2017 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, SEPT. 25–28, 2017, TOKYO, JAPAN

MACHINE LEARNING REGRESSION BASED ON PARTICLE BERNSTEIN POLYNOMIALS FOR NONLINEAR SYSTEM IDENTIFICATION

Giorgio Biagetti, Paolo Crippa, Laura Falaschetti, Claudio Turchetti

DII – Dipartimento di Ingegneria dell'Informazione
Università Politecnica delle Marche
Via Brecce Bianche 12, I-60131 Ancona, Italy
{g.biagetti, p.crippa, l.falaschetti, c.turchetti}@univpm.it

ABSTRACT

Polynomials have shown to be useful basis functions in the identification of nonlinear systems. However estimation of the unknown coefficients requires expensive algorithms, as for instance it occurs by applying an optimal least square approach. Bernstein polynomials have the property that the coefficients are the values of the function to be approximated at points in a fixed grid, thus avoiding a time-consuming training stage. This paper presents a novel machine learning approach to regression, based on new functions named particle-Bernstein polynomials, which is particularly suitable to solve multivariate regression problems. Several experimental results show the validity of the technique for the identification of nonlinear systems and the better performance achieved with respect to the standard techniques.

Index Terms— Machine learning, Bernstein polynomials, supervised learning, regression function, system identification.

1. INTRODUCTION

The goal on nonlinear system identification is to estimate a relation between inputs $e(t)$ and outputs $y(t)$ generated by an unknown dynamical system [1]. As this problem is equivalent to derive a model given a set of input-output data, in the language of machine learning it falls within the problem of supervised learning. Over the years a large variety of different approaches has been proposed in the literature to face this problem [2]. One of the most popular is the Lee-Schetzen method that identifies the Volterra kernels of nonlinear systems stimulated by random inputs with assigned statistic [3, 4]. Simplified Volterra-based models which combine a static nonlinearity and a linear dynamical system (Hammerstein-Wiener systems) have been profitably used to overcome calculation of multidimensional Volterra kernels [5–7]. Because of nonlinear signal processing and learning capability, artificial neural networks (ANN's) have become a powerful tool for nonlinear system identification [8, 9]. Recently machine learning techniques such as support vector machine (SVM) are progressing rapidly, and overcomes the neural networks' shortcomings, that is local minimizing and inadequacy to statistical problems [10, 11].

Machine learning techniques for nonlinear system identification reduce to solving a regression problem, thus polynomials play a central role in this context due to their property of approximating a function with arbitrary accuracy. Among a great variety of polynomials Bernstein polynomials [12] have the property that the coefficients of polynomials are given by the function to be approximated evaluated at points in a fixed grid. This useful property avoid the need of an algorithm to determine the unknown coefficients, as in other techniques occur.

The aim of this paper is to present a machine learning technique for regression of input-output relationships based on a set of new functions named particle-Bernstein polynomials, that is useful for nonlinear system identification. A wide experimentation on typical nonlinear time series examples known in the literature shows the benefit of the technique proposed.

2. THEORY

2.1. Supervised learning of non-linear systems

Consider a nonlinear discrete-time dynamic system

$$y(t) = h\left(y(t-1), \dots, y(t-p), e(t-1), \dots, e(t-q)\right), \quad t = 1, \dots, n \qquad (1)$$

where $e(t)$ and $y(t)$ denote input and output variables at time instant t and are considered known in the supervised learning. Assuming with no lack of generality the input, depends on some random parameters $x(p+1), \dots, x(d)$

$$e(t) = e\left(t, x(p+1), \dots, x(d)\right) \qquad (2)$$

then (1) becomes

$$y(t) = h\left(y(t-1), \ldots, y(t-p), x(p+1), \ldots, x(d), t\right), \quad t = 1, \ldots, n \quad . \tag{3}$$

