A Neuro-Fuzzy Development Tool for Fuzzy Controllers under MATLAB/SIMULINK

Andreas Nürnberger, Detlef Nauck and Rudolf Kruse
Department of Computer Science, University of Magdeburg
Institute for Information and Communication Systems, Neural and Fuzzy Systems
Universitaetsplatz 2, D-39106 Magdeburg, Germany
Phone: +49.391.67.11358, Fax: +49.391.67.12018
E-Mail: a.nuernberger@iik.cs.uni-magdeburg.de
URL: http://fuzzy.cs.uni-magdeburg.de/~nuernb

Abstract: The design and optimization process of fuzzy controllers can be supported by learning techniques derived from neural networks. Such approaches are usually called neuro-fuzzy systems. In this paper we describe an updated version of the neuro-fuzzy model NEFCON. This model is able to learn and optimize the rulebase of a Mamdani like fuzzy controller online by a reinforcement learning algorithm that uses a fuzzy error measure. Therefore we also describe some methods to determine a fuzzy error measure of a dynamic system. In addition we present an implementation of the model and an application example under the MATLAB/SIMULINK development environment. The implementation uses a graphical user interface to control the learning process interactively. The optimized fuzzy controller can be detached from the development environment and can be used in real time environments. The tool is available free of charge for non-commercial purposes (http://fuzzy.cs.uni-magdeburg.de/nefcon).

1 Introduction

The main problems in fuzzy controller design are the construction of an initial rulebase and in particular the optimization of an existing rulebase. The methods presented in this paper have been developed to support the user in both of these cases.

One of the main objectives of our project is to develop algorithms that are able to determine online an appropriate and interpretable rulebase within a small number of simulation runs. Besides it must be possible to use prior knowledge to initialize the learning process. This is a contrast to 'pure' reinforcement strategies [BARTO et. al., 1983] or methods based on dynamic programming [BARTO et. al., 1995; RIEDMILLER and JANUSZ, 1995] which try to find an optimal solution using neural network structures. These methods need many runs to find even an approximate solution for a given control problem. On the other hand, they have the advantage of using less information about the error of the current system state. However, in many cases a simple error description can be achieved with little effort. In this paper we present some methods to determine a fuzzy error measure of a dynamic system.

The first prototype implementation of the described algorithms and the development of a user friendly interface under MATLAB/SIMULINK[1] was done in cooperation with the Daimler-Benz Aerospace Airbus GmbH, Hamburg [NÜRNBERGER, 1996]. The tool can be obtained free of charge for non-commercial purposes via our Internet Web-Server (http://fuzzy.cs.uni-magdeburg.de/nefcon or ftp://fuzzy.cs.uni-magdeburg.de/nefcon/nefconma).

The NEFCON-Model

The NEFCON-Model is based on a generic fuzzy perceptron [NAUCK, 1994; NAUCK et. al., 1996]. An example, which describes the structure of a fuzzy controller with 5 rules, 2 inputs, and one output is shown in Fig. 1. The inner nodes R_1, …, R_5 represent the rules, the nodes ξ_1, ξ_1, and η the input and output values, and $\mu_r^{(i)}$, ν_r the fuzzy sets describing the antecedents $A_r^{(i)}$ and conclusions B_r. Rules with the same antecedents use so-called shared weights, which are represented by ellipses (see Fig. 1). They ensure the integrity of the rulebase. The node R_1 for example represents the rule: R_1: if ξ_1 is $A_1^{(1)}$ and ξ_2 is $A_1^{(2)}$ then η is B_1.

[1] *Remark: MATLAB/SIMULINK is a simulation tool developed and distributed by 'The Mathworks' Inc., 24 Prime Park Way, Natick, Mass.01760; WWW: http://www.mathworks.com.*

2 The Learning Algorithms

The learning process of the NEFCON model can be divided into two main phases. The first phases is designed to learn an initial rulebase, if no prior knowledge of the system is available. Furthermore it can be used to complete a manually defined rulebase. The second phase optimizes the rules by shifting or modifying the fuzzy sets of the rules. Both phases use a fuzzy error E, which describes the quality of the current system state, to learn or to optimize the rulebase. The fuzzy error plays the role of the critic element in reinforcement learning models (e.g. [BARTO et. al., 1983; BARTO, 1992]). In addition the sign of the optimal output value η_{opt} must be known. So the extended fuzzy error E* is defined as

$$E^*(x_1, ..., x_n) = \mathrm{sgn}(\eta_{opt}) \cdot E(x_1, ..., x_n),$$

with the crisp input $(x_1, ..., x_n)$.

