**Slovak University of Technology in Bratislava**
**Institute of Information Engineering, Automation, and Mathematics**

# PROCEEDINGS

**of the 18th International Conference on Process Control**

**Hotel Titris, Tatranská Lomnica, Slovakia, June 14 – 17, 2011**

**ISBN 978-80-227-3517-9**

**http://www.kirp.chtf.stuba.sk/pc11**

**Editors: M. Fikar and M. Kvasnica**

Szucs, A., Kvasnica, M., Fikar, M.: MATLAB Toolbox for Automatic Approximation of Nonlinear Functions, Editors: Fikar, M., Kvasnica, M., In *Proceedings of the 18th International Conference on Process Control*, Tatranská Lomnica, Slovakia, 119–124, 2011.

Full paper online: http://www.kirp.chtf.stuba.sk/pc11/data/abstracts/054.html

# MATLAB Toolbox for Automatic Approximation of Nonlinear Functions

**Alexander Szücs** [*,1], **Michal Kvasnica** [*], and **Miroslav Fikar** [*]

[*] *Institute of Automation, Information Engineering and Mathematics, Slovak University of Technology, 812 37 Bratislava, Slovakia*

**Abstract:** : Given a nonlinear dynamical system in analytic form, the paper proposes a novel method for approximating the system by a suitable hybrid model such that the approximation accuracy is maximized. Specifically, the problem of approximating generic nonlinear functions by piecewise affine (PWA) models is considered. We show that under mild assumptions, the task can be transformed into a series of one-dimensional approximations, for which we propose an efficient solution method based on solving simple nonlinear programs. Moreover, the paper discusses a software implementation of the proposed procedure in form of a MATLAB toolbox.

*Keywords:* hybrid systems, approximation, nonlinear optimization

## 1. INTRODUCTION

Mathematical models of physical plants play a vital role in many areas, such as in rigorous simulations, analysis, or control synthesis. Typically, high model accuracy is usually desired while keeping the model complexity on an acceptable level. Traditionally, nonlinear models were preferred from simulations, while most of available control techniques are based on a local approximation around a single operating point. The concept of hybrid systems (Branicky, 1995) can be viewed as a compromise solution between accuracy of the model and its complexity. Hybrid models feature a collection of local models accompanied with logic IF-THEN conditions which enforce switching of the local dynamics. When all local models are linear (or affine), such systems are referred to as *linear* hybrid systems. Although still nonlinear due to the presence of switches, the underlying piecewise linearity allows for somewhat easier control synthesis and analysis compared to using full nonlinear models. Several mathematical frameworks capable of capturing the relation between logic rules and linear dynamics can be used: Piecewise Affine (PWA) models (Sontag, 1981), Mixed Logical Dynamical (MLD) systems (Bemporad and Morari, 1999), Linear Complementarity systems (Heemels et al., 2000) and max-min-plus-scaling models (De Schutter and Van den Boom, 2001). Under mild assumptions, all these frameworks are equivalent to each other and it is possible to transform e.g. the MLD system into a PWA model and vice-versa (Heemels et al., 2001). For the purpose of this work we consider PWA models, which use the concept of multiple linearization to approximate a given nonlinear system with arbitrary accuracy.

The problem which we address in this paper is the following: given a nonlinear dynamical model $x^+ = f(x,u)$ and a fixed complexity of its PWA approximation $\widetilde{f}(x,u) \approx f(x,u)$, how should one design $\widetilde{f}$ which minimizes the approximation error $\int (f(x,u) - \widetilde{f}(x,u))^2$? The answer is non-trivial even putting optimality of the approximation aside. Traditionally, two distinct approaches for deriving PWA approximations are used. When the mathematical formulation of the original nonlinear system is known, one can design the approximation by hand. This is usually done by employing human knowledge and experience to devise several linearization points around which the original nonlinear model should be linearized. Needless to say, placement of such points has a crucial impact on the accuracy of the approximation. The HYSDEL (Hybrid Systems Description Language) tool (Torrisi and Bemporad, 2004; Kvasnica and Herceg, 2010) can be used to accelerate this line of development. Formally, HYSDEL transforms a linguistic description of a hybrid system into the corresponding MLD model, which can then be converted into the PWA form. The language allows to define IF-THEN switching rules which, based on whether some logic condition is satisfied or not, enforce certain continuous dynamics. Another option is to use hybrid identification techniques (Ferrari-Trecate et al., 2001; Roll et al., 2004; Ferrari-Trecate, 2005) to construct the PWA approximation from the input-output measurements. The crucial advantage is that the model of the original nonlinear system is not required to be fully available. The downside, however, is that the approximation is only accurate in the interval captured by the identification data. Moreover, the procedure is computationally expensive and suited mainly to low-dimensional problems.

In this work we propose to use an optimization-based approach to derive PWA approximations of nonlinear systems whose vector field is an a-priori known function of multiple variables. After formally stating the problem in Section 2, we show in Section 3 that an optimal PWA approximation of generic nonlinear functions in one variable can be formulated and solved as a nonlinear programming problem. Subsequently, the approach is extended to deriving PWA approximations of multivariable functions in Section 4. We show that, under a certain assumption, the

[1] Corresponding author, e-mail: alexander.szucs@stuba.sk

problem boils down to solving a series of one-dimensional approximations.