Suppose the initial conditions are also random, then we can write

$$x(1) = y(0), \ldots, x(p) = y(1-p) \quad . \tag{4}$$

The solution $y(t),\ t = 1, \ldots, n$ of (3), which can be derived by solving iteratively (3), is a function of both the parameters and the initial conditions, so that (3) can be rewritten as

$$y = h'(x),\ y \in \mathbb{R}^n \tag{5}$$

where x is the vector $x = (x(1), \ldots, x(d))$ and $h'(\cdot)$ is a function that differs from $h(\cdot)$. Assuming x ranges in a set U, then we have

$$h'(x) :\ x \in U \subseteq \mathbb{R}^d \rightarrow y \in \mathbb{R}^n \quad . \tag{6}$$

In the supervised learning of time series data the goal is to derive a model given a set of input-output data

$$S = \{(y_j, x_j),\ j = 1, \ldots, N\} \quad . \tag{7}$$

As the models (3) and (5) give the same solution y, thus supervised learning of nonlinear time series given by (3) reduces to the regression of the input-output function (6) given the training data (7).

2.2. Bernstein Polynomials

Polynomials have shown to be useful basis functions to represent input-output relationships of the kind given by (6) [13, 14]. These techniques are founded on the well known Weierstrass approximation theorem which states that every continuous function $f\ :\ [0,1] \rightarrow \mathbb{R}$ can be approximated by a polynomial with arbitrary accuracy in the uniform norm. In the class of such basis functions, Bernstein polynomials are of great importance both for theoretical and practical aspects [12, 15–17].

The univariate Bernstein polynomials of degree m over the interval $[0,1]$ are defined by:

$$b_k^m(x) = \binom{m}{k} x^k (1-x)^{m-k}, \quad k = 0, 1, \ldots, m \tag{8}$$

where $x \in [0,1]$ and $\binom{m}{k} = \frac{m!}{(m-k)!\,k!}$. They form a complete basis for the space of polynomials of degree less than or equal to m, as they span the space of polynomials and they are linearly independent. These polynomials satisfy several useful properties such as symmetry $b_k^m(x) = b_{m-k}^m(1-x)$, positivity $b_k^m(x) \geq 0$ and they form a partition of unity $\sum_{k=0}^m b_k^m(x) = 1$. It is worth to notice that each term $x^k(1-x)^{m-k}$ is maximum at $x = k/m$, as it is illustrated in Fig. 1

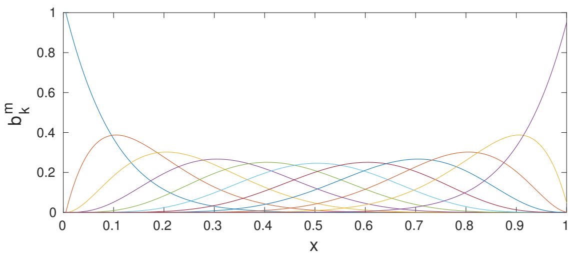


Fig. 1. The functions $b_k^m(x)$ for $m = 20$ and $k = 1, \ldots, m$.

for $m = 20$ and some values of k. A complete treatment of Bernstein polynomials is reported in [12] where a number of other properties can be found.

One of the main properties of Bernstein polynomials is that the coefficients of polynomials are given by the function f evaluated at points $\{k/m,\ k = 0, \ldots, m\}$. More precisely it can been shown that given a continuous function $f(x)$ in $[0,1]$ the sequence $B_m(x)$ defined as

$$B_m(x) = \sum_{k=0}^{m} f(k/m)\, b_k^m(x) \tag{9}$$

converges uniformly to f as $m \rightarrow \infty$. This relationship can be easily generalized in the case of multivariate functions giving

$$B_m(x_1, \ldots, x_d) = \sum_{k_1=0}^{m} \ldots \sum_{k_d=0}^{m} f\left(\frac{k_1}{m}, \ldots, \frac{k_d}{m}\right) \times b_k^m(x) \tag{10}$$

where $x = (x_1, \ldots, x_d), k = (k_1, \ldots, k_d)$ and

$$b_k^m(x) = \binom{m}{k_1} \ldots \binom{m}{k_d} x_1^{k_1}(1-x_1)^{m-k_1} \ldots x_d^{k_d}(1-x_d)^{m-k_d} \quad . \tag{11}$$

As you can see this representation only requires the knowledge of the function at the points

$$x = (k_1/m, \ldots, k_d/m),\ k_1 = 0, \ldots, m,\ k_d = 0, \ldots, m \tag{12}$$

without the need of an algorithm to determine the unknown coefficients, as in other techniques occurs. The set of points in (12) defines a grid in the space $\mathbb{R}^d$ of the input variable formed by $(m+1)^d$ points. The grid must be sufficiently dense to get a good approximation of f, thus requiring an increasing value of m for better accuracy. However as the dimensionality d of input space increases, the computational cost of (10) increases dramatically. In order to remove this limitation a different approach will be developed in the following.