The updated NEFCON learning algorithm learns and optimizes the rulebase of a Mamdani like fuzzy controller [MAMDANI and ASSILIAN, 1975]. The fuzzy sets of the antecedents and conclusions can be represented by any symmetric membership function. Triangular, trapezoidal, and Gaussian functions are supported by the presented implementation.

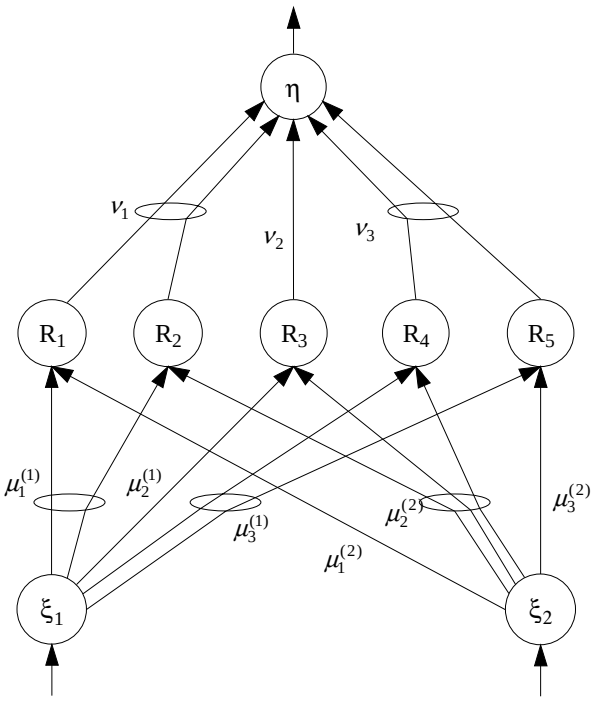

Fig. 1: A NEFCON System with two inputs, 5 rules and one output

2.1 Learning a Rulebase

Methods to learn an initial rulebase can be divided into three classes: Methods starting with an empty rulebase [NAUCK et. al., 1995; TSCHICHOLD-GÜRMAN, 1995], starting with a 'full' rulebase (combination of every fuzzy set in the antecedents with every conclusion) [NAUCK and KRUSE, 1993] and methods starting with a random rulebase [LIN, 1994]. We implemented algorithms of the first two classes.

Modified Algorithm NEFCON I

This algorithm is based on the original NEFCON model [NAUCK and KRUSE, 1993; NAUCK et. al., 1996]. It starts with a 'full' rulebase. The algorithm can be divided into two phases which are executed during a fixed period of time or a fixed number of iteration steps. During the first phase, rules with an output sign different from that of the optimal output value η_{opt} are removed. During the second phase, a rulebase is constructed for each control action by selecting randomly one rule from every group of rules with identical antecedents. The error of each rule (the output error of the whole network weighted by the activation of the individual rule) is accumulated. At the end of the second phase from each group of rule nodes with identical antecedents the rule with the least error value remains in the rulebase. All other rule nodes are deleted. In addition rules used very rarely are removed from the rulebase. The original algorithm used triangular membership functions. The improved implementation also supports trapezoidal and Gaussian membership functions. Besides the algorithm was enhanced for dynamic systems which need a static offset.

The 'Bottom-Up'-Algorithm

The 'Bottom-Up'-Algorithm starts with an empty rulebase. An initial fuzzy partitioning of the input and output intervals must be given. The algorithm can be divided into two phases. During the first phase, the rules' antecedents are determined by classifying the input values, i.e. finding that membership function for each variable that yields the highest membership value for the respective input value. Then the algorithm tries to 'guess' the output value by deriving it from the current fuzzy error. During the second phase the rulebase is optimized by changing the conclusion to an adjacent membership function, if this is necessary. The improved implementation supports trapezoidal and Gaussian membership functions, too.

