Modelling and Simulation of Hybrid Systems with Neural Networks

Stefanie Winkler * **Martin Bicher** ** **Andreas Körner** *
Felix Breitenecker *

* *TU Wien, Institute for Analysis and Scientific Computing, Vienna, Austria (e-mail: {stefanie.winkler, andreas.koerner, felix.breitenecker}@tuwien.ac.at)*
** *dwh GmbH, Vienna, Austria (e-mail: martin.bicher@dwh.at)*

Keywords: Hybrid Modelling, Neural Network, Mathematical Modelling, Simulation Comparison, Data Modelling

1. INTRODUCTION

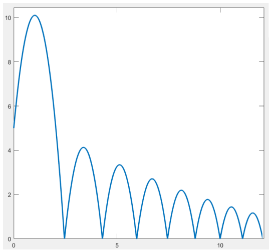
Most processes in industry as well as in nature can be rarely described with one simple model. Therefore various modelling methods established in the last years deal with implementations of complex structures. Two of this methods are topic of this contribution. Due to increasing availability of data from multiple resources research in the field of neural networks increased exponentially. Neural networks are used to imitate the human brain and enable algorithms to make their own decisions. Also in the industrial sector the importance of data increases. Therefore the research field big data is also important in current industrial research projects. Urban infrastructure is one example: Cars driving on their own, gathering information while driving to make decisions on human behalf. But it would be careless to use only data-based models for simulation of complex processes involving heavy machines. Therefore first principle models are still important and the base of modelling and simulation. In this contribution, a comparison of these controversial approaches is discussed.

2. HYBRID DYNAMIC SYSTEMS

The term hybrid is often mentioned in connection with the auto mobile industry. Apart from that there are many different areas where hybrid is used and in all these cases the wording stands for a combination of different methods or approaches respectively. In terms of mathematical modelling hybrid defines a combination of multiple modelling approaches in one model. This contribution focuses on hybrid dynamic models which consist of different discrete submodels as well as continuous structures to implement real life scenarios. Switching from one submodel to another, state variables or even underlying mathematical descriptions change at discrete points in time, called events. In order to describe such hybrid systems different formalisms were introduced over the last decade. The usage of automaton, as in Körner (2016) is very common because it gives important information about the model structure. A rough description of the submodels as well as the jump conditions for switching can be included. This formalism focuses on the mathematical modelling and the corresponding mathematical definitions. A common alternative and more simulation driven formalism would be DEVS& DESS. This formalism was first introduced by Ziegler et al. (2000) and was implemented in MATLAB by Deactu et. al (2012). This formalism started with Discrete Event System Specification (DEVS) and was later extended to include dynamic processes (DESS). Furthermore linear affine systems are a widely used method to overcome challenges of hybrid system simulation as well. Especially in the field of control this approach is used to implement hybrid system structures, as described in Potočnik et al. (2010).

3. NEURAL NETWORKS

Artificial neural networks are nowadays a commonly used method, especially in the field of computer learning. In general neural networks are based on the biological nerve structure of human brains. The artificial neural network imitates the reaction chain of a human neural network. The basic structure consists of three layers: the input, the hidden and the output layer. As the names suggest the first and the latter function as input and output nodes. The most interesting one is the hidden layer which contains a specific activation function to process the incoming signal. All three layers are connected by a pattern of edges. Each of these edges carries a certain weight which amplifies or damps the incoming signal before sending it to the next node. In general a neural network gets an input and transforms it according to the weights and activation function until the final output is generated. The remaining question is how to determine these weights and activation functions. In literature one can find different possible definitions from step and linear functions via logistic functions through to sigmoid functions as activation function. This decision depends on the application field of the artificial neural network. The weights of the edges can only be specified using training data consisting of input and corresponding output data. There are different training options for neural networks but a very common method is the back propagation. This means that the available data sets are partly used to tune the weights until the error of the neural network output to the data set output is small enough.



$$\begin{aligned} \ddot{h} &= -g \\ h(0) &= h_0 \\ \dot{h}(0) &= v_0\ . \end{aligned} \tag{1}$$

Fig. 1. The equations and graph of a bouncing ball is given.