2.3. Particle-Bernstein Polynomials

In Bernstein polynomials both the variables m and k are integer, as the binomial coefficients are defined for integer values alone. We can relax this constraint assuming k is real and denoting this value with ξ so that a set of new functions, called particle-Bernstein polynomials, can be defined as follows

$$C_\xi^m(x) = \alpha_\xi^m \, x^\xi \, (1-x)^{m-\xi} = \alpha_\xi^m \, k_\xi^m(x), \qquad \xi \in \mathbb{R}^1, \;\; \xi \in [0, m] \quad . \tag{13}$$

The coefficients α_ξ^m are chosen in such a way the integral constraint

$$\int_0^1 C_\xi^m(x)\, dx = 1, \tag{14}$$

holds. It can be shown that

$$\int_0^1 x^\xi \, (1-x)^{m-\xi} \, dx = \frac{\Gamma(\xi+1)\,\Gamma(-\xi+m+1)}{\Gamma(m+2)} \tag{15}$$

where $\Gamma(\cdot)$ is the gamma function, thus combining (14) and (15) yields

$$\alpha_\xi^m = \frac{\Gamma(m+2)}{\Gamma(\xi+1)\,\Gamma(-\xi+m+1)} \tag{16}$$

Fig. 2 shows several examples of functions defined by (13) for $m = 20$ and different values of ξ. As you can see all

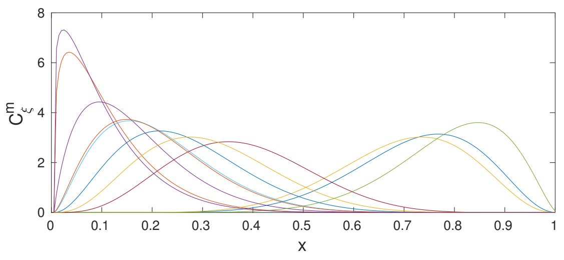


Fig. 2. The function $C_\xi^m(x)$ for $m = 20$ and several values of ξ.

the functions have the same area and attain their maximum at $x = \xi/m$. These are specific properties of such polynomials, that are not valid for other known polynomials, for instance Gegenbauer polynomials, since they don't exhibit a maximum alone in the interval $[0, 1]$. These properties can be used to approximate a function $f(x)$ around a given point. In fact supposing the function $C_\xi^m(x)$ is mostly concentrated around its maximum, and this is true for $m \gg 1$, the following approximation for $f \in \mathbb{R}^1$

$$f\,(\xi/m) \cong \int_0^1 f(x)\, C_\xi^m(x)\, dx \tag{17}$$

holds, where $f\,(\xi/m)$ is the value of $f(\cdot)$ at the maximum of $C_\xi^m(x)$. Thus the integral

$$f_m(\xi/m) = \int_0^1 f(x)\, C_\xi^m(x)\, dx \tag{18}$$

represents an approximating function of $f(x)$ around the point $x = \xi/m$. For a given set of values $\{x^{(j)},\ j = 1, 2, \ldots, N\}$ the integral in (18) can be approximated as

$$\int_0^1 f(x)\, C_\xi^m(x) dx \cong \frac{1}{N} \sum_{j=1}^{N} f(x^{(j)}) C_\xi^m(x^{(j)}) \tag{19}$$

Assuming x is a random variable, the convergence rate of approximation (9) can be shown to be $\mathcal{O}(N^{-1/2})$ [18]. Thus (18) reduces to

$$f(\xi/m) \cong \frac{1}{N} \sum_{j=1}^{N} f(x^{(j)}) C_\xi^m(x^{(j)}) \tag{20}$$

(20) can be used to estimate the function $f(x)$ at testing point ξ/m given the training set $\{f\left(x^{(j)}\right), j = 1, \ldots, N\}$. Thus the estimate of function at a given point does not require the knowledge of the function in a fixed grid, as in Bernstein polynomials occurs. Instead only the function at a set of N points randomly chosen is required.

