EPiC Series in Computer Science

Volume 34, 2015, Pages 98–112

ARCH14-15. 1st and 2nd International Workshop on Applied veRification for Continuous and Hybrid Systems

Piecewise-Affine Approximations for a Powertrain Control Verification Benchmark

Jyotirmoy V. Deshmukh[1], Hisahiro Ito[1], Xiaoqing Jin[1], James Kapinski[1], Ken Butts[1], Jürgen Gerhard[2], Behzad Samadi[2], Kevin Walker[2], and Yuzhen Xie[3]

[1] Toyota Technical Center, USA
{firstname.lastname}@tema.toyota.com
[2] Maplesoft, Canada
{jgerhard, bsamadi, kwalker}@maplesoft.com
[3] Critical Outcome Technologies Inc., Canada
yxie@criticaloutcome.com

Abstract

We present a benchmark example of an automotive powertrain control system converted to a hybrid system with piecewise-affine (PWA) continuous dynamics. The purpose is to provide an example of an industrial nonlinear system that is amenable to existing software tools for performing verification of safety properties for hybrid systems. Existing algorithmic approaches to hybrid system verification require that system representations are restricted to specific classes of models. Therefore, it is important to develop and evaluate techniques to generate approximate models that adhere to the required class of systems without introducing an unacceptable amount of approximation errors. The example we present is intended to demonstrate a symbolic method to automatically approximate a nonlinear model with a PWA approximation while respecting a given bound on the approximation error. While existing tools are applicable to the resulting model, the scale of the PWA model is intended to challenge the capabilities of these tools. We conduct an experimental comparison between the original and PWA models and present our observations and discuss challenges for the research community.

Category: Industrial **Difficulty:** High

1 Introduction

A large body of extant literature in control theory and hybrid systems theory relies on a linear/affine or piecewise-affine (PWA) representation of the underlying systems for design or analysis. In industrial-scale systems, such as those found in the automotive domain, such an assumption is usually unrealistic, as high-fidelity models for physical processes are often nonlinear. In order to apply the rich set of analysis and design techniques for affine and PWA systems, we then have to make further approximations, and translate the nonlinear dynamical-systems models to linear or PWA models.

There are numerous tools and techniques for generating PWA approximations of nonlinear models [2, 14]. A typical challenge faced by many of these techniques is a tradeoff between complexity (i.e., the number of affine regions in the resulting model) and accuracy (i.e., the maximum error between the original nonlinear model and the PWA model in any given region). While accuracy is a serious consideration if the PWA model is to be used for any meaningful analysis or design, a representation with too many regions will usually render any analysis technique too slow to be practical.

PWA models are of special interest when applying formal verification approaches such as reachability analysis. Given a set of initial states, and a time horizon T, reachability analysis computes a conservative approximation of the set of states reachable from time 0 to time T. Several software tools have been developed to verify safety properties for hybrid systems. Tools such as Kronos [15] and UPPAAL [9] apply to timed systems, i.e., systems whose continuous dynamics are given by clocks. The HyTech tool extends to systems with continuous dynamics given by constant derivatives [6].

In this paper, we evaluate two methods: a simplex-partitioning method that imposes a simplicial decomposition on the global state space, and a *nested* method that performs local PWA approximations for nonlinear subexpressions appearing in the dynamical equations for the system under consideration. Both methods are provided by a proprietary package from Maplesoft [5]. Our experiments target the SpaceEx tool, which is a tool for verifying safety properties of linear hybrid systems [4]. SpaceEx allows the designer to specify a hybrid system and a property to be verified, in the form of a temporal logic formula. SpaceEx handles hybrid systems with affine continuous dynamics and polytopic guards and invariant sets. SpaceEx is based on the PHAver technology, which uses infinite precision representations of polytopic sets to perform the reachable set estimation [3]. SpaceEx incorporates several recent advances in reachable set estimation, such as zonotope [1] representations and support functions [10], which can increase the efficiency and accuracy of the reachable set estimations.

2 A/F Ratio Control Model

In this section, we present a brief review of the air-fuel control model; in the sequel, we present two models obtained by piecewise-affine approximations. The model we present is a closed-loop model, i.e., it contains a model of the plant and a model of the controller. This model is unchanged from the one that appeared before in [7] and [8].

Plant Model. The plant model contains a representation of the throttle, the intake manifold, the cylinder subsystem and the exhaust subsystem. Several physical phenomena such as fuel injection dynamics (e.g. fuel puddling after injection), exhaust system dynamics (e.g. exhaust gas transport dynamics), and sensor dynamics (e.g. the O_2 sensor dynamics) are assumed to be either removed or replaced with first-order approximations. Furthermore, we replace look-up tables in the plant that model nonlinear relationships between system parameters and the operating conditions with polynomial approximations. The result is a nonlinear, continuous-time model with two exogenous inputs: the throttle angle in degrees (denoted θ) and the engine speed in rad/sec (denoted ω), and two continuous states: the intake manifold pressure in bars (denoted p) and the measured air-fuel ratio (denoted λ). The actual equations are presented in the appendix.