The algorithmic and software implementation of the approximation procedure are then discussed in Section 5. Specifically, we introduce a new software toolbox which packs the proposed approximation strategy in an easily accessible form. Specifically, the toolbox allows user to perform the approximation either directly from MATLAB's command line, or by using a custom graphical user interface. Short, yet illuminating examples are provided to illustrate capabilities of the toolbox.

## 2. PROBLEM STATEMENT

We consider generic dynamic systems in discrete-time

$$x^+ = f(x,u), \tag{1}$$

where the vector field $f(\cdot,\cdot)$ is assumed to be continuous in the state variables $x \in \mathbb{R}^{n_x}$ and in the inputs $u \in \mathbb{R}^{n_u}$. System states and inputs are assumed to be constrained to connected and closed domains $\mathcal{X} \subset \mathbb{R}^{n_x}$ and $\mathcal{U} \subset \mathbb{R}^{n_u}$, respectively.

The objective is to approximate (1) by a different dynamic system $x^+ = \widetilde{f}(x,u)$ whose vector field $\widetilde{f}(x,u)$ is a PWA function which consists of a pre-specified number $N$ of local linear dynamics:

$$\widetilde{f}(x,u) = \begin{cases} A_1x + B_1u + c_1 & \text{if } \left[\begin{smallmatrix} x \\ u \end{smallmatrix}\right] \in \mathcal{R}_1 \\ \quad\vdots & \quad\vdots \\ A_Nx + B_Nu + c_N & \text{if } \left[\begin{smallmatrix} x \\ u \end{smallmatrix}\right] \in \mathcal{R}_N. \end{cases} \tag{2}$$

Here, $A_i \in \mathbb{R}^{n_x \times n_x}$, $B_i \in \mathbb{R}^{n_x \times n_u}$, $c_i \in \mathbb{R}^{n_x}$, are the state-update matrices of the $i$-th local linear approximation, and $\mathcal{R}_i \subset \mathbb{R}^{n_x \times n_u}$ is the region of validity of the $i$-th local model satisfying $\mathcal{R}_i \neq \emptyset$, $\mathcal{R}_i \cap \mathcal{R}_j = \emptyset$, $\forall i \neq j$, and $\cup_i \mathcal{R}_i = \mathcal{X} \times \mathcal{U}$.

Formally, the problem which we aim at solving can be stated as follows:

*Problem 2.1.* Given a nonlinear vector field $f(x,u)$ of system (1), find the PWA approximation (2) of pre-specified complexity which minimizes the approximation error

$$e_{\text{aprx}} := \int (f(x,u) - \widetilde{f}(x,u))^2 \,\mathrm{d}x\mathrm{d}u, \tag{3}$$

where the integral is evaluated over the whole region of validity of (1), i.e. over $\mathcal{X} \times \mathcal{U}$.

In the sequel we show how to solve Problem 2.1 provided that the vector field $f(z)$, $z = [x,\ u]^T$ satisfies the following assumption.

*Assumption 2.2.* The function $f(z_1,\dots,z_n)$ can be written as $\sum_{i=1}^{n} \alpha_i \Big( \prod_{j=p_i}^{q_i} f_j(z_j) \Big)$.

As an example, the function $z_1 e^{z_2}$ satisfies such an assumption, while the function $e^{z_1 z_2}$ does not. Although the assumption is somewhat restrictive, the gained advantage is that approximating any multivariable function $f(z_1,\dots,z_n)$ boils down to solving a series of 1D problems, as evidenced in the following two sections.

*Remark 2.3.* Since the approximation procedure discussed in the sequel considers only the vector field in the right-hand-side of (1), continuous-time systems $\dot{x} = f(x,u)$ can be treated as well.

## 3. FUNCTIONS IN ONE VARIABLE

First, we consider the one-dimensional case, i.e. approximating a nonlinear function $f(z) : \mathbb{R} \mapsto \mathbb{R}$, with domain $\mathcal{Z} \subset \mathbb{R}$, by a PWA function $\widetilde{f}(z) = a_i z + c_i$ if $z \in \mathcal{R}_i$. Since $\mathcal{Z}$ is assumed to be connected and closed, it is a line segment $[\underline{z},\ \overline{z}]$. Regions $\mathcal{R}_i$ define the partition of such a line into $N$ non-overlapping parts, i.e. $\mathcal{R}_1 = [\underline{z},\ r_1]$, $\mathcal{R}_2 = [r_1,\ r_2], \dots, \mathcal{R}_{N-1} = [r_{N-2},\ r_{N-1}], \mathcal{R}_N = [r_{N-1},\ \overline{z}]$ with $\cup_i \mathcal{R}_i = [\underline{z},\ \overline{z}]$. Solving Problem 2.1 then becomes to find the slopes $a_i$, offsets $c_i$ and breakpoints $r_i$ such that the approximation error is minimized, i.e.

