Improved Piecewise Linear Approximation of Nonlinear Functions in Hybrid Control

S. Kozak, ∗ J. Stevek ∗∗

∗ Slovak University of Technology, Faculty of Electrical Engineering and Information Technology, Bratislava, Slovakia (e-mail: stefan.kozak@stuba.sk).

∗∗ Slovak University of Technology, Faculty of Informatics and Information Technologies, Bratislava, Slovakia (e-mail: stevek@fiit.stuba.sk)

Abstract: The paper addresses the issue of the PWA approximation of nonlinear functions in hybrid modeling. We propose an original approach to PWA approximation of common nonlinear functions of one and more variables by orthogonal activation function based neural network (OAF NN). Complexity of the PWA system is the basic limitation in model predictive control design. We present a universal linearization method for generating PWA model of a common nonlinear system from process data. The presented method gives good results in precision in proportion to the number of linearization points. The method allows linearization of a nonlinear dynamical system in few number of linearization points for the purpose of hybrid modeling and model predictive control. The proposed method was successfully verified on many case studies for SISO and MIMO systems.

Keywords: Piecewise linear analysis, Parameter identification, Orthogonal polynomial, Chebyshev polynomial, Nonlinear equations, PWA systems, Generalized Fourier series.

1. INTRODUCTION

Methods of hybrid modeling and model predictive control of hybrid systems were presented in many publications (Bemporad et al., 2002a; Goodwin et al., 2005; Kozak and Stevek, 2010). In (Borrelli et al., 2006; Bemporad et al., 2003; Corona et al., 2006) authors addressed the problem of piecewise linear approximation of nonlinear function of one or two variables. Using piecewise linear approximation it is possible to insert a nonlinear object into a hybrid model structure. A piecewise affine function (PWA) is suitable for approximating a nonlinear functions in a wide range of operating points. Linearization of a one-dimensional function is easy to hand by PWA. For functions of a higher dimension the situation is more complicated. The number of linearization points plays a key role in model complexity. High complexity of PWA system prohibits model predictive control design (Kvasnica (2009)).

In (Ferrari-Trecate et al., 2003; Nakada et al., 2005) identification methods based on clustering algorithms were presented. One dimensional problems are approximated with lines and higher dimensional problems by hyperplanes. Kvasnica (2010) proposed an approach to PWA approximation of nonlinear function of two variables. This approach uses simple transformation of a nonlinear formula into a quadratic form of one variable. The quadratic function of one variable is then approximated by several lines. The drawback of this approach is necessity of a priori knowledge of algebraic nonlinear formula and requirement of algebraic modification. Also this approach is not suitable for functions of higher dimensions.

We propose a more universal method to PWA approximation and extend the approach for common nonlinear systems. We use methodology of generalized Fourier series with orthogonal polynomials. In (Leondes, 1997) orthogonal polynomials were used as activation functions for special case of neural network with one hidden layer - Orthogonal Activation Function based Neural Network (OAF NN). For this type of neural network online and offline training algorithm has been defined with fast convergence properties. After simple modification of OAF NN it is possible to use this technique for PWA approximation of a common nonlinear system.

The paper is divided in five sections. First, we formulate the identification and linearization problem in Model Predictive Control (MPC) of hybrid systems. Section 3 presents theoretic background for modeling of nonlinear process by OAF NN and the linearization concept (PWA OAF NN).

Case studies of modeling are shown in Section 4. The PWA OAF NN linearization of one-dimensional function is compared with traditional PWA linearization using HIT toolbox (Ferrari-Trecate, 2005). Linearization of a 3-D function is presented in Subsection 4.2. And linearization of benchmark nonlinear dynamical system is presented in Subsection 4.3.