Controller. The controller contains two parts, the first is an open-loop feedforward component that estimates the intake manifold pressure p by observing the mass of the air ($\dot{m}_{af}$) flowing into the manifold. This is then used to compute the mass of the air flowing into the cylinder

that is used during the combustion process. In a real system, such an estimator is designed to compensate for phenomena such as parameter variation and sensor noise by using, for example, an extended Kalman filter. For simplicity, we choose a naive observer that assumes nearly perfect knowledge of the nonlinear function modeling the relationship between $\dot{m}_{af}$ and p. The second component is the Proportional + Integral (PI) controller that regulates the air-fuel ratio. Thus, the controller is assumed to have two states: the estimated intake manifold pressure (denoted p_{est}), and the state of the integrator in the PI controller (denoted i).

In the actual system, the control software is modeled as a discrete-time system that operates a single task at a fixed frequency. In order to facilitate PWA approximation, we further simplify the closed-loop model by considering a continuous-time controller, while retaining the nonlinearities in the plant. Finally, we fix the inputs to the closed-loop model to constant values: 15° for the θ input, and 200 rad/sec for the ω input. The resulting closed-loop model is thus an autonomous model with four continuous states (p, λ, p_{est}, i) evolving according to differential equations with continuous, nonlinear dynamics.

3 Piecewise Affine Approximations

In this section, we present PWA approximations of the system dynamics described in Sec. 2. We first state the problem definition: Consider a nonlinear system of the form: $\dot{\mathbf{x}} = f(\mathbf{x}),\ \ \mathbf{x} \in \mathbb{R}^n$. The objective is to find a *good* PWA approximation, which can be defined as the set of PWA dynamical systems as follows: $\dot{\mathbf{x}} = \mathbf{A}_i\mathbf{x} + \mathbf{c}_i,\ \ \mathbf{x} \in R_i$. Here, $\mathbf{A}_i \in \mathbb{R}^{n\times n}$, $\mathbf{c}_i \in \mathbb{R}^n$, and $R_i \subseteq \mathbb{R}^n$. Furthermore, the sets R_i are disjoint, i.e., for any i, j, $R_i \cap R_j = \emptyset$[1]. A good PWA approximation is one in which the approximation error is acceptable. Formally, the approximation error e_i for region R_i is defined as: $e_i = \max_{\mathbf{x}\in R_i} \|f(\mathbf{x}) - (\mathbf{A}_i x + \mathbf{c}_i)\|$. Here $\|.\|$ denotes a norm in a suitable topological space, such as the L_2 norm over $\mathbb{R}^n$. Another metric of the utility of a given PWA approximation is the number of regions R_i. A high number is typically indicative of a more precise approximation; however, a high number is not always desirable, especially when performing reachability analysis with tools such as SpaceEx.

In what follows, we give a brief description of two methods that we used to benchmark the PWA approximation models generated using the PWATools package developed by Maplesoft for Toyota [5].

PWA approximations with the simplex-partitioning method. The objective of the simplex-partitioning approach is to compute a PWA approximation that is continuous and has a manageable number of regions (even for higher dimensional spaces). Here, we consider PWA approximations with polytopic regions that are simplices. A simplex is a polytope with $n+1$ vertices for a system with n dimensions. In a simplectic decomposition, an affine approximation over a given simplex R_i can be computed by equating the value of the affine expression to the value of the given nonlinear function at the $(n+1)$ vertices of the simplex. If all the local PWA approximations are computed in this fashion, the vector field across the partition boundaries is continuous, which is advantageous with regards to the reachable set estimation computations. Maplesoft's tools provide a number of simplex-partitioning algorithms; we chose the scheme known as *regular edge bisection*.

The goal of the regular edge bisection method is to produce a limited family of simplices that are not degenerate (e.g. having unit normal vectors whose inner product is close to 1)

[1]Note that in a hybrid automaton model of the PWA, a location (mode) is assigned to each R_i, whose invariant set is the closure of R_i and whose outgoing guard sets are defined on the boundary of the closure of R_i.

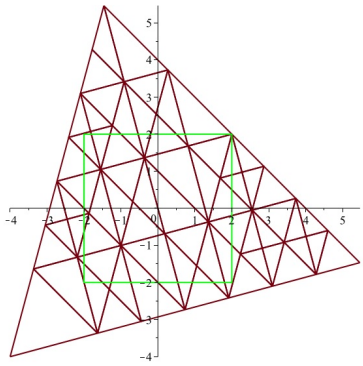


Figure 1: Regular edge bisection method for simplicial decomposition. Green lines denote the domain of interest, and the brown lines show the simplicial decomposition.

using the simplex bisection scheme by Maubach [13]. Fig. 1 depicts the result of applying the regular edge bisection procedure on a two-dimensional example.

PWA approximations with the nested method. The nested method computes the approximation to nonlinear vector functions by hierarchically approximating the functions according to the structure of their subexpressions, which can produce a smaller relative number of partition elements than other techniques. This technique can be useful for higher dimensional spaces (e.g., $n \geq 6$), where the number of partition elements required to achieve a desired error bound is often prohibitively high.

Essentially, this approach computes a PWA approximation of the nonlinear vector field function using nested univariate PWA functions. The method utilizes three key ideas: (1) PWA approximation of the sum of nonlinear functions is defined as the sum of PWA approximations of individual functions, (2) the PWA approximation of the composition of nonlinear functions is defined as the composition of PWA approximations of individual functions, and (3) multivariate nonlinear terms can sometimes be decomposed into a sum of univariate expressions by applying transformations and introducing new intermediate variables. This allows dividing the overall problem into several smaller problems over the domains of the embedded univariate nonlinear expressions. Basically, this approach follows the method introduced in [12] for deriving optimal PWA of nonlinear systems with known analytic form.