2.4. Multivariate regression

The main benefit of previous approach is achieved in the regression of multivariate functions

$$f(x),\ x \in U \subseteq \mathbb{R}^d \tag{21}$$

since in this case the integral in (17) reduces to a simple summation of N terms. In this case (13) becomes

$$C_\xi^m(x) = \alpha_\xi^m \, x_1^{\xi_1}(1-x_1)^{m-\xi_1} \ldots x_d^{\xi_d}(1-x_d)^{m-\xi_d} = \alpha_\xi^m \, k_\xi^m(x) \tag{22}$$

with $\xi = (\xi_1, \ldots, \xi_d)$, $x = (x_1, \ldots, x_d)$ and

$$\alpha_\xi^m = \prod_{t=1}^{d} \frac{\Gamma(m+2)}{\Gamma(\xi_t+1)\,\Gamma(-\xi_t+m+1)} \tag{23}$$

$$k_\xi^m = \prod_{t=1}^{d} x_t^{\xi_t} (1-x_t)^{m-\xi_t} \quad . \tag{24}$$

Having defined the multivariate particle-Bernstein polynomials the estimate of function f at points ξ/m is given by the same expression (20).

3. EXPERIMENTAL RESULTS

The experiments were conducted on systems described by (3), where $x \in [0,1]^d$ are random variables with a given probability distribution $p(x)$ and y is an $n-$vector. To determine the performance of the proposed approach four different techniques were used: *a)* Bernstein polynomials (BP), *b)* particle-Bernstein polynomials (PBP), *c)* Optimal Bernstein polynomials (OBP), *d)* Support Vector Machine (SVM). Bernstein polynomials and particle-Bernstein polynomials refers to representation (10) and (20) respectively. In optimal Bernstein polynomials (10) reduces to

$$B_m(x_1,\ldots,x_d)=\sum_{k_1=0}^{m}\cdots\sum_{k_d=0}^{m} a_{k_1}\ldots a_{k_d}\ \times b_k^m(x) \quad (25)$$

where the coefficients $a_{k_1},\ldots,a_{k_d}$ are estimated by a least-squares optimization algorithm, following the approach given in [19]. SVM uses a kernel SVM regression algorithm.

3.1. Experiment 1

The first experiment was conducted on the nonlinear system proposed by Narendra *et al.* in [8] whose equation is

$$\begin{aligned} y(t) &= 0.3\,y(t-1)+0.6\,y(t-2)+f[e(t-1)] \\ f(e) &= 0.6\sin(\pi\, e)+0.3\sin(2\pi e)+0.1\sin(5\pi e) \\ & \qquad t=1,\ldots,n \end{aligned} \quad (26)$$

where $e(t)$ is the input signal depending on the x-vector. We considered three cases for the input $e(t)$, each characterized by a different dimension of x.

Case 1) In this case the input only depends on the parameter $x(1)$

$$e(t)=\sin[(1+x(1))\omega_0 t] \quad (27)$$

where $\omega_0 = 2\pi/100$, $x(1)$ is an r.v.uniformly distributed in $[0,1]$ and the initial conditions are $y(0)=y(1)=1$.

Case 2) The input depends on two parameters $x(1)$, $x(2)$

$$e(t)=x(2)*\sin[(1+x(1))\omega_0 t] \quad (28)$$

where $x(1), x(2)$ are r.v.uniformly distributed in $[0,1]^2$, while the other parameters are the same of case 1.

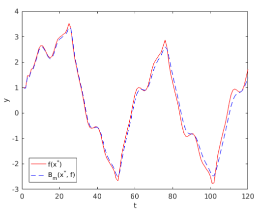
Case 3) In this case the input is given by

$$e(t)=x(2)*\sin[(1+x(1))\omega_0 t+x(3)] \quad (29)$$

where $x(1)$, $x(2)$, $x(3)$ are r.v.uniformly distributed in $[0,1]^3$, and the other parameters are the same of previous cases.

