymbol{z}$.

In Figure 1, we visualized the results obtained through some boxplots. The estimator based on the MPK outperforms the standard polynomial kernel, since in all the tests the Fit% obtained with MPK are higher than the ones obtained with PK. Besides improving the estimation accuracy, the MPK parametrization improves also the generalization performance. Indeed, comparing results obtained in *Experiment 1* and *Experiment 3*, we can appreciate how the penalties learned by the MPK estimator by optimizing ML provides robustness to variations of the inputs distribution; more specifically, the MPK performance decreases less than the PK performance when the test input locations that are far from the training inputs.
As far as variations of the system excitation, comparing results obtained in *Experiment 3* and *Experiment 4*, we can observe how not sufficiently exciting training samples can lead to a bad identification of the MPK parameters. Notice how from *Experiment 3* to *Experiment 4*, the variance of MPK based estimator grows up more that the one of the PK based, highlighting the importance of using sufficiently exciting input trajectories.

4.2 Identification of the Silverbox system

In this subsection we test the estimators based on MPK and PK with data collected on a real system, the Silverbox Wigren and Schoukens (2013). The Silverbox is an electrical system that simulates a mass-spring-damper system. The spring exhibits nonlinear behaviors, and the system is described by the following differential equation

$$m\ddot{z}(t) + d\dot{z}(t) + k_1 z(t) + k_3 z^3(t) = u(t),$$

where z and u are, respectively, the mass displacement and the input force applied to the mass, while d, k_1, and k_3 are the parameters of the damper and the nonlinear spring.
We used the MPK and PK based estimators to learn the evolution of the mass displacement, modeling the unknown target function with a third order polynomial. We considered as input of the model the past $m = 5$ u and y, namely, at time k, the model input and output are, respectively, $[u_k \ \dots \ u_{k-m} \ z_{k-1} \ \dots \ z_{k-m}]$ and z_k. Notice that, due to the dependence on $z_{k-1} \dots z_{k-m}$, this input-output model is a polynomial NARMAX model. The original training and test dataset are publicly available [1]. The training dataset accounts approximately for 80000, obtained inputting to the system an odd random phase multisine signal, while the test set accounts for approximately 40000 samples, collected exciting the system with filtered Gaussian noise. To further stress generalization properties, we derived and trained the estimators using just the first 200 samples of the training dataset. Besides testing the one-step-ahead prediction performance, we measured also the simulation performance: given an initial state of the system we simulate its evolution evaluating iteratively the function learned for the one-step-ahead problem, inputting to the estimator the past predicted output instead of the measured output. In this context, we noticed that optimizing the hyperparameters through cross-validation increases the simulation performance. To deal with the considerable number of parameters, we adopted a gradient-based strategy. Specifically, we randomly select 5 partitions of the training data. Each partition accounts for two sets, composed of 100 samples. As loss function we considered the sum of the mean squared errors (MSEs) in validation. Namely, for each partition, we derive the estimator based on the first set, and we compute the MSE of the second set. The loss function is the sum of the MSEs. The kernel hyperparameters are updated minimizing the loss though a gradient-based procedure, iterated until convergence of the loss.
Performance is reported in Table 1. As before, we compare the one-step-ahead performance using the Fit%. Simulation performance are measured both with Fit% and root

[1] http://www.nonlinearbenchmark.org/

Table 1. One-step-ahead (Pred.) and simulation (Sim.) performance of the PK and MPK based estimators obtained in the Silverbox test dataset.

	Pred. (Fit%)	Sim. (Fit%)	Sim. (RMSE [mV])
PK	97.79	81.13	17.2213
MPK	99.70	98.67	0.8862

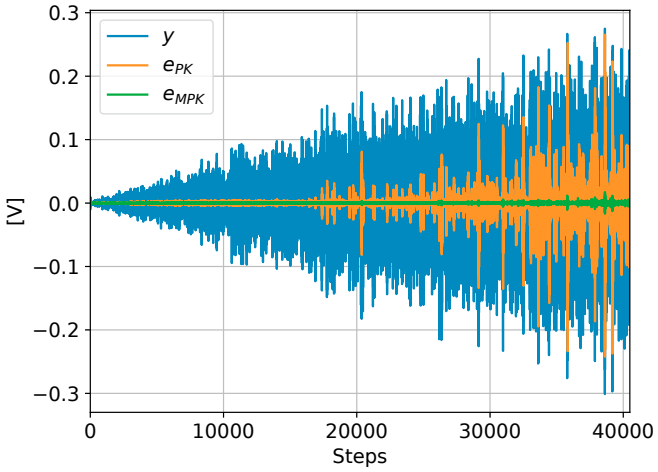

Fig. 2. Evolution of the test output together with the simulation errors e_{PK} and e_{MPK}.

mean squared error (RMSE). The MPK outperforms the standard polynomial kernel, both in one-step-ahead prediction and simulation. The gap is particularly evident in simulation, where MPK significantly outperforms PK; see Figure 2 to compare the evolution of the simulation errors. Despite we used just the first 200 training samples to derive the model, and hyperparameters optimization was focused on optimizing the one-step-ahead performance, the MPK performance is close to the best results obtained in this benchmark.