A natural way to think about the nested PWA model obtained by the nested method is to think of it as a composition of several components. If we represent each univariate subexpression as an intermediate variable, then the PWA approximation of this expression is a component whose input is the variable appearing in the expression, and the output is the intermediate variable. Global behavior can then be understood in terms of parallel composition of these components.

The simplex-partitioning approach neglects the structure of the function, seeking to construct an acceptable PWA approximation by strategically splitting the domain to minimize the number of regions required to achieve a desired accuracy. In contrast, the nested method does not limit the number of global regions, but exploits the structure of the function itself. In general, there is little control over the number of global polytopic partitions of the domain that may be formed by the nested method. As we decrease the tolerated error, the number of regions generated increases exponentially.

In spite of the high number of regions that the nested method may generate, reachability

State	RMS error	
	$M^{\text{PWA}}_{\text{simplex}}$	$M^{\text{PWA}}_{\text{nested}}$
p	0.22	0.04
λ	1.493	0.17
p_{est}	0.75	0.35
i	1.56	0.17

Table 1: Comparison of the simplex-partioning method and the nested methods.

Error	PWA regions/expression		
Tolerance	Min	Max	Average
0.1	1	2	1.1
0.01	2	5	2.5
0.001	4	15	7.1
0.0001	12	46	21.1

Table 2: Distribution of the num. of approximations across expressions for the nested PWA method.

analysis tools could still scale to nested PWA models, as long as the projection of the reachable sets on individual univariate subexpressions do not span multiple regions. The representation of the nested PWA model is also succinct as it avoids computing the partition imposed on the global state-space by the Cartesian product of the local PWA regions. Note that a tool such as SpaceEx that supports component-based modeling may be able to run on such a succinct representation directly. Translation to models in the SpaceEx format remains an important part of the future work. Another promising direction is to try on-the-fly generation of linear regions, i.e. a linear region is only generated when it is reached [11].

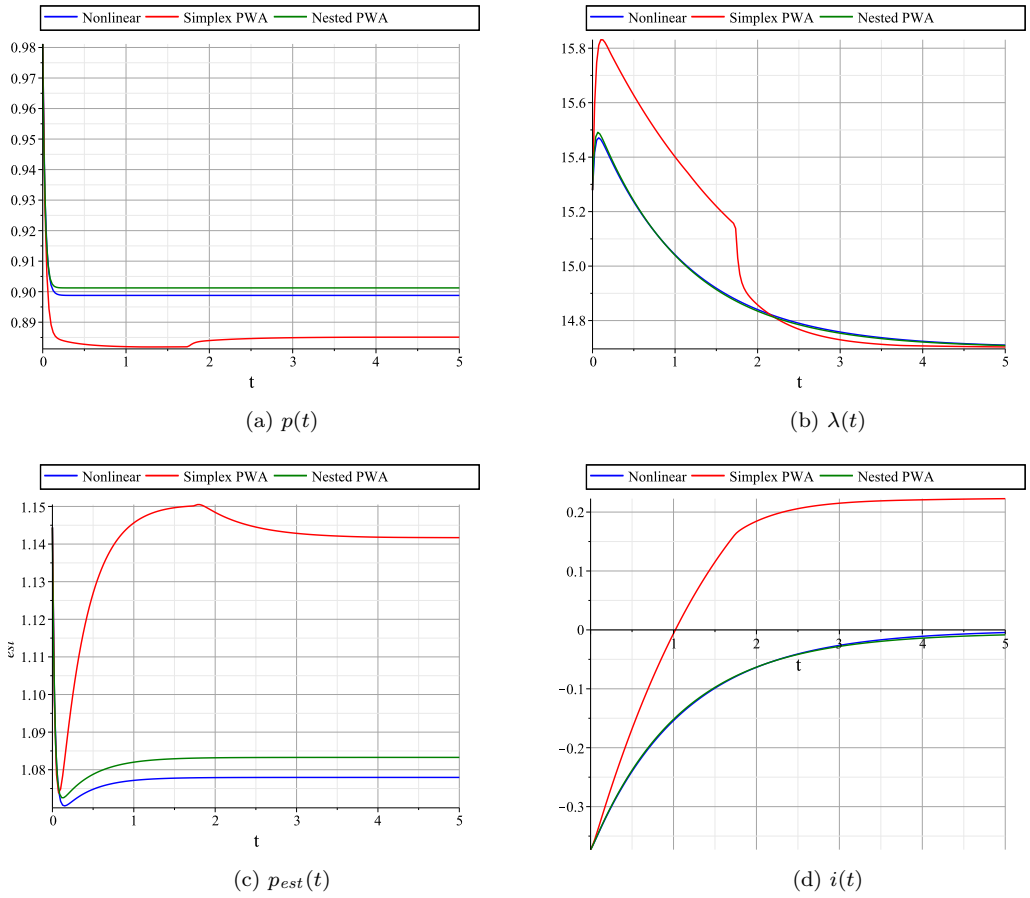
4 Experimental Evaluation and Conclusion