2. PROBLEM FORMULATION

The main problem in real time implementation of explicit MPC (Bemporad et al., 2002b) is a reasonable computation time of PWA representation of control law (offline complexity) and the size of calculated PWA func-
The principal advantage of explicit MPC is off-line computation. To perform computation of PWA function in a reasonable time requires a simple system model with satisfactory accuracy. Small model definition decreases off-line and space complexity of MPC problem. This work is focused on reducing number of linearization points in PWA approximation of nonlinear system while maintaining required precision.

A clustering-based algorithms were employed in traditional identification of PWA maps. The driving idea of clustering-based procedures is that PWA map is locally linear. Then, if the local models around two data points are similar, it is likely that the data points belong to the same mode. Clustering-based algorithm consists of four basic steps (Ferrari-Trecate, 2005):

a) Associate to each data point a local affine model;
b) Aggregate local models with similar features into clusters;
c) Classify in the same way data points corresponding to local models in the same cluster;
d) Estimate parameter vectors and regions.

This approach is suitable for two and more dimensional problem too. Disadvantages are complexity and heuristic character of procedure. Example of PWA map for one-dimensional case is presented in Fig. 1.

In Kvasnica (2010) an approach to PWA approximation of nonlinear functions of two variables was proposed. This approach uses simple transformation of nonlinear formula \( f = x_1 x_2 \) to the form \( f = \frac{1}{2}(u_1^2 - u_2^2) \), where \( u_1 = (x_1 + x_2) \), \( u_2 = (x_1 - x_2) \). A quadratic form of one variable is then approximated by several lines (Fig. 2). Drawback of this approach is necessity of a priori knowledge of algebraic nonlinear formula and requirement of algebraic modification. Also this approach is not suitable for functions of three or more variables.

Improvement of this approach provides generalized Fourier series. Generalized Fourier series is based on a set of one-dimensional orthonormal functions \( \phi_i^{(N)} \) defined as

\[
\int_{x_1}^{x_2} \phi_i^{(N)}(x) \phi_j^{(N)}(x) = \delta_{ij}
\]

where \( \delta_{ij} \) is the Kronecker delta function and \([x_1, x_2]\) is the domain of interest. Several examples of orthonormal functions are the normalized Fourier (harmonic) functions, Legendre polynomials, Chebyshev polynomials and Laguerre polynomials (Leondes, 1997). In this paper only Chebyshev polynomials will be discussed.

OAF NN is employed in the task of nonlinear approximation. PWA approximation of every used orthonormal polynomial creates Piecewise Affine Orthogonal Activation Function based Neural Network (PWA OAF NN).

3. ORTHOGONAL ACTIVATION FUNCTION BASED NEURAL NETWORK

3.1 Structure of OAF NN

The orthogonal activation function based neural network is a three-layer neural structure with an input layer, an output layer and a hidden layer as shown in Fig. 3 for a multi-input-single-output (MISO) system.

The hidden layer consists of neurons with orthogonal (preferably orthonormal) activation functions. The activation functions for these neurons belong to the same class of orthogonal functions and no two neurons have the same order of activation function. The input and output layers consist of linear neurons. The weights between the input and the hidden layer are fixed, and depend on the orthogonal activation functions. The nodes on the right of the orthogonal neurons implement the product (\(\Sigma\)). Each node has \(m\) input signals from different input blocks.
Fig. 3. Orthogonal Activation Function Based Neural Network structure

are considered as part of the hidden layer because there is no weighting operation between orthogonal neurons and nodes (Fig. 3).

The network output is defined by a linear combination of weights

\[ \hat{y}(\bar{x}, \hat{w}) = \sum_{\bar{n}=0}^{N_{m}-1} \sum_{m_{m}=0}^{N_{m}-1} \bar{w}_{n_{1} \ldots n_{m}} \phi_{n_{1} \ldots n_{m}}(\bar{x}) = \Phi^{T}(\bar{x})\hat{w}, \quad (2) \]

where \( \bar{x} = [x_{1}, x_{2}, \ldots, x_{m}] \) is m-dimensional input vector, \( N_{i} \) is the neuron number of i-th input and \( \hat{w} \) between the hidden and output layers. Functions \( \phi_{n_{1} \ldots n_{m}}(\bar{x}) \) are orthogonal functions in m-dimensional space given by