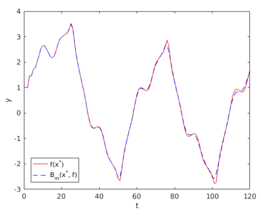
In general the solution of (26) is a function $f(x)$ with $x = x(1)$, $x=(x(1),x(2))$, $x=(x(1),x(2),x(3))$ for the case 1, case 2 and case 3 respectively. Thus the identification of the system with these three inputs reduces to the multivariate regression of 1D, 2D and 3D functions.

Fig. 3 to Fig. 5 report the experimental results for the three cases and the four regression techniques, compared with the data generated by the time series (26). The train and the test time was recorded using the stopwatch timer of MATLAB. The order of polynomials is $m = 40$ for both BP, PBP and OBP, while a number of $N = 10^4$ of training points has been chosen for PBP, OBP and SVM. In the case 3 the training points used for OBP were reduced to $N = 10^3$ since convergence could not be reached for a greater value. As you can see all the techniques model data with a good accuracy. Instead execution time for training and testing stages are very different depending on the approach used. In particular both OBP and SVM require a longer execution time for training than time required by the techniques suggested in this paper.



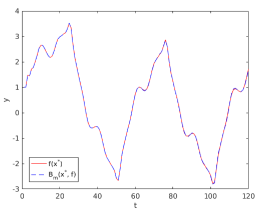
(a) Particle-Bernstein polynomials

$t_{\text{train}} = 2.052\ s;\ t_{\text{test}} = 2.592\ s$



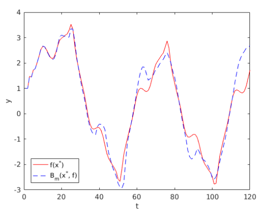
(b) Bernstein polynomials

$t_{\text{train}} = 0.019\ s;\ t_{\text{test}} = 0.046\ s$



(c) Optimal Bernstein polynomials

$t_{\text{train}} = 6.993\cdot 10^2\ s;\ t_{\text{test}} = 0.086\ s$



(d) SVM

$t_{\text{train}} = 2.301\cdot 10^3\ s;\ t_{\text{test}} = 0.074\ s$

Fig. 3. Comparison of results obtained by nonlinear system (26) and input (27) (continuous line) with those obtained by four different models (broken line). t_{train} and t_{test} are the execution times required for training and testing stages respectively.

3.2. Experiment 2

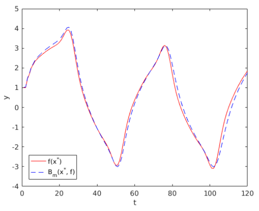
In the second experiment the nonlinear system is governed by the difference equation proposed by Narendra *et al.* in [8]

$$y(t)=\frac{y(t-1)y(t-2)[y(t-1)-2.5]}{1+y^2(t-1)+y^2(t-2)}+e(t-1) \qquad t=1,\ldots,n \quad (30)$$

where $e(t)$ is the input signal. Also in this experiment three cases for the input $e(t)$, with different dimensions of x were considered.

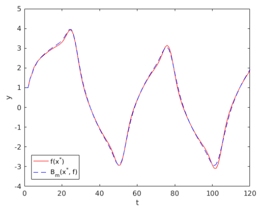
Fig. 6 to Fig. 8 report the experimental results for the three cases and the four regression techniques, compared with the data generated by the time series (30). The other experimental conditions are the same as those adopted for Experiment 1.

Finally Tab. 1 reports for the four different identification techniques, the execution time required for both the training and testing stages. As you can see the performance obtained with Bernstein and particle-Bernstein polynomials are significantly better than those obtained with the other techniques.