In this section, we evaluate the two PWA approximation methods. In the first experiment, we compare the results of simulating the three models: the original nonlinear model $M_{\text{nonlinear}}$, the PWA model obtained by simplex-partitioning method $M^{\text{PWA}}_{\text{simplex}}$, and the PWA model obtained by the nested method $M^{\text{PWA}}_{\text{nested}}$. All simulations reported in the sequel were performed using MapleSim. For the approximation from $M_{\text{nonlinear}}$ to $M^{\text{PWA}}_{\text{simplex}}$, there is one approximation parameter that the user controls: the maximum number of allowed PWA regions. For this experiment, we picked the number to be 500, and after the approximation was finished, the number of simplices generated was reported as 453. For the approximation from $M_{\text{nonlinear}}$ to $M^{\text{PWA}}_{\text{nested}}$, there are two approximation parameters that the user controls: the allowed error tolerance (per region associated with a subexpression), and the maximum number of PWA regions per nonlinear expression. For this experiment, we chose an error tolerance of 0.01, and the maximum number of regions to be 64. Upon termination of the approximation procedure, the maximum number of possible regions in the composed and flattened PWA model was $27,824$.[2]

To compare the two models, we randomly choose 10 initial conditions for each state variable from the following intervals: p: $[0.8, 1.0]$, λ: $[14.0, 15.4]$, p_{est}: $[0.9, 1.2]$, i: $[-0.5, 0.5]$. We then simulate the 3 models for a duration of 5.0 seconds, and measure the RMS error between the trajectories of models $M^{\text{PWA}}_{\text{simplex}}$ and $M^{\text{PWA}}_{\text{nested}}$ with the corresponding trajectories of the model $M_{\text{nonlinear}}$. We then compute the average RMS error across the 10 randomly chosen simulations, and report the results in Table 1. We show sample plots comparing the 4 states in Fig. 2.

As can be seen from Table 1, the nested method is more accurate than the simplex-partitioning method. The error in the states having transient behavior is worse for the simplex-partitioning method. We remark that for the simplex-partitioning based approximation with

[2]Note that many of the potential regions in the composed model are empty, but it is difficult to identify empty regions.



(a) $p(t)$ (b) $\lambda(t)$

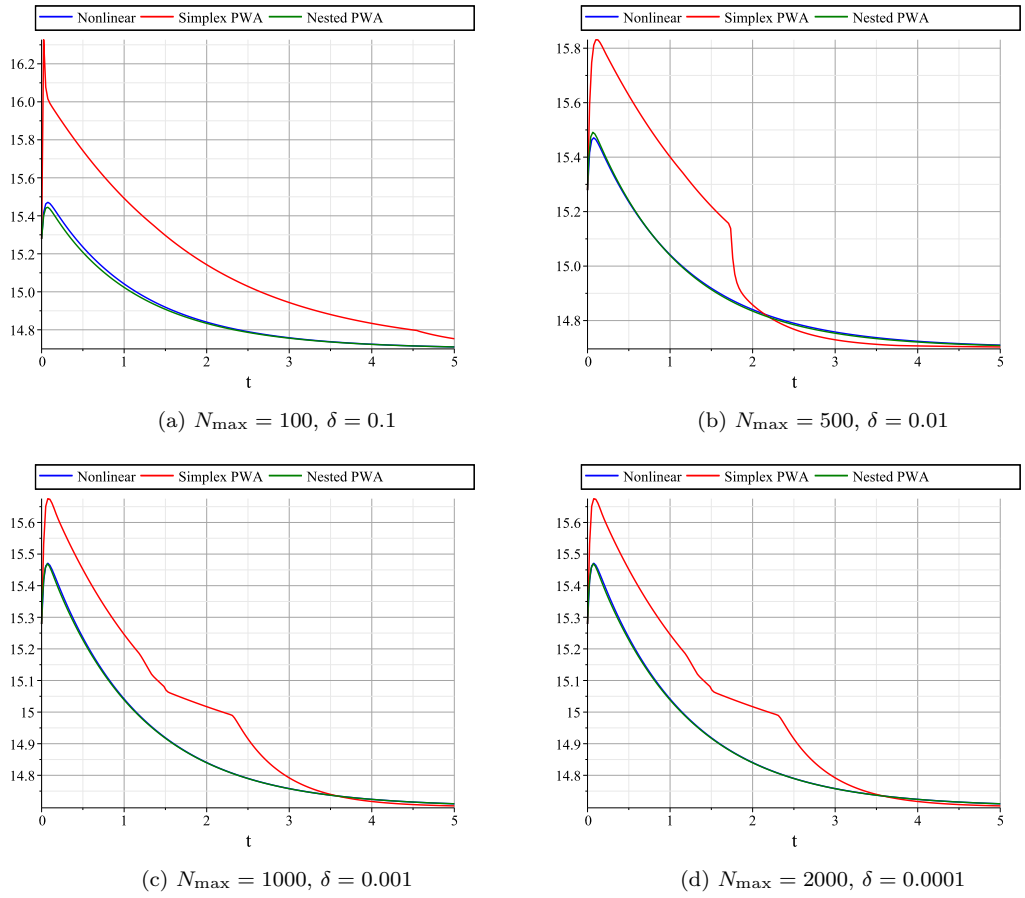
(c) $p_{est}(t)$ (d) $i(t)$

Figure 2: State trajectories comparison. For $M^{\text{PWA}}_{\text{simplex}}$, $N_{\max} = 500$, for $M^{\text{PWA}}_{\text{nested}}$, $\delta = 0.01$.

a maximum of 500 simplices allowed, we are able to finish the simulation starting from each of the 10 randomly chosen initial conditions. This is not true in general, as we often obtain an approximation such that the resulting PWA dynamics are unstable in some regions. This causes the state variables to grow exponentially in such a region. In these cases, the RMS error approaches ∞.