4. CASE STUDY: BOUNCING BALL

4.1 Model Description

In the following the bouncing ball, an academic example of hybrid systems, is discussed. Considering a bouncing ball, the question might be which part of this process defines it as hybrid. As mentioned in section 2 hybrid models combine continuous and discrete processes. Regarding the bouncing ball the bounce itself represents the discrete part of the model. The bounce only occurs for a single point in time where two things happen: the ball changes its direction and additionally, to take note of the underlying physical damping process, decreases its total velocity. This process can be described mathematically with height h and velocity v. The relation of height and velocity is of course $v = \dot{h}$ and for the acceleration the relation $a = \dot{v} = \ddot{h}$ is valid. To realise the discrete event of the bounce the acceleration has to face opposite direction of gravity. Therefore the model behaviour can be given as an ODE of second order with initial conditions as shown in equation (1).

4.2 Modelling and Simulation

Equation (1) can be transformed into an ODE system applying basic transformations and therefore also be formulated as state space description. As mentioned in the model description the discrete event is defined as the moment when the ball touches the ground. If the following condition, further called jump condition, is fulfilled

$$\left\{(h(t), \dot{h}(t)) \colon h(t) = 0 \wedge \dot{h}(t) = v(t) \leq 0\right\}$$

the event is located and the ball's direction changes as defined in (2). The hybrid model description then consists of the state space description, the jump (2) and the jump condition.

$$J(v(t_e)) = -\lambda v(t_e),\ \lambda \in (0,1) \tag{2}$$

In terms of neural networks hybrid is also a known term and describes a mixture of first principle models and neural networks, as seen in Psichogios et. al (1992). Due to the fact that the bounding ball can be solved analytically, as shown below, training data for the neural network can be generated. Changing the initial condition constants c_1 and c_2 varies and multiple data sets can be created.

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -\frac{g}{2}t^2 + c_2 t + c_1 \\ -gt + c_2 \end{pmatrix}$$

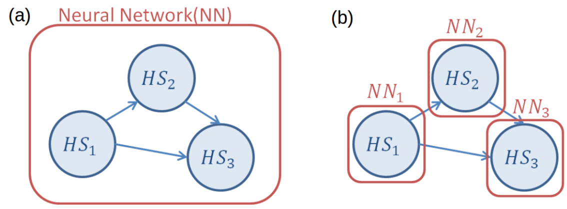


Fig. 2. Different possibilities to apply neural networks for hybrid systems.

5. DISCUSSION

Which data should be used for the input and output set? Are the initial conditions as input sufficient or is it necessary to include the time line as well. Is one hidden layer enough and which certain structure should be used. How many neural networks are necessary to simulate the bouncing ball. Is one neural network with several hidden layers sufficient, as in Fig. 2(a) or is it necessary to establish a different neural networks each simulating one flying phase of the ball, as seen in Fig. 2(b). The latter could be seen as a hybrid model using neural networks as submodels. Another possibility would be to use the mentioned approach of hybrid neural networks. The intuitive implementation takes initial values and timeline in consideration using a fully connected neural network. This of course might not be the best choice.

6. OUTLOOK

This contribution discusses the usage of neuronal networks for simulating hybrid systems in the field of engineering. The chosen case study, the bouncing ball, provides all necessary data to implement and train a neural network. Implementing the different possible structures of neural networks and compare them with common Simulink© and MATLAB© realisations determines if neural networks are useful for simulating hybrid systems. In most cases, the actual hybrid model can not be given in detail, then the gained experiences of this comparison may be applied. Another improvement of performance might be the possibility to use hybrid neural network. Therefore a more detailed analysis of these different approaches is necessary.

REFERENCES

A. Körner, S. Winkler and F. Breitenecker *Possibilities in State Event Modelling of Hybrid Systems* 9th EUROSIM Congress on Modelling and Simulation, Oulu, 2016.

B. P. Zeigler, H. Praehofer and T. G.Kim. E*Theory of Modeling and Simulation, 2nd Edition* academic Press, 2000.

B. Potočnik , G. Mušič and B. Zupančič. A New Technique for Translating Discrete Hybrid Automata into Piecewise Affine Systems *Mathematical and Computer Modelling of Dynamical Systems* pages 41-57, February 2010.

C. Deatcu and T. Pawletta. A Qualitative Comparison of Two Hybrid DEVS Approaches *SNE Simulation Notes Europe vol. 22(1)*, pages 15-24, Vienna, 2012.

D. C. Psichogios and L. H. Ungar. A Hybrid Neural Network-First Principles Approach to Process Modeling *AIChE Journal vol. 38, No. 10*, pages 1499-1511, 1992.