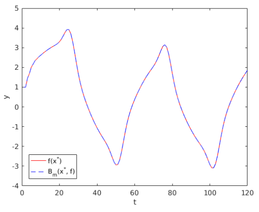
(a) Particle-Bernstein polynomials

$t_{\text{train}} = 2.052\ s;\ \ t_{\text{test}} = 5.196\ s$



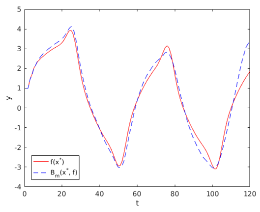
(b) Bernstein polynomials

$t_{\text{train}} = 0.366\ s;\ \ t_{\text{test}} = 1.751\ s$



(c) Optimal Bernstein polynomials

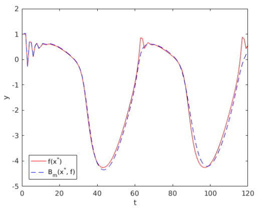
$t_{\text{train}} = 3.246 \cdot 10^3\ s;\ \ t_{\text{test}} = 0.168\ s$



(d) SVM

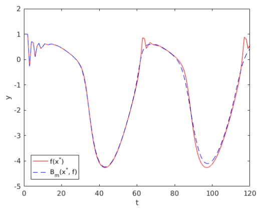
$t_{\text{train}} = 5.254 \cdot 10^3\ s;\ \ t_{\text{test}} = 0.080\ s$

Fig. 4. Comparison of results obtained by nonlinear system (26) and input (28) (continuous line) with those obtained by four different models (broken line). t_{train} and t_{test} are the execution times required for training and testing stages respectively.



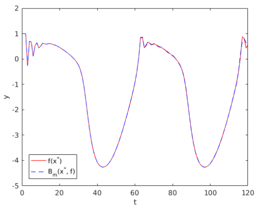
(a) Particle-Bernstein polynomials

$t_{\text{train}} = 1.531\ s;\ \ t_{\text{test}} = 2.748\ s$



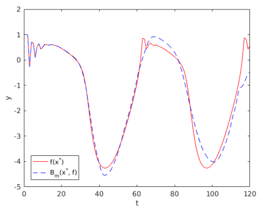
(b) Bernstein polynomials

$t_{\text{train}} = 0.025\ s;\ \ t_{\text{test}} = 0.066\ s$



(c) Optimal Bernstein polynomials

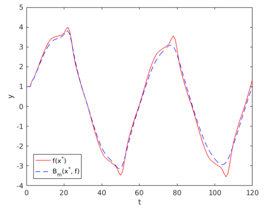
$t_{\text{train}} = 6.292 \cdot 10^2\ s;\ \ t_{\text{test}} = 0.070\ s$



(d) SVM

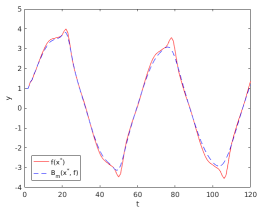
$t_{\text{train}} = 1.662 \cdot 10^3\ s;\ \ t_{\text{test}} = 0.071\ s$

Fig. 6. Comparison of results obtained by nonlinear system (30) and input (27) (continuous line) with those obtained by four different models (broken line). t_{train} and t_{test} are the execution times required for training and testing stages respectively.



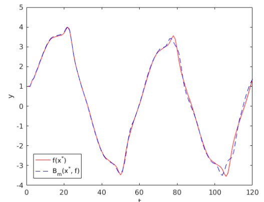
(a) Particle-Bernstein polynomials

$t_{\text{train}} = 2.160\ s;\ \ t_{\text{test}} = 7.884\ s$



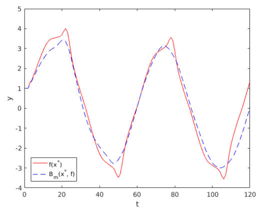
(b) Bernstein polynomials

$t_{\text{train}} = 14.217\ s;\ \ t_{\text{test}} = 1.066 \cdot 10^2\ s$



(c) Optimal Bernstein polynomials

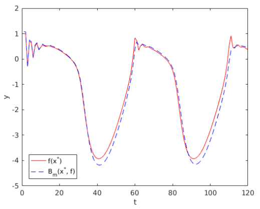
$t_{\text{train}} = 3.656 \cdot 10^3\ s;\ \ t_{\text{test}} = 0.192\ s$



(d) SVM

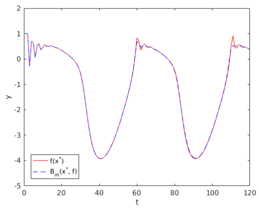
$t_{\text{train}} = 8.827 \cdot 10^3\ s;\ \ t_{\text{test}} = 0.084\ s$

Fig. 5. Comparison of results obtained by nonlinear system (26) and input (29) (continuous line) with those obtained by four different models (broken line). t_{train} and t_{test} are the execution times required for training and testing stages respectively.