Next, we study the effect of increasing the maximum number of allowed simplices ($N_{\max}$) for the simplex-partitioning method, and decreasing the allowed error tolerance (δ) for the nested method. For the nested method, as δ is decreased, the average number of PWA regions required to approximate each nonlinear expression increases (shown in Table 2). We report the average RMS error on the state λ (A/F ratio) for each of the models in Table 3. For each model, we let N represent the final number of PWA regions generated by the method if its hierarchical structure were flattened. We show the effect of different levels of approximation on the trajectories of state λ in Fig. 3.

As expected, the RMS error decreases with increasing level of precision for both methods. The cost-to-benefit ratio, however, sharply declines for the simplex-partitioning method. As can be observed, the simulation time increases nearly 10x from 500 simplices to 2,000 simplices.



(a) $N_{\max} = 100$, $\delta = 0.1$

(b) $N_{\max} = 500$, $\delta = 0.01$

(c) $N_{\max} = 1000$, $\delta = 0.001$

(d) $N_{\max} = 2000$, $\delta = 0.0001$

Figure 3: Effect of using different levels of approximation in both methods.

The nested method scales much better with increasing demands on precision. Furthermore, even the least precise run of the nested method produces a better average RMS error than the most precise run of the simplex-partitioning based method. This is mainly due to the ability of the nested method to exploit formula structure. Unlike the simplex-partitioning method, the nested method is not constrained to have vector fields continuous across neighboring regions. Discontinuous approximations usually happen if the original model contains complex nonlinear subexpressions such as piecewise functions or mathematical functions of more than one variable (e.g., $\log(x+y)$. The freedom to have discontinuous approximations could potentially contribute to the superior accuracy exhibited by the nested method.

Acknowledgements. We thank the anonymous reviewers for their feedback.

		N	Approximation		Avg. Sim.	RMS error
			Time (secs)	Mem. (GB)	Time (secs)	for $\lambda(t)$
$M^{\text{PWA}}_{\text{simplex}}$	$N_{\max} = 100$	99	27	1.8	9.7	∞
	$N_{\max} = 500$	453	117	7.7	40.2	1.493
	$N_{\max} = 1000$	990	260	16.8	128	0.97
	$N_{\max} = 2000$	1968	535	33.4	423	0.96
$M^{\text{PWA}}_{\text{nested}}$	$\delta = 0.1$	2	1.2	0.017	2.09	0.18
	$\delta = 0.01$	$2.8\text{x}10^{4}$	2.1	0.074	2.42	0.17
	$\delta = 0.001$	$1.5\text{x}10^{10}$	5.9	0.322	3.27	0.06
	$\delta = 0.0001$	$1.8\text{x}10^{15}$	31	1.75	9.46	0.04

Table 3: Effect of increasing precision for the simplex-partitioning and the nested PWA approximation methods.

References

[1] M. Althoff, O. Stursberg, and M. Buss. Computing reachable sets of hybrid systems using a combination of zonotopes and polytopes, 2009.

[2] M. Z. Fekri, B. Samadi, and L. Rodrigues. Pwatools: A matlab toolbox for piecewise-affine controller synthesis. In *American Control Conference (ACC), 2012*, pages 4484–4489. IEEE, 2012.

[3] G. Frehse. Phaver: algorithmic verification of hybrid systems past hytech. *STTT*, 10(3):263–279, 2008.

[4] G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. Spaceex: Scalable verification of hybrid systems. In *CAV*, 2011.

[5] J. Gerhard, E. Postma, B. Samadi, J. Stewart, K. Walker, and Y. Xie. Reports on symbolic control: Toyota internal reports. Technical report, 2013–2015.

[6] T. A. Henzinger, P.-H. Ho, and H. Wong-toi. Hytech: A model checker for hybrid systems. *Software Tools for Technology Transfer*, 1:460–463, 1997.

[7] X. Jin, J. Deshmukh, J. Kapinski, K. Ueda, and K. Butts. Benchmarks for model transformations and conformance checking. In *Proc. 1st international workshop on Applied Verification for Continuous and Hybrid Systems (ARCH)*, 2014.

[8] X. Jin, J. V. Deshmukh, J. Kapinski, K. Ueda, and K. Butts. Powertrain control verification benchmark. In *HSCC*, pages 253–262, 2014.

[9] K. G. Larsen, P. Pettersson, and W. Yi. Uppaal in a Nutshell. *Int. Journal on Software Tools for Technology Transfer*, 1(1–2):134–152, Oct. 1997.