$$\min_{a_i,c_i,r_i} \int_{\underline{z}}^{\overline{z}} (f(z) - \widetilde{f}(z))^2 \,\mathrm{d}z \tag{4a}$$

$$\text{s.t. } \widetilde{f}(z) = \begin{cases} a_1 z + c_1 & \text{if } z \in [\underline{z},\ r_1] \\ \quad\vdots & \quad\vdots \\ a_N z + c_N & \text{if } z \in [r_{N-1},\ \overline{z}] \end{cases} \tag{4b}$$

$$\underline{z} \leq r_1 \leq \dots \leq r_{N-1} \leq \overline{z}, \tag{4c}$$

$$a_i r_i + c_i = a_{i+1} r_i + c_{i+1},\ i = 1,\dots,N-1, \tag{4d}$$

where (4d) enforces continuity of $\widetilde{f}(z)$ along the breakpoints $r_i$. The IF-THEN based nonlinear constraint (4b) can be eliminated by observing that, by definition, regions $\mathcal{R}_i$ are non-overlapping, and the integral in (4a) can hence be written as

$$\int_{\underline{z}}^{\overline{z}} \big(f(z) - \widetilde{f}(z)\big)^2 \mathrm{d}z = \sum_{i=1}^{N} \Big( \int_{r_{i-1}}^{r_i} \big(f(z) - (a_i z + c_i)\big)^2 \mathrm{d}z \Big), \tag{5}$$

with $r_0 = \underline{z}$ and $r_N = \overline{z}$. The NLP (4) can therefore be written as

$$\min_{a_i,c_i,r_i} \sum_{i=1}^{N} \Big( \int_{r_{i-1}}^{r_i} \big(f(z) - (a_i z + c_i)\big)^2 \,\mathrm{d}z \Big) \tag{6a}$$

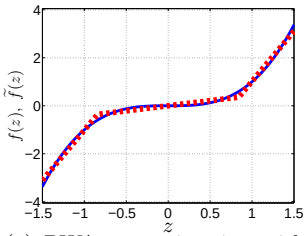
$$\text{s.t. } \underline{z} \leq r_1 \leq \dots \leq r_{N-1} \leq \overline{z}, \tag{6b}$$

$$a_i r_i + c_i = a_{i+1} r_i + c_{i+1},\ i = 1,\dots,N-1. \tag{6c}$$

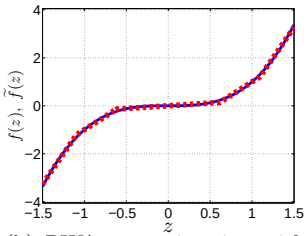
*Remark 3.1.* The number of approximation segments can be be reduced by normalizing the domain of $f$ to an interval $[-1,\ 1]$.

For simple functions $f(z)$, the integral in (6a) can be expressed in an analytical form in unknowns $a_i, c_i, r_i$, along with the corresponding gradients. For more complex expressions, the integrals can be evaluated numerically, e.g. by using the trapezoidal rule. In either case, problem (6) can be solved to a local optimality e.g. by using the `fmincon` solver of MATLAB. Alternatively, one can use global optimization methods (Adjiman et al., 1996; Papamichail and Adjiman, 2004; Chachuat et al., 2006) that guarantee that an $\epsilon$-neighborhood of the global optimum can be found.

*Example 3.2.* Consider the function $f(z) = z^3$ on domain $-1.5 \leq z \leq 1.5$. The analytic form of the integral (6a) is




(a) PWA approximation with 3 regions.




(b) PWA approximation with 5 regions.

Fig. 1. Graph of $f(z) = z^3$ (blue line) and the PWA approximations $\widetilde{f}(z)$ (red dashed lines).

$$\sum_{i=1}^{N} \big(c_i^2(r_i + r_{i-1}) + a_i c_i(r_i^2 - r_i^2) + \frac{a_i^2}{3}(r_i^3 - r_{i-1}^3) - \\ -\frac{c_i}{2}(r_i^4 - r_{i-1}^4) - \frac{2a_i}{5}(r_i^5 - r_{i-1}^5) + \frac{1}{7}(r_i^7 - r_{i-1}^7)\big),$$

with $r_0 = -1.5$ and $r_N = 1.5$. The PWA approximation of $f(z)$ with $N = 3$ regions was found by solving the NLP (6) using `fmincon`, which took 0.05 seconds on a 2.4 GHz CPU running MATLAB 2009b. The obtained PWA approximation is then given by

$$\widetilde{f}(z) = \begin{cases} 4.1797z + 3.1621 & \text{if } -1.5 \le z \le -0.8423 \\ 0.4257z & \text{if } -0.8423 \le z \le 0.8423 \\ 4.1797z - 3.1621 & \text{if } 0.8423 \le z \le 1.5 \end{cases}$$

The approximation accuracy can be increased by roughly a factor of 10 by approximating $f(z)$ by $N = 5$ regions, as can be seen from Figure 1.

## 4. MULTIVARIABLE FUNCTIONS

The task is to approximate a given multivariable function $f(z_1, \dots, z_n) : \mathbb{R}^n \mapsto \mathbb{R}$ with domain $\mathcal{Z} \subset \mathbb{R}^n$ by a PWA function $\widetilde{f}(z_1, \dots, z_n)$, defined over the same domain, such that the approximation error (3) is minimized.

*Definition 4.1.* (Williams (1993)). Function $f(z_1, \dots, z_n)$ is called *separable* if it can be expressed as the sum of functions of a single variable, i.e. $f(z_1, \dots, z_n) = f_1(z_1) + \dots + f_n(z_n)$.