\[ \phi_{n_{1} \ldots n_{m}}(\bar{x}) = \prod_{i=1}^{m} \phi_{n_{i}}(x_{i}), \quad (3) \]

where \( \phi_{i}(\bar{x}) \) are one-dimensional orthogonal functions implemented by each hidden layer neuron. For linear modeling it is necessary to remove connections into \( \prod \) which increase dimension of the function (3). Then we get generalized Fourier series of the optional nonlinear function (Fig. 4).

### 3.2 Orthogonal polynomials

A system of one-dimensional functions \( \phi_{i}(x) \) is orthogonal/orthonormal in \( < x_{1}, x_{2} > \) if

\[ \int_{x_{1}}^{x_{2}} \phi_{i}(x)\phi_{j}(x) = \begin{cases} 0 & \text{if } i = j \\ K_{i}^{2} & \text{if } i \neq j \end{cases} \quad (4) \]

where

\[ K_{i} = \sqrt{\int_{x_{1}}^{x_{2}} (\phi_{i}(x))^{2} dx} \quad (5) \]

is a norm of the orthogonal function. In case of orthonormal functions \( K_{i} = 1 \). The orthonormal system \( \phi_{\text{norm}}(x) \) is obtained from the orthogonal one \( \phi_{i}(x) \) by dividing by the norm \( \phi_{\text{norm}}(x) = \phi_{i}(x)/K_{i} \). In the examples we have used Chebyshev polynomials which are orthogonal in \( < -1, 1 > \) and the weighting function is \( 1/\sqrt{1-x^2} \). The Chebyshev polynomials of the first kind can be defined by the trigonometric identity

\[ T_{n}(x) = \cos(n \arccos(x)) \quad (6) \]

with norm defined as follows

\[ \frac{1}{\sqrt{1-x^2}} (T_{n}(x))^{2} dx = \begin{cases} \pi & n = 0 \\ \pi/2 & n \neq 1 \end{cases} \quad (7) \]

Recursive generating formula for Chebyshev polynomials:

\[ T_{0}(x) = 1, \]

\[ T_{1}(x) = x, \quad (8) \]

\[ T_{n+1}(x) = 2xT_{n}(x) - T_{n-1}(x), \quad (9) \]

\[ T_{n}(x) = U_{n+1}(x) - U_{n-1}(x). \quad (10) \]

where \( U_{n} \) is the Chebyshev polynomial of the second kind generated by the recursive formula:

\[ U_{0}(x) = 1, \quad (12) \]

\[ U_{1}(x) = 2x, \quad (13) \]

\[ U_{n+1}(x) = 2xU_{n}(x) - U_{n-1}(x). \quad (14) \]

The first few Chebyshev polynomials of the first kind are

\[ T_{0}(x) = 1, \quad (15) \]

\[ T_{1}(x) = x, \quad (16) \]

\[ T_{2}(x) = 2x^2 - 1 \quad (17) \]

\[ T_{3}(x) = 4x^3 - 3x \quad (18) \]

\[ T_{4}(x) = 8x^4 - 8x^2 + 1. \quad (19) \]
3.3 Least-squares learning (LSQ)

For OAF NN parameter identification least-squares estimator of Kariya and Kurata (2004) is used. If system is observed during periods then we can formulate following matrix equation

\[ Y_N = \Phi_N \theta^* + \Xi_N \]

where

\[ Y_N = [y(1), y(2), \ldots, y(N)]^T, \]
\[ \Phi_N = [\phi^T(1), \phi^T(2), \ldots, \phi^T(N)]^T, \]
\[ \Xi_N = [\xi(1), \xi(2), \ldots, \xi(N)]^T. \]

For particular vector of parameters \( W \) the output prediction the k-th step is

\[ \hat{y}(k, W) = \phi^T(k)W \]