[10] C. Le Guernic and A. Girard. Reachability analysis of hybrid systems using support functions. In *CAV*, pages 540–554. Springer, 2009.

[11] H.-S. L. Lee, M. Althoff, S. Hoelldampf, M. Olbrich, and E. Barke. Automated generation of hybrid system models for reachability analysis of nonlinear analog circuits. In *Proc. of the Asia and South Pacific Design Automation Conference*, pages 725–730, 2015.

[12] A. S. M. Kvasnica and M. Fikar. Automatic derivation of optimal piecewise affine approximations of nonlinear systems. In A. C. S. Bittanti and S. Zampieri, editors, *IFAC World Congress, Volume #18, Part #1*, pages 8675–8680, Universita Cattolica del Sacro Cuore, Milano, Italy, 2011.

[13] J. M. Maubach. Local bisection refinement for n-simplicial grids generated by reflection. *SIAM J. on Scientific Computing*, 16(1):210–227, 1995.

[14] A. Szücs, M. Kvasnica, and M. Fikar. Matlab toolbox for automatic approximation of nonlinear functions. *Proceedings of the Process Control*, 2011.

[15] S. Yovine. Kronos: A verification tool for real-time systems. *Int. Journal on Software Tools for Technology Transfer*, 1:123–133, 1997.

A Appendix

A.1 Nonlinear Model Equations

In this section, we outline the equations for the model described in Sec. 2.

The function encoding the geometry of the throttle is given as a function of the throttle angle θ:

$$\hat{\theta} \quad = \quad c_6 + c_7\theta + c_8\theta^2 + c_9\theta^3. \tag{1}$$

The inlet air mass flow rate $\dot{m}_{af}$ is then given by the product of the above function, and a function encoding a physical phenomenon relating the atmospheric pressure (c_{10}) to the intake manifold pressure p:

$$\dot{m}_{af} \quad = \quad 2\hat{\theta}\sqrt{\frac{p}{c_{10}} - \left(\frac{p}{c_{10}}\right)^2}. \tag{2}$$

The pumping polynomial is a function of the engine speed ω (in rad/sec) and the intake manifold p:

$$\dot{m}_c \quad = \quad c_{12}\left(c_2 + c_3\omega p + c_4\omega p^2 + c_5\omega^2 p\right). \tag{3}$$

Finally, the ODE for the intake manifold pressure is described as follows:

$$\frac{dp}{dt} = c_1\left(2\hat{\theta}\sqrt{\frac{p}{c_{10}} - \left(\frac{p}{c_{10}}\right)^2} - c_{12}\left(c_2 + c_3\omega p + c_4\omega p^2 + c_5\omega^2 p\right)\right). \tag{4}$$

The ODE governing the measured A/F ratio λ is given below.

$$\frac{d\lambda}{dt} = c_{26}\left(\frac{\dot{m}_c}{c_{25}F_c} - \lambda\right). \tag{5}$$

The state equation for the manifold pressure estimator is as given below.

$$\frac{dp_{est}}{dt} = c_1 \cdot \left(c_{23}\hat{\dot{m}}_{af} - \hat{\dot{m}}_c\right) \tag{6}$$

In the above equation, $\hat{\dot{m}}_c$ denotes the estimated air entering the cylinder, as computed from the estimated intake manifold pressure. This quantity is given by the following equation:

$$\hat{\dot{m}}_c \quad = \quad c_{27} \cdot \left(c_2 + c_3\omega p_{est} + c_4\omega p_{est}^2 + c_5\omega^2 p_{est}\right). \tag{7}$$

The feedback PI controller update equation is given by

$$\frac{di}{dt} \quad = \quad c_{14}(c_{24}\lambda - c_{11}). \tag{8}$$

The fuel command (F_c) output by the controller is given by:

$$F_c \quad = \quad \frac{1}{c_{11}} \cdot (1 + c_{13}(c_{24}\lambda - c_{11}) + i) \cdot \hat{\dot{m}}_c \tag{9}$$

Table 4 provides a list of the model parameter values.

Table 4: Model Parameters.

Param	Value	Unit	Description
c_1	0.41328	RT/Vm	
c_2	-0.366		Coefficient for *Pumping* polynomial
c_3	0.08979		Coefficient for *Pumping* polynomial
c_4	-0.0337		Coefficient for *Pumping* polynomial
c_5	0.0001		Coefficient for *Pumping* polynomial
c_6	2.821		Coefficient for $f(\theta)$ polynomial
c_7	-0.05231		Coefficient for $f(\theta)$ polynomial
c_8	0.10299		Coefficient for $f(\theta)$ polynomial
c_9	-0.00063		Coefficient for $f(\theta)$ polynomial
c_{10}	1.0	bar	Atmospheric pressure
c_{11}	14.7/12.5		Desired air-fuel ratio (all other modes / `power` mode)
c_{12}	0.9		Manifold pressure estimate error factor
c_{13}	0.05		Proportional gain for PI controller
c_{14}	0.03		Integral gain for PI controller
c_{23}	1.0		MAF sensor constant error factor
c_{24}	1.0		Oxygen sensor constant error factor
c_{25}	1.0		Fuel injector actuator error factor
c_{26}	4.0		First-order transfer function constant
c_{27}	0.9		Observer error factor
u_1		degrees	Throttle angle
u_2		rad/sec	Engine speed

A.2 Details on the PWA approximation methods

The following Lorenz attractor example is used to illustrate how the methods approach the approximation problem at a system level and the structure of the approximated models.