If $f(z_1, \dots, z_n)$ is readily separable (e.g. when $f(z_1, z_2) = e^{z_1} + \sin(z_2)$), its optimal PWA approximation can be obtained by applying the 1D scenario of Section 3 to the individual components of the function, i.e. $\widetilde{f}(z_1, \dots, z_n) = \widetilde{f}_1(z_1) + \dots + \widetilde{f}_n(z_n)$. The total number of regions over which the PWA approximation $\widetilde{f}(\cdot)$ is defined is hence given by $\sum_{j=1}^{n} N_j$, where $N_j$ is the pre-specified complexity of the $j$-th approximation $\widetilde{f}_j(z_j)$.

A surprisingly large number of non-separable functions can be converted into the separable form by applying a simple trick, elaborated in more details e.g. in Williams (1993). To introduce the procedure, consider a non-separable function $f(z_1, z_2) = z_1 z_2$ with domain $\mathcal{Z} := [\underline{z}_1,\ \overline{z}_1] \times [\underline{z}_2,\ \overline{z}_2]$. Define two new variables

$$y_1 = (z_1 + z_2), \quad y_2 = (z_1 - z_2). \tag{7}$$

Then it is easy to verify that $1/4(y_1^2 - y_2^2) = z_1 z_2$. The coordinate transformation therefore transforms the original function into a separable form, where both terms ($y_1^2$ and $y_2^2$) are now functions of a single variable. The procedure of Section 3 can thus be applied to compute PWA approximations of $f_{y_1}(y_1) := y_1^2$ and $f_{y_2}(y_2) := y_2^2$, where the function arguments relate to $z_1$ and $z_2$ via (7). Important to notice is that $f_{y_1}(\cdot)$ and $f_{y_2}(\cdot)$ have different domains, therefore their PWA approximations $\widetilde{f}_{y_1}(y_1) \approx y_1^2$ and $\widetilde{f}_{y_2}(y_2) \approx y_2^2$ will, in general, be different. Specifically, the domain of $f_{y_1}(\cdot)$ is $[\underline{y}_1,\ \overline{y}_1]$ with $\underline{y}_1 = \min\{z_1 + z_2 \mid \underline{z}_1 \le z_1 \le \overline{z}_1,\ \underline{z}_2 \le z_2 \le \overline{z}_2\}$ and $\overline{y}_1 = \max\{z_1 + z_2 \mid \underline{z}_1 \le z_1 \le \overline{z}_1,\ \underline{z}_2 \le z_2 \le \overline{z}_2\}$. Similarly, the domain of $f_{y_2}(\cdot)$ is $[\underline{y}_2,\ \overline{y}_2]$, whose boundaries can be computed by respectively minimizing and maximizing $z_1 - z_2$ subject to the constraint $[z_1,\ z_2]^T \in \mathcal{Z}$. The overall PWA approximation $\widetilde{f}(z_1, z_2) \approx z_1 z_2$ then becomes

$$\widetilde{f}(z_1, z_2) = 1/4(\widetilde{f}_{y_1}(z_1 + z_2) - \widetilde{f}_{y_2}(z_1 - z_2)). \tag{8}$$

The value of $\widetilde{f}(z_1, z_2)$ for any points $z_1, z_2$ is obtained by subtracting the value of the PWA function $\widetilde{f}_{y_2}(\cdot)$ evaluated at the point $z_1 - z_2$ from the function value of $\widetilde{f}_{y_1}(\cdot)$ evaluated at $z_1 + z_2$, followed by a linear scaling.

The procedure naturally extends to multivariable functions represented by the product of two nonlinear functions of a single variable, i.e. $f(z_1, z_2) = f_1(z_1) f_2(z_2)$. Here, the transformation (7) becomes

$$y_1 = f_1(z_1) + f_2(z_2), \quad y_2 = f_1(z_1) - f_2(z_2). \tag{9}$$

Therefore, $1/4(y_1^2 - y_2^2) = f(z_1, z_2)$ still holds. Let $f_{y_1}(y_1) := y_1^2$ and $f_{y_2}(y_2) := y_2^2$. The domain of $f_{y_1}(\cdot)$ is $[\underline{y}_1,\ \overline{y}_1]$ and $\operatorname{dom} f_{y_2}(\cdot) = [\underline{y}_2,\ \overline{y}_2]$ with

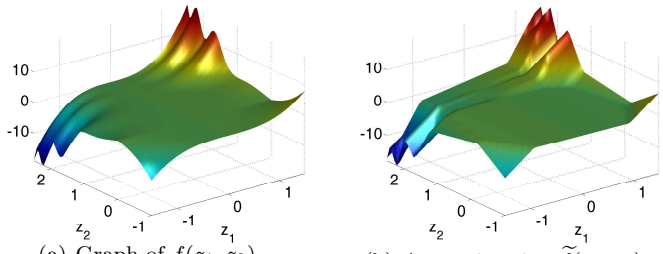
$$\underline{y}_1 = \min\{f_1(z_1) + f_2(z_2) \mid [z_1,\ z_2]^T \in \mathcal{Z}\}, \tag{10a}$$
$$\overline{y}_1 = \max\{f_1(z_1) + f_2(z_2) \mid [z_1,\ z_2]^T \in \mathcal{Z}\}, \tag{10b}$$
$$\underline{y}_2 = \min\{f_1(z_1) - f_2(z_2) \mid [z_1,\ z_2]^T \in \mathcal{Z}\}, \tag{10c}$$
$$\overline{y}_2 = \max\{f_1(z_1) - f_2(z_2) \mid [z_1,\ z_2]^T \in \mathcal{Z}\}, \tag{10d}$$