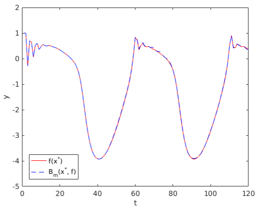
(a) Particle-Bernstein polynomials

$t_{\text{train}} = 1.511\ s;\ \ t_{\text{test}} = 5.112\ s$



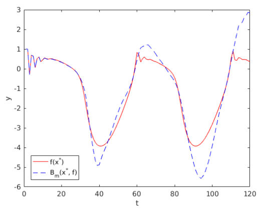
(b) Bernstein polynomials

$t_{\text{train}} = 0.270\ s;\ \ t_{\text{test}} = 1.730\ s$



(c) Optimal Bernstein polynomials

$t_{\text{train}} = 3.493 \cdot 10^3\ s;\ \ t_{\text{test}} = 0.132\ s$



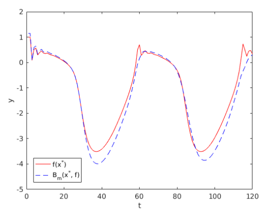
(d) SVM

$t_{\text{train}} = 6.512 \cdot 10^3\ s;\ \ t_{\text{test}} = 0.086\ s$

Fig. 7. Comparison of results obtained by nonlinear system (30) and input (28) (continuous line) with those obtained by four different models (broken line). t_{train} and t_{test} are the execution times required for training and testing stages respectively.

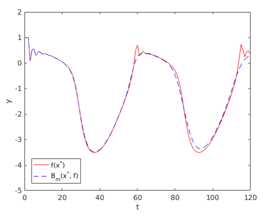
4. CONCLUSION

The main drawback of using polynomials as basis functions in the identification of nonlinear systems, is related to the estimation of unknown coefficients that requires a time-consuming training stage. This paper shows that Bernstein polynomials overcome this problem as they only require the knowledge of the function to be approximated at points in a fixed grid. In addition a set of new functions named particle-Bernstein polynomials have shown to be particularly useful in nonlinear system identification, relaxing the constraint imposed by the fixed grid.



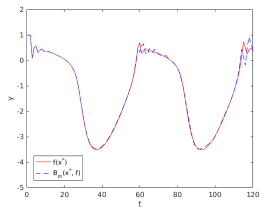
(a) Particle-Bernstein polynomials

$t_{\text{train}} = 1.624\ s;\ t_{\text{test}} = 7.716\ s$



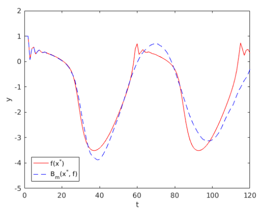
(b) Bernstein polynomials

$t_{\text{train}} = 10.255\ s;\ t_{\text{test}} = 1.066 \cdot 10^2\ s$



(c) Optimal Bernstein polynomials

$t_{\text{train}} = 3.940 \cdot 10^3\ s;\ t_{\text{test}} = 0.192\ s$



(d) SVM

$t_{\text{train}} = 1.379 \cdot 10^4\ s;\ t_{\text{test}} = 0.086\ s$

Fig. 8. Comparison of results obtained by nonlinear system (30) and input (29) (continuous line) with those obtained by four different models (broken line). t_{train} and t_{test} are the execution times required for training and testing stages respectively.

Table 1. Execution time for training and testing stages; $d = 2, N = 10^4, n = 40$.