The prediction error is defined as

\[ \epsilon(k, W) = y_p(k) - \hat{y}_m(k, W) \]

Then we define the performance index

\[ J_N = 1/2 \sum_{i=1}^{N} \epsilon^2(k, W) \]

3.4 PWA approximation of Chebyshev polynomials

A proper piecewise linear approximation plays a key role in effective modeling. T2, T3, T4, T5 Chebyshev polynomials with linearization in one point are shown in Fig. 5. A convenient feature of all Chebyshev polynomial is their symmetry. All polynomials of even order are symmetrical by vertical axis and all polynomial of odd order are symmetrical by origin. These properties allow decreasing number of linearization points to half while keeping precision. To get the lowest number of shift cases of generated PWA model we linearized the polynomials in the same points. The term 'linearization point' denotes the interval division point where the PWA function breaks.

4. EXAMPLES

4.1 PWA approximation of 1-D function

Example of a 1-D nonlinear function is defined as

\[ f(x) = |x|^{\sin(x)}, x \in [-1, 1] \]

We have made sample data with added white noise (Fig. 6). Straightforward approach to linearization of 1-D function is piecewise linear approximation (Fig. 6A) in three points, it means we get four shifting cases. Mean square error for this approach is MSE = 0.0136. Using PWA OAF NN approach we obtain a more precise approximation with the same number of points (Fig. 6B). Mean square error in this case was MSE = 0.0061. Before parameter estimation it was necessary to normalize data into the interval \( < -1, 1 > \) where Chebyshev polynomials are orthogonal. We used the first six Chebyshev polynomials \( T_0 \div T_5 \).

Table 1. OAF NN parameters for 1-D function

<table>
<thead>
<tr>
<th>W1</th>
<th>W2</th>
<th>W3</th>
<th>W4</th>
<th>W5</th>
<th>W6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0347</td>
<td>-0.2544</td>
<td>-0.3154</td>
<td>0.0955</td>
<td>0.2043</td>
<td>-0.0976</td>
</tr>
</tbody>
</table>
4.2 PWA approximation of a 3-D function

Consider a 3-D nonlinear function defined as

\[
f(x_1, x_2) = -0.2(\sin(x_1 + 4x_2)) - 2\cos(2x_1 + 3x_2) \nonumber \]
\[
-3\sin(2x_1 - x_2) + 4\cos(x_1 - 2x_2) \nonumber \]
\[
x_1 \in <0, 1>, \nonumber \]
\[
x_2 \in <0, 1>. \nonumber \]

(32)

We used the first six Chebyshev polynomials, up to the fifth order \(T_0 \div T_5\), linearized in 1 point, each polynomial by two lines (Fig. 5). The total number of shifting cases for the resulting PWA function is \(n_{lp} + 1\) where \(n_u\) is the number of neural network inputs and \(lp\) is the number of linearization points. For the 3-D function example (32) we get \(2^2 = 4\) shifting cases. The result are plotted in Fig. 8. For this approximation \(\text{MSE}=0.0144\).

![3-D function example](image1)

**Fig. 7. 3-D function example**

![PWA approximation of 3-D function](image2)

**Fig. 8. PWA approximation of a 3-D function**

### Table 2. OAF NN parameters for 3-D function

<table>
<thead>
<tr>
<th>(W_1)</th>
<th>(W_2)</th>
<th>(W_3)</th>
<th>(W_4)</th>
<th>(W_5)</th>
<th>(W_6)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-6.6854</td>
<td>1.6247</td>
<td>0.4006</td>
<td>0.2972</td>
<td>0.8422</td>
<td>-0.2714</td>
</tr>
<tr>
<td>W7</td>
<td>W8</td>
<td>W9</td>
<td>W10</td>
<td>W11</td>
<td></td>
</tr>
<tr>
<td>1.6247</td>
<td>0.4006</td>
<td>0.2972</td>
<td>0.8422</td>
<td>-0.2714</td>
<td></td>
</tr>
</tbody>
</table>