Example application: the Lorenz attractor. The Lorenz attractor is given by:

$$\begin{aligned}\dot{x}_1 =& \sigma(x_2 - x_1)\\ \dot{x}_2 =& \rho x_1 - x_2 - x_1 x_3\\ \dot{x}_3 =& -\beta x_3 + x_1 x_2,\end{aligned} \tag{10}$$

where $\sigma = 10$, $\rho = 28$ and $\beta = \frac{8}{3}$. The domain considered is

$$\mathcal{X} = \{(x_1, x_2, x_3) | -25 \leq x_1 \leq 25, -25 \leq x_2 \leq 25, 0 \leq x_3 \leq 50\}. \tag{11}$$

The simplex-partition method approximates the system (10) by grouping the right hand sides of the differential equations as the vector field f:

$$f = \begin{bmatrix} 10(x_2 - x_1) \\ 28x_1 - x_2 - x_1 x_3 \\ -2.67x_3 + x_1 x_2 \end{bmatrix} \tag{12}$$

Only the nonlinear portion of (12) needs to be approximated, so f is partitioned into its linear (f_{lin}) and non-linear (f_{nonlin}) components as follows:

$$f = f_{lin} + f_{nonlin}, \tag{13}$$

$$f_{lin} = \begin{bmatrix} 10(x_2 - x_1) \\ 28x_1 - x_2 \\ -2.67x_3 \end{bmatrix} \tag{14}$$

$$f_{nonlin} = \begin{bmatrix} 0 \\ -x_1 x_3 \\ x_1 x_2 \end{bmatrix}. \tag{15}$$

The simplex-partition approach approximates f_{nonlin} as $\tilde{f}_{nonlin}$ so that the approximation to the original vector field is then $\tilde{f} = f_{lin} + \tilde{f}_{nonlin}$. The nonlinear portion f_{nonlin} is a function of three variables and will be approximated as such, even though the structure of the system only has products of two variables. Using 1,000 regions for the vector field's approximation, the resulting expression for $\tilde{f}_{nonlin}$ is far too large for inclusion here. It consists of nested binary piecewise functions whose branch values and switching conditions are affine expressions in the model's state variables.

The nested PWA approach first processes the model to extract the nonlinear terms into separate equations. The multivariate nonlinear expressions are broken down into sums of univariate nonlinear expression and then many lower dimensional approximations are computed, as opposed to the simpex-partition's approach of computing one high dimensional approximation. For the Lorenz system, the model is transformed into

$$\begin{aligned} z_1 &= -x_1 x_3 \\ z_2 &= x_1 x_2 \\ \dot{x}_1 &= 10x_2 - 10x_1 \\ \dot{x}_2 &= 28x_1 - x_2 + z_1 \\ \dot{x}_3 &= -2.67x_3 + z_2. \end{aligned} \tag{16}$$

There are four piecewise functions with affine switching conditions and two branches each, giving 16 regions in the domain $\mathcal{X}$.

The initial conditions of $x_1(0) = 10$, $x_2(0) = 10$ and $x_3(0) = 10$ gives the simulation results in Figure 4 for the original nonlinear system, the simplex-partition approximation and the nested PWA approximation. Even though the nested PWA model has only 16 regions compared to the 1,000 regions of the simplex-partition model, the nested results in Figure 6 better reproduce the behaviour of the original system in Figure 4, as the simplex results in Figure 5 settles to an equilibrium point while the other systems continue to oscillate.

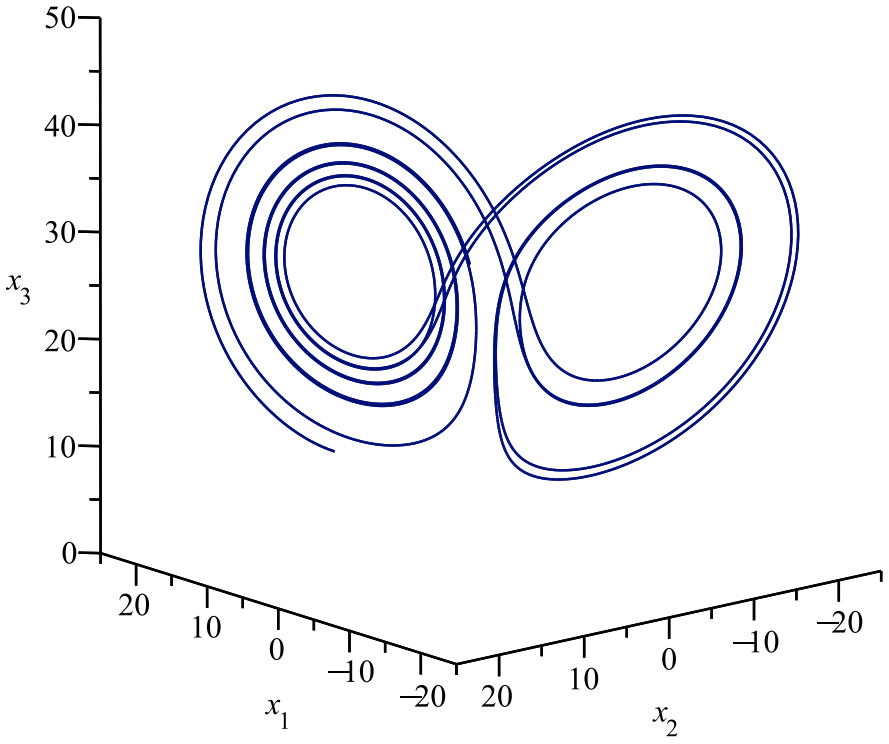


Figure 4: Simulation of the exact nonlinear Lorenz attractor.