5. CONCLUSIONS

In this paper we have introduced the MPK. Compared to the standard polynomial kernel, the MPK is equipped with a set of parameters that allows to better select the monomials that really influence the system output. As proven by numerical results, this fact entails improvements in terms of accuracy and generalization.

REFERENCES

Billings, S., Chen, A., and Korenberg, M. (1989). Identification of MIMO non-linear systems using a forward-regression orthogonal algorithm. *Intern. J. of Control*, 49, 2157 – 2189.

Birpoutsoukis, G., Marconato, A., Lataire, J., and Schoukens, J. (2017). Regularized nonparametric volterra kernel estimation. *Automatica*, 82, 324 – 327.

Boyd, S. and Chua, L. (1985). Fading memory and the problem of approximating nonlinear operators with volterra series. *IEEE Transactions on Circuits and Systems*, 32(11), 1150–1161.

Chen, S., Billings, S.A., and Luo, W. (1989). Orthogonal least squares methods and their application to non-linear system identification. *International Journal of Control*, 50, 1873–1896.

Cheng, C., Peng, Z., Zhang, W., and Meng, G. (2017). Volterra-series-based nonlinear system modeling and its engineering applications: A state-of-the-art review. *Mechanical Systems and Signal Processing*, 87, 340 – 364.

Espinoza, M., Suykens, J.A.K., and De Moor, B. (2005). Kernel based partially linear models and nonlinear identification. *IEEE Trans. on Automatic Control*, 50(10), 1602–1606.

Franz, M. and Schölkopf, B. (2006). A unifying view of Wiener and volterra theory and polynomial kernel regression. *Neural Computation*, 18, 3097–3118.

Haber, R. and Unbehauen, H. (1990). Structure identification of nonlinear systems-a survey. *Automatica*, 26, 651–677.

Hall, J., Rasmussen, C., and Maciejowski, J. (2012). Modelling and control of nonlinear systems using Gaussian processes with partial model information. In *Proceedings of the 51st Annual Conference on Decision and Control (CDC)*.

Lind, I. and Ljung, L. (2008). Regressor and structure selection in NARX models using a structured ANOVA approach. *Automatica*, 44, 383–395.

Ljung, L. (1999). *System Identification - Theory for the User*. Prentice-Hall, Upper Saddle River, N.J., 2nd edition.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch.

Pillonetto, G. and De Nicolao, G. (2010). A new kernel-based approach for linear system identification. *Automatica*, 46(1), 81–93.

Poggio, T. and Girosi, F. (1990). Networks for approximation and learning. In *Proceedings of the IEEE*, volume 78, 1481–1497.

Rasmussen, C. and Williams, C. (2006). *Gaussian Processes for Machine Learning*. The MIT Press.

Rugh, W. (1980). *Nonlinear System Theory: The Volterra-Wiener Approach*. Johns Hopkins University Press.

Scholkopf, B. and Smola, A.J. (2001). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

Sjöberg, J., Zhang, Q., Ljung, L., A. Benveniste, B.D., Glorennec, P., Hjalmarsson, H., and Juditsky, A. (1995). Nonlinear black-box modeling in system identification: A unified overview. *Automatica*, 31(12), 1691–1724.

Spinelli, W., Piroddi, L., and Lovera, M. (2005). On the role of prefiltering in nonlinear system identification. *IEEE Transactions on Automatic Control*, 50(10), 1597–1602. doi:10.1109/TAC.2005.856655.

Stoddard, J.G., Welsh, J.S., and Hjalmarsson, H. (2017). Em-based hyperparameter optimization for regularized volterra kernel estimation. *IEEE Control Systems Letters*, 1(2), 388–393.

Tibshirani, R. (1996). Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society, Series B.*, 58, 267–288.

Wigren, T. and Schoukens, J. (2013). Three free data sets for development and benchmarking in nonlinear system identification. In *2013 European Control Conference (ECC)*, 2933–2938. doi:10.23919/ECC.2013.6669201.