which can be computed by solving four NLP problems. Finally, since all expressions are now functions of a single variable, the PWA approximations $\widetilde{f}_1(z_1) \approx f_1(z_1)$, $\widetilde{f}_2(z_2) \approx f_2(z_2)$, $\widetilde{f}_{y_1}(y_1) \approx f_{y_1}(y_1)$, and $\widetilde{f}_{y_2}(y_2) \approx f_{y_2}(y_2)$ can be computed by solving the NLP (6). The overall optimal PWA approximation $\widetilde{f}(z_1, z_2) \approx f(z_1, z_2)$ then becomes

$$\widetilde{f}(z_1, z_2) = 1/4\Big(\widetilde{f}_{y_1}\big(\widetilde{f}_1(z_1) + \widetilde{f}_2(z_2)\big) - \widetilde{f}_{y_2}\big(\widetilde{f}_1(z_1) - \widetilde{f}_2(z_2)\big)\Big). \tag{11}$$

The evaluation procedure is similar as above. I.e., given the arguments $z_1$ and $z_2$, one first evaluates $\widetilde{z}_1 = \widetilde{f}_1(z_1)$ and $\widetilde{z}_2 = \widetilde{f}_2(z_2)$. Subsequently, one evaluates $\widetilde{y}_1 = \widetilde{f}_{y_1}(\cdot)$ with the argument $\widetilde{z}_1 + \widetilde{z}_2$, then $\widetilde{y}_2 = \widetilde{f}_{y_2}(\cdot)$ at the point $\widetilde{z}_1 - \widetilde{z}_2$. Finally, $\widetilde{f}(z_1, z_2) = 1/4(\widetilde{y}_1 - \widetilde{y}_2)$.

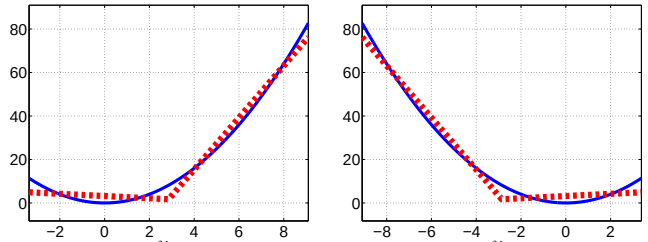
*Example 4.2.* Consider a non-separable function given by $f(z_1, z_2) = f_1(z_1) f_2(z_2)$ with $f_1(z_1) = z_1^3$, $f_2(z_2) = |z_2| + 0.5 z_2^2 - \sin(z_2)^3$ on domain $[-1.5,\ 1.5] \times [-1,\ 2.5]$. Graph of the function is shown in Figure 2(a). In order to convert $f(z_1, z_2)$ into a separable form, we introduce variables $y_1$ and $y_2$ as per (9). The PWA approximation $\widetilde{f}(z_1, z_2) \approx f(z_1, z_2)$ is then given by (11). Here, $\widetilde{f}_1(z_1)$ was obtained by approximating $f_1(z_1)$ by a PWA




(a) Graph of $f(z_1, z_2)$. (b) Approximation $\widetilde{f}(z_1, z_2)$

Fig. 2. Graph of $f(z_1, z_2)$ and its PWA approximation (11) in Example 4.2.

function with 3 regions as shown in Figure 1(a), while $\widetilde{f}_2(z_2) \approx f_2(z_2)$ was approximated by 7 regions. Subsequently, the domains $[\underline{y}_1,\ \overline{y}_1]$ and $[\underline{y}_2,\ \overline{y}_2]$ were computed via (10), which resulted into $\operatorname{dom} y_1 = [-3.374,\ 9.095]$ and $\operatorname{dom} y_2 = [-9.095,\ 3.374]$. Finally, the PWA approximations $\widetilde{f}_{y_1}(y_1) \approx y_1^2$ and $\widetilde{f}_{y_2}(y_2) \approx y_2^2$ were obtained by solving the NLP (6) with $N = 2$. Graphs of $y_1^2$, $y_2^2$ and their respective PWA approximations are presented in Figure 3. The overall approximation $\widetilde{f}(z_1, z_2)$ therefore consists of 14 regions. Despite a rather crude approximation of the square functions, the combined PWA function (11), shown in Figure 2(b), features only a minor average approximation error of 3% and a worst-case error of 15%. By increasing the number of linearizations for $y_1^2$ and $y_2^2$ from $N = 2$ to $N = 4$ (hence increasing the complexity of $\widetilde{f}(z_1, z_2)$ from 14 to 18 regions), the average and worst-case errors can be further reduced to 1% and 8%, respectively.




(a) $y_1^2$ (b) $y_2^2$

Fig. 3. Functions $y_i^2$ (blue) and their PWA approximation $\widetilde{f}_{y_i}(y_i)$ (red dashed lines) in Example 4.2.