Order	**Particle-Bernstein**		**Bernstein**		**Optimal Bernstein**		**SVM**	
	Train [s]	Test [s]	Train [s]	Test [s]	Train [s]	Test [s]	Train [s]	Test [s]
10	0.6533	1.5182	0.0186	0.1107	118.212	0.0166	3640.040	0.0254
20	0.6378	1.5970	0.0421	0.2318	224.028	0.0254	3649.600	0.0244
30	0.6275	1.6177	0.1064	0.4036	454.452	0.0350	3653.996	0.0258
40	0.6238	1.6271	0.1619	0.6789	929.976	0.0680	3660.984	0.0241
50	0.6529	1.6498	0.2381	1.0107	1464.912	0.0600	3634.928	0.0249

5. REFERENCES

[1] S. A. Billings, "Identification of nonlinear systems-a survey," *IEE Proc-D*, vol. 127, no. 6, pp. 272–285, Nov 1980.

[2] G. B. Giannakis and E. Serpedin, "A bibliography on nonlinear system identification," *Signal Process.*, vol. 81, no. 3, pp. 533–580, 2001.

[3] M. Schetzen, "Nonlinear system modeling based on the Wiener theory," *Proc. IEEE*, vol. 69, no. 12, pp. 1557–1573, Dec 1981.

[4] M. Inagaki and H. Mochizuki, "Bilinear system identification by Volterra kernels estimation," *IEEE Trans. Autom. Control*, vol. 29, no. 8, pp. 746–749, Aug 1984.

[5] A. Novak *et al.*, "Nonparametric identification of nonlinear systems in series," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 8, pp. 2044–2051, Aug 2014.

[6] C. Yu *et al.*, "A new deterministic identification approach to Hammerstein systems," *IEEE Trans. Signal Process.*, vol. 62, no. 1, pp. 131–140, Jan 2014.

[7] C. Turchetti *et al.*, "Nonlinear system identification: an effective framework based on the Karhunen–Loève transform," *IEEE Trans. Signal Process.*, vol. 57, no. 2, pp. 536–550, 2009.

[8] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Netw.*, vol. 1, no. 1, pp. 4–27, 1990.

[9] S. Lu and T. Basar, "Robust nonlinear system identification using neural-network models," *IEEE Trans. Neural Netw.*, vol. 9, no. 3, pp. 407–429, 1998.

[10] M. Espinoza *et al.*, "Kernel based partially linear models and nonlinear identification," *IEEE Trans. Autom. Control*, vol. 50, no. 10, pp. 1602–1606, Oct 2005.

[11] J. Xie, "Time series prediction based on recurrent LS-SVM with mixed kernel," in *2009 Asia-Pacific Conference on Information Processing*, July 2009, vol. 1, pp. 113–116.

[12] R. T. Farouki, "The Bernstein polynomial basis: A centennial retrospective," *Comput. Aided Geom. Des.*, vol. 29, no. 6, pp. 379–419, 2012.

[13] A. Carini and G.L. Sicuranza, "Fourier nonlinear filters," *Signal Process.*, vol. 94, pp. 183–194, 2014.

[14] A. Carini *et al.*, "Legendre nonlinear filters," *Signal Process.*, vol. 109, pp. 84 – 94, 2015.

[15] J. Berchtold and A. Bowyer, "Robust arithmetic for multivariate bernstein-form polynomials," *Comput. Aided Des.*, vol. 32, no. 11, pp. 681 – 689, 2000.

[16] S. Riachy *et al.*, "Multivariate numerical differentiation," *J. Comput. Appl. Math.*, vol. 236, no. 6, pp. 1069 – 1089, 2011.

[17] M. Mboup *et al.*, "Numerical differentiation with annihilators in noisy environment," *Numerical Algorithms*, vol. 50, no. 4, pp. 439–467, Apr 2009.

[18] H. Niederreiter, "Quasi-Monte Carlo methods and pseudo-random numbers," *Bull. Amer. Math. Soc.*, vol. 84, no. 6, pp. 957–1041, 11 1978.

[19] A. Marco *et al.*, "Accurate polynomial interpolation by using the Bernstein basis," *Numerical Algorithms*, pp. 1–20, 2016.