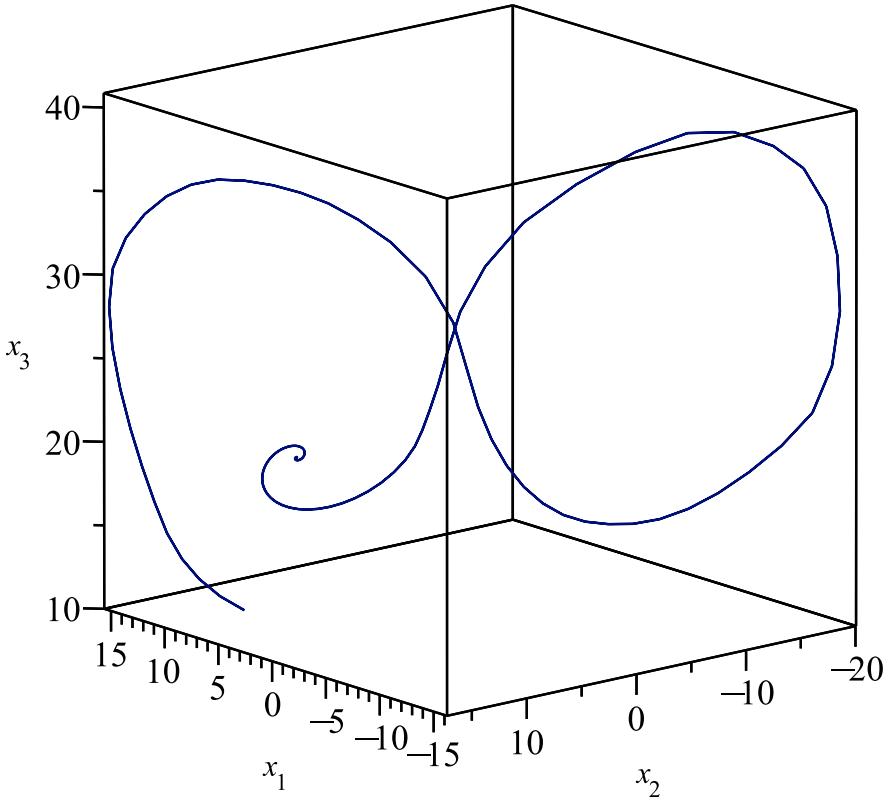


Figure 5: Simulation of the simplex-partition approximation to the Lorenz attractor.

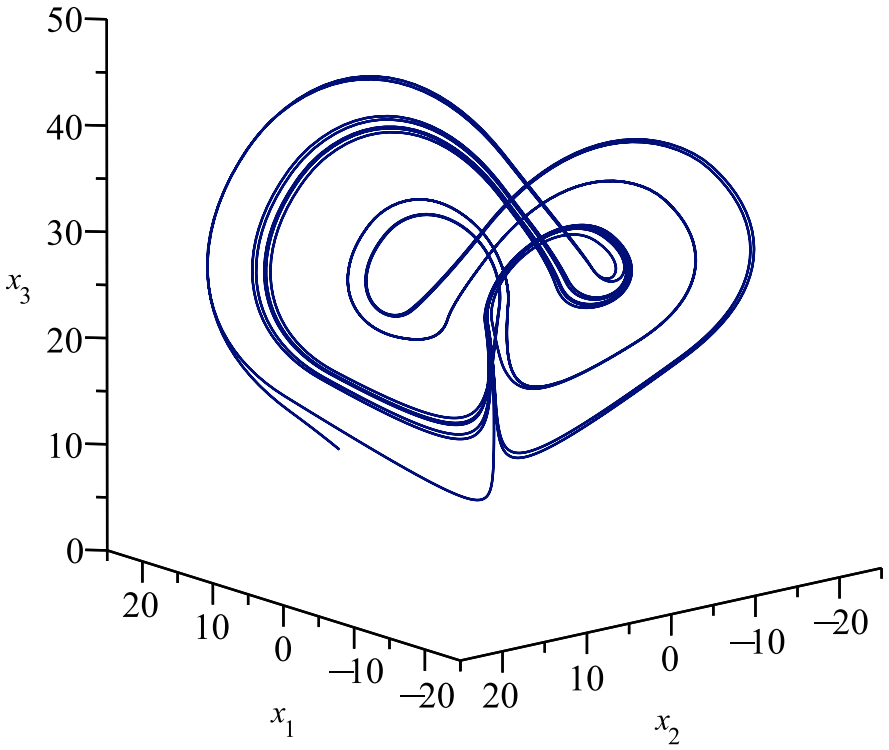


Figure 6: Simulation of the nested PWA approximation to the Lorenz attractor.