### 4.3 Modeling of benchmark nonlinear dynamic system

To evaluate the performance of the PWA OAF NN identifier, a simulation study was conducted by applying the identifier to a benchmark nonlinear system model (Leondes, 1997) defined by the nonlinear difference equation

\[
y(k + 1) = \frac{0.6y(k)}{1 + y(k) + 0.8y^2(k)} + 0.1u^3(k). \quad (33)\]

The input to the system was a harmonic signal \(u(t) = \sin(0.5t) - \cos(0.25t)\) which varies within the bounds \([-2, 2]\). Fig. 9 shows the system signals \(u(k)\) and \(y(k)\) and the response of PWA OAF NN identifier after the training. The network has 2 input nodes \(u(k-1)\) and \(y(k-1)\) and one output node \(y(k)\). There are eleven neurons in the hidden layer. The input training data were uniformly sampled over a training cycle of 25 s with a sampling

![Time response of the system (33) and the PWA OAF NN identifier after training](image3)

**Fig. 9. Time response of the system (33) and the PWA OAF NN identifier after training**

![Time response of the system (33) and the PWA OAF NN for unknown input signal](image4)

**Fig. 10. Time response of the system (33) and the PWA OAF NN for unknown input signal**
Orthogonal polynomials were linearized in three points (four pairs of line parameters). The total number of shifting cases is $2^4 = 16$ (PWA shifting cases). There are applied Chebyshev polynomials $T_0 \div T_3$. Total number of network parameters is 11. The robustness of the identified neural model was examined by feeding in an unknown input signal $u(t) = \sin^2(0.25t) + \cos(0.125t) - 1$ not previously used for training (Fig. 10).

Table 3. OAF NN parameters for system (33)

<table>
<thead>
<tr>
<th>W1</th>
<th>W2</th>
<th>W3</th>
<th>W4</th>
<th>W5</th>
<th>W6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0347</td>
<td>-0.2544</td>
<td>-0.3154</td>
<td>0.0955</td>
<td>0.2043</td>
<td>-0.0976</td>
</tr>
<tr>
<td>W7</td>
<td>W8</td>
<td>W9</td>
<td>W10</td>
<td>W11</td>
<td></td>
</tr>
<tr>
<td>1.0347</td>
<td>-0.2544</td>
<td>-0.3154</td>
<td>0.0955</td>
<td>0.2043</td>
<td></td>
</tr>
</tbody>
</table>

5. CONCLUSION

The proposed method significantly improves performance of existing MPC algorithms for hybrid nonlinear systems. Three studied cases were presented. It was shown that the proposed approach is effective in model precision and allowed to decrease the number of linearization points. The 3-D function was linearized by PWA function with four shifting cases and the nonlinear dynamic SISO system with sixteen shifting cases. Computation of network parameters is fast and it allows to execute identification for various parameters (order of used Chebyshev polynomials, number of linearization points) to get better performance or even to use genetic approach. Accuracy of the PWA OAF NN approximation depends on the number of linearization points, the highest order of used Chebyshev polynomials and absolute value of computed parameters of the neural network. More linearization points give better precision of the approximation but complexity of the PWA model increases. It is necessary to find suitable proportion between the number of linearization points and required precision. Absolute value of the network parameter vector plays an important role, too. Neural network with strong parameters amplifies linearization error of orthogonal polynomials. It is desirable to bound absolute values of the parameters. Suitable normalization and training method to define upper and lower bounds can be helpful. Optimal selection of linearization points (interval division) is another way to increase precision. Shifting of linearization points can refine approximation in those neurons where identified parameters are ‘strong’.

The method can be generalized for a wide range of applications, i.e. power systems, biotechnologies, automotive etc..

ACKNOWLEDGEMENTS

This paper was supported by VEGA project Nr. 1/0822/08/4.

REFERENCES