The 'Bottom-Up'-Algorithm is much faster than NEFCON I in case of a large number of input variables and a fine initial fuzzy partitioning. This is caused by the huge initial rulebase used by the NEFCON I algorithm. Nevertheless the 'Bottom-Up'-Algorithm should not be used for complex dynamic systems up to now, because of the heuristic approach of finding the conclusions.

2.2 Optimization of a Rulebase

The presented algorithms are designed to optimize a rulebase of a fuzzy controller by shifting and/or modifying the support of the fuzzy sets. They do not modify the rules or the structure of the given network.

The Algorithm NEFCON I

The algorithm NEFCON I [NAUCK et. al., 1996] is motivated by the backpropagation algorithm for the multilayer perceptron. The extended fuzzy error E^* is used to optimize the rulebase by 'reward and punishment'. A rule is 'rewarded' by shifting its conclusion to a higher value and by widening the support of the antecedents, if its current output has the same sign as the optimal output η_{opt}. Otherwise the rule is 'punished' by shifting its conclusion to a lower value and by reducing the support of the antecedents. The original model used monotonic membership functions [TSUKAMOTO, 1979] in the conclusions to make it possible to use an backpropagation algorithm. In the current implementation this restriction was resolved by storing the activation of every rule during the inference mechanism. Because of that it is possible to use symmetric fuzzy sets in the conclusions and the antecedents.

The Algorithm NEFCON II

In contrast to the algorithm NEFCON I, which uses only the current fuzzy error E^*, the algorithm NEFCON II also makes use of the change of the fuzzy error E^* to optimize the rulebase [NAUCK et. al., 1995]. This is a heuristic approach to include the dynamic of the system into the optimization process. Let E^* be the extended fuzzy error at time t and $E^{*'}$ the extended fuzzy error at time t+1, then the error tendency τ is defined as

$$\tau = \begin{cases} 1 & \textit{if} \quad (|E^{*'}| \geq |E^*|) \wedge (E^{*'} \cdot E^* \geq 0) \\ 0 & \textit{if} \quad (|E^{*'}| < |E^*|) \wedge (E^{*'} \cdot E^* \geq 0) \\ -1 & \textit{otherwise} \end{cases}$$

If $\tau = 0$, the system moves to an optimal state. In this case the rulebase will not be modified. If $\tau = 1$, the error is rising without changing its sign. The output of each rule is increased by shifting its conclusions. The antecedents of rules with conclusions increasing the output will be 'rewarded', while with conclusions decreasing the output 'punished'. If $\tau = -1$, the system has overshot. The output is decreased and the antecedents 'punished' or 'rewarded', accordingly.

2.3 Description of System Error

In case of simple dynamic systems the error can be described sufficiently well by simply using the difference between the reference signal and the system response. In case of more complex and sensitive systems the error must be described more exactly to obtain a satisfying rulebase with the presented algorithms.

A Linguistic Error Description

The optimal state of a dynamic system can be described by a vector of system state variable values. Usually the state can not be described exactly, or we are content, if the system variables have roughly taken these values. Thus the quality of a current state can be described by fuzzy rules. With an error definition that uses a linguistic error description with fuzzy rules it is also easily possible to describe compensatory situations [NAUCK et. al., 1996]. These are situations in which the dynamic system is driven towards its optimal state. In Fig. 2 an error description is shown, which is part of the implementation presented in 3.1. This rulebase also describes an overshoot situation (rules 7 and 8).