Separation of multivariable functions with more than two terms can be performed in an inductive manner. Consider $f(z_1, z_2, z_3) = f_1(z_1)f_2(z_2)f_3(z_3)$. First, approximate the product $f_1(z_1)f_2(z_2)$ by a PWA function of the form of (11), which requires four PWA approximations

$$\widetilde{f}_1(\cdot) \approx f_1(\cdot),\ \widetilde{f}_2(\cdot) \approx f_2(\cdot),\ \widetilde{f}_{y_1}(\cdot) \approx y_1^2,\ \widetilde{f}_{y_2}(\cdot) \approx y_2^2,$$

with $y_1$ and $y_2$ as in (9). Let $f_a(z_1, z_2) := f_1(z_1)f_2(z_2)$. Then $f(z_1, z_2, z_3) = f_a(z_1, z_2)f_3(z_3)$, which can again be approximated as a product of two functions. Specifically, define

$$y_3 = f_a(\cdot) + f_3(z_3), \quad y_4 = f_a(\cdot) - f_3(z_3), \tag{12}$$

and hence $f_a(z_1, z_2)f_3(z_3) = 1/4(y_3^2 - y_4^2)$. The domains over which $y_3^2$ and $y_4^2$ need to be approximated are, respectively, $[\underline{y}_3,\ \overline{y}_3]$ and $[\underline{y}_4,\ \overline{y}_4]$ with

$$\underline{y}_3 = \min\{f_1(z_1)f_2(z_2) + f_3(z_3) \,|\, z \in \mathcal{Z}\}, \tag{13a}$$
$$\overline{y}_3 = \max\{f_1(z_1)f_2(z_2) + f_3(z_3) \,|\, z \in \mathcal{Z}\}, \tag{13b}$$
$$\underline{y}_4 = \min\{f_1(z_1)f_2(z_2) - f_3(z_3) \,|\, z \in \mathcal{Z}\}, \tag{13c}$$
$$\overline{y}_4 = \max\{f_1(z_1)f_2(z_2) - f_3(z_3) \,|\, z \in \mathcal{Z}\}, \tag{13d}$$

and $z = [z_1,\ z_2,\ z_3]^T$. Subsequently, three additional PWA approximations

$$\widetilde{f}_{y_3}(y_3) \approx y_3^2,\ \widetilde{f}_{y_4}(y_4) \approx y_4^2,\ \widetilde{f}_3(z_3) \approx f_3(z_3)$$

need to be computed over the corresponding domains. The aggregated optimal PWA approximation $\widetilde{f}(z_1, z_2, z_3) \approx f(z_1)f(z_2)f(z_3)$ consists of 7 individual approximations and is given by

$$\widetilde{f}(\cdot) = 1/4\Big(\underbrace{\widetilde{f}_{y_3}\big(\widehat{f}_a + \widetilde{f}_3(z_3)\big)}_{\widehat{y}_3} - \underbrace{\widetilde{f}_{y_4}\big(\widehat{f}_a - \widetilde{f}_4(z_3)\big)}_{\widehat{y}_4}\Big). \tag{14}$$

Here, $\widehat{f}_a$ is the function value of $\widetilde{f}_a(z_1, z_2) \approx f_1(z_1)f_2(z_2)$ at $z_1$ and $z_2$, where $\widetilde{f}_a(\cdot)$ is obtained from (11), i.e.:

$$\widehat{f}_a = 1/4\Big(\underbrace{\widetilde{f}_{y_1}\big(\widetilde{f}_1(z_1) + \widetilde{f}_2(z_2)\big)}_{\widehat{y}_1} - \underbrace{\widetilde{f}_{y_2}\big(\widetilde{f}_1(z_1) - \widetilde{f}_2(z_2)\big)}_{\widehat{y}_2}\Big). \tag{15}$$

The overall PWA approximation $\widetilde{f}(z_1, z_2, z_3)$ can then be evaluated, for any $z_1,\ z_2,\ z_3 \in \mathcal{Z}$, by computing the function values of the respective approximations in the following order:

**Step 1:** $\widehat{y}_1 = \widetilde{f}_{y_1}(\widetilde{f}_1(z_1) + \widetilde{f}_2(z_2))$,
**Step 2:** $\widehat{y}_2 = \widetilde{f}_{y_2}(\widetilde{f}_1(z_1) - \widetilde{f}_2(z_2))$,
**Step 3:** $\widehat{y}_3 = \widetilde{f}_{y_3}(1/4(\widehat{y}_1 - \widehat{y}_2) + \widetilde{f}_3(z_3))$,
**Step 4:** $\widehat{y}_4 = \widetilde{f}_{y_4}(1/4(\widehat{y}_1 - \widehat{y}_2) - \widetilde{f}_3(z_3))$,
**Step 5:** $\widetilde{f}(z_1, z_2, z_3) = 1/4(\widehat{y}_3 - \widehat{y}_4)$.

Such an inductive procedure can be repeated *ad-infinitum* to derive PWA approximations of any multivariable function which satisfies Assumption 2.2. In general, the PWA approximation will consists of $2p + n$ individual PWA functions, where $n$ is the number of variables in $f(z_1, \ldots, z_n)$ and $p$ is the number of products between individual subfunctions $f_j(z_j)$. As an example, for $f(\cdot) := \alpha_1 f_1(z_1)f_2(z_2)f_4(z_4) + \alpha_2 f_3(z_3)f_5(z_5)$ we have $p = 3$. We remark that inclusion of scalar multipliers $\alpha_j$ into the PWA description of the form (14)–(15) is straightforward and only requires linear scaling of the corresponding terms.

*Remark 4.3.* Since approximation of multivariable functions boils down to a series of 1D approximations which are then aggregated by a linear relation in (8), the overall approximation error is proportional to the sum of individual approximation errors.

*Remark 4.4.* Due to a linear nature of the aggregation in (8) and due to the fact that each single 1D approximation is continuous due to (6c), the overall multivariable approximation $\widetilde{f}$ is continuous as well.

## 5. SOFTWARE IMPLEMENTATION

Next, we discuss software implementation of the approximation procedure described above. The implementation is provided in a form of an open-source MATLAB toolbox, called AUTOPROX, which is freely available

from `http://www.kirp.chtf.stuba.sk/~sw/`. The toolbox provides two types of user interfaces. Input data can either be provided directly from the command line or, alternatively, entered using a graphical interface.

### *5.1 Command-Line Interface*

The command-line interface is illustrated first by revisiting Example 3.2. To approximate the function $f(z) = z^3$, one proceeds as follows:

```
syms z
f = z^3
bounds = [-1.5, 1.5]
regions = 3
[aprx, data] = autoprox_1d(f, bounds, regions)
```

Here, AUTOPROX uses the Symbolic Toolbox to define symbolic representation of the function to be approximated on a given domain (represented by the `bounds` variable), with a given number of PWA segments (the `regions` variable). The first output argument (denotes as `aprx` here) is a function handle, which can be used e.g. to plot the approximation:

```
x = -1.5:0.001:1.5
plot(x, x.^3, x, aprx(x), '--')
```

which will generate a plot as seen in Figure 1(a). The second output (stored in the `data` variable) can be used to export the PWA approximation into the HYSDEL language:

```
hysdel_1d(data, 'filename.hys')
```

The generated HYSDEL model can be subsequently compiled by the HYSDEL compiler, which will provide a mathematical model suitable e.g. for control synthesis.

Approximation of 2D functions can be performed in a similar manner. Let us again consider Example 4.2, i.e. the task is to approximate the function $f(z_1, z_2) = z_1^3(|z_2| + 0.5z_2^2 - \sin(z_2)^3)$ on domain $[-1.5,\ 1.5] \times [-1,\ 2.5]$. Again, the first step is to define the function using symbolic variables:

```
syms z1 z2
f1 = z1^3
f2 = abs(z2) + 0.5*z2^2 - sin(z2^3)
```

Next, the function domain and number of approximation segments need to be provided:

```
f1_bounds = [-1.5, 1.5]
f2_bounds = [-1, 2.5]
f1_regions = 3
f2_regions = 7
y1_regions = 2
y2_regions = 2
```

Finally, the approximation $\widetilde{f}(z_1, z_2)$ can be obtained by calling

```
[aprx,data] = autoprox_2d(f1,f2,f1_bounds,f2_bounds,...
 f1_regions, f2_regions,y1_regions, y2_regions)
```

Similarly as in the previous example, the `aprx` output is a function handle which can be used to directly evaluate the approximation at some given values of $z_1$ and $z_2$, e.g.

```
z1 = 0.5
z2 = -1
true_value = z1^3*(abs(z2) + 0.5*z2^2 - sin(z2^3))
aprx_value = aprx(z1, z2)
```

The second output (called `data`) again serves to generate the HYSDEL version of the approximation:

```
hysdel_2d(data, 'filename.hys')
```

Approximation of $n$-dimensional functions can be obtained by calling the `autoprox_nd` function. A detailed description of its calling syntax is omitted due to brevity, but is provided in the distribution package of AUTOPROX.

### *5.2 Graphical User Interface (GUI)*

The GUI allows to perform the approximation in an easily accessible manner where all data can be entered conveniently without the need to remember the exact calling syntax of individual approximation functions.

The main window of the GUI is shown in Figure 4. The user starts by selecting the type of approximation using radio buttons. Then, he provides the symbolic representation of the function to approximate in the *FUNCTION* text box. The domain of the function, represented by its minimal and maximal bounds, has to be filled out next. After providing all necessary details, the user can select the

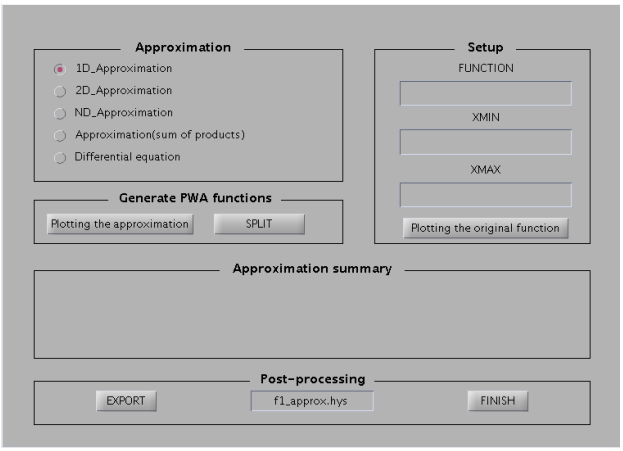



Fig. 4. Basic GUI window.

number of approximation regions by a drop-down menu, as shown in Figure 5. Afterwards, the approximation is computed by clicking the *SPLIT* button. A concise statistical evaluation of the approximation will then appear in a corresponding section of the GUI. It informs the user about the approximation quality, represented by average and worst-case approximation errors. Finally, the approximation can be exported to a HYSDEL source by clicking the *EXPORT* button.

It should be noted that the GUI is still subject to active development and substantial modifications are expected within next following months.

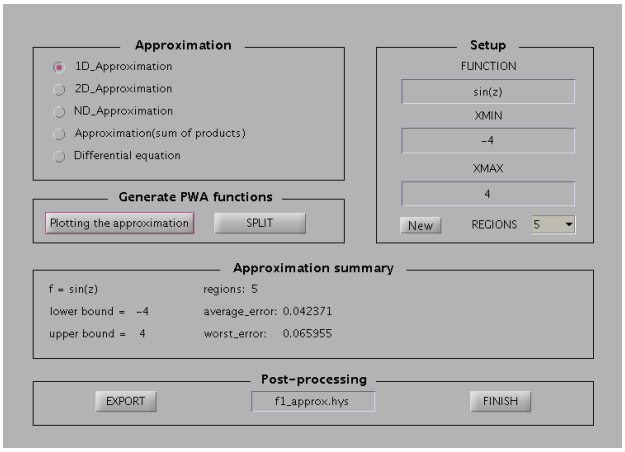



Fig. 5. GUI windows after performing approximation.

## 6. CONCLUSIONS

We have shown that a large class of dynamical systems with nonlinear vector fields can be approximated by PWA systems of fixed complexity in an optimal manner. The procedure boils down to solving a series of one-dimensional problems for which efficient solution methods exist. Derivation of the approximation can be easily automated and the HYSDEL variant of the hybrid approximation can be generated, hence allowing for subsequent control synthesis based on the hybrid model. A MATLAB toolbox which implements the proposed approximation strategy was described as well. The toolbox allows user to enter data either via command line, or by using a graphical user interface. An experimental version of the toolbox is available for free download at http://www.kirp.chtf.stuba.sk/~sw/.

## ACKNOWLEDGMENT

The authors are pleased to acknowledge the financial support of the Scientific Grant Agency of the Slovak Republic under the grants 1/0071/09 and 1/0095/11. This work was supported by the Slovak Research and Development Agency under the contracts No. VV-0029-07 and No. LPP-0092-07.

## REFERENCES

Adjiman, C.S., Androulakis, I.P., Maranas, C.D., and Floudas, C.A. (1996). A global optimization method, $\alpha$BB for process design. *Computers and Chemical Engineering*, 20, 419–424.

Bemporad, A. and Morari, M. (1999). Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3), 407–427.

Branicky, M. (1995). *Studies in hybrid systems: modeling, analysis, and control*. Ph.D. thesis, LIDS-TH 2304, Massachusetts Institute of Technology, Cambridge, MA.

Chachuat, B., Singer, A.B., and Barton, P.I. (2006). Global methods for dynamic optimization and mixed-integer dynamic optimization. *Ind. Eng. Chem. Res.*, 45(25), 8373–8392.

De Schutter, B. and Van den Boom, T. (2001). On model predictive control for max-min-plus-scaling discrete event systems. *Automatica*, 37(7), 1049–1056.

Ferrari-Trecate, G. (2005). Hybrid Identification Toolbox (HIT). Available from http://www-rocq.inria.fr/who/Giancarlo.Ferrari-Trecate/HIT_toolbox.html.

Ferrari-Trecate, G., Muselli, M., Liberati, D., and Morari, M. (2001). Identification of Piecewise Affine and Hybrid Systems. In *Proc. on the American Control Conference*, 3521–3526. Arlington (VA), USA.

Heemels, W.P.M., De Schutter, B., and Bemporad, A. (2001). Equivalence of hybrid dynamical models. *Automatica*, 37(7), 1085–1091.

Heemels, W., Schumacher, J., and Weiland, S. (2000). Linear complementarity systems. *SIAM Journal on Applied Mathematics*, 60(4), 1234–1269.

Kvasnica, M. and Herceg, M. (2010). HYSDEL 3.0. Available from http://kirp.chtf.stuba.sk/~kvasnica/hysdel3/.

Papamichail, I. and Adjiman, C.S. (2004). Global optimization of dynamic systems. *Computers and Chemical Engineering*, 28, 403–415.

Roll, J., Bemporad, A., and Ljung, L. (2004). Identification of piecewise affine systems via mixed-integer programming. *Automatica*, 40, 37–50.

Sontag, E.D. (1981). Nonlinear regulation: The piecewise linear approach. *IEEE Trans. on Automatic Control*, 26(2), 346–358.

Torrisi, F. and Bemporad, A. (2004). HYSDEL — A tool for generating computational hybrid models for analysis and synthesis problems. *IEEE Transactions on Control Systems Technology*, 12, 235–249.

Williams, H. (1993). *Model Building in Mathematical Programming*. John Wiley & Sons, Third Edition.