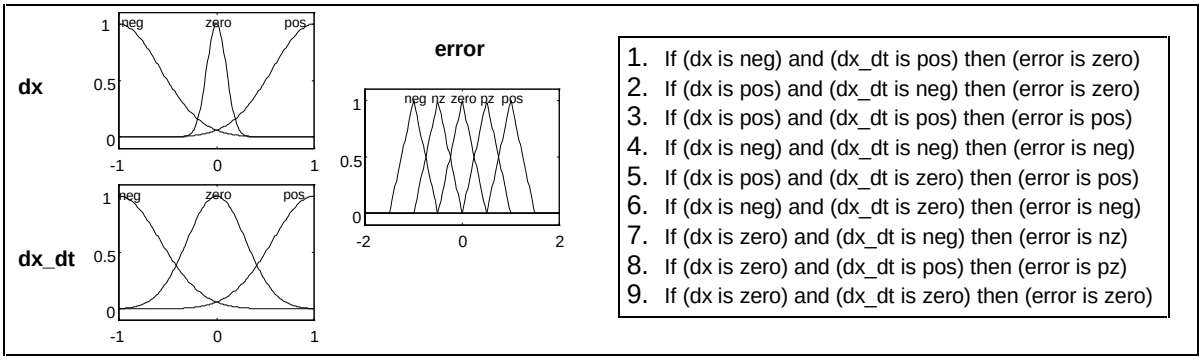

Fig. 2: Sample Rulebase for Fuzzy Error Description

An Error Description with 'Fuzzy Intervals'

The error description with 'fuzzy intervals' has been developed for the presented implementation. It makes it possible to describe a 'soft' region for the system response which satisfies our request to the system behavior in a simple and intuitive way. Fig. 3 presents an error description for a simple switch signal. The error signal remains zero, if the response signal of the dynamic system stays in the defined interval between the reference signal and the bounds (thick lines). If the signal leaves the bounds of the interval, the fuzzy error is determined using a linguistic error definition as described above.

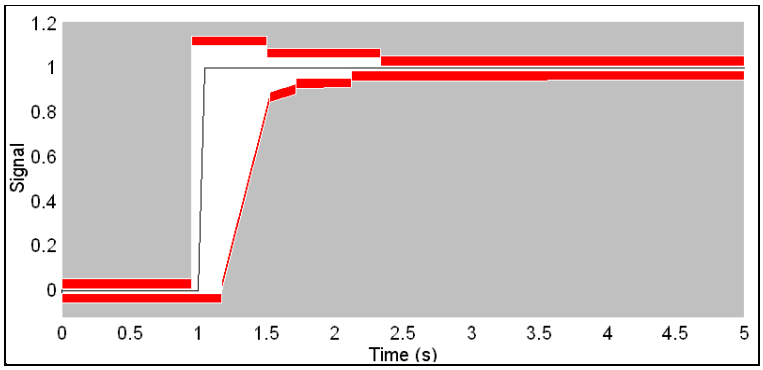

Fig. 3: Sample of an Error Description using 'Fuzzy Intervals'

3 Implementation

The aim of the implementation under MATLAB/SIMULINK was to develop an interactive tool for the construction and optimization of a fuzzy controller. This frees the user of programming and supports him in concentrating on controller design. It is possible to include prior knowledge into the system, to stop and to resume the learning process at any time, and to modify the rulebase and the optimization parameters interactively. Besides, a graphical user interface has been designed to support the user during the development process of the fuzzy controller. Fig. 4 presents the simulation environment of a sample application.

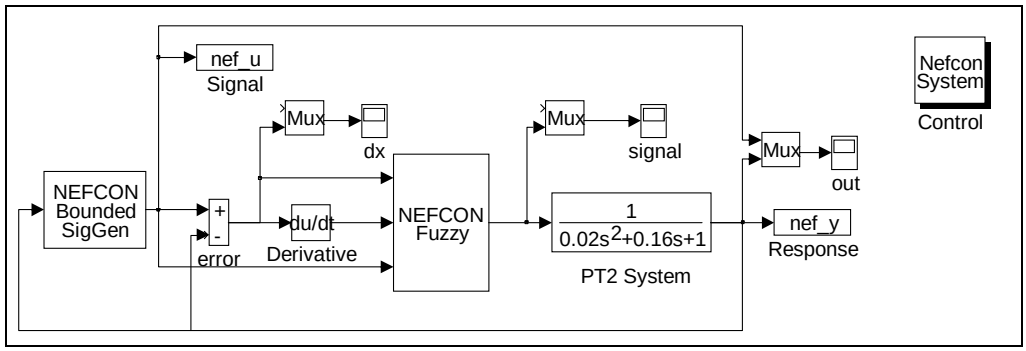

Fig. 4: Sample of a Development Environment under MATLAB/SIMULINK (PT_2-System)

3.1 Example

Since simulation results concerning the inverted pendulum problem are comparable to the results presented in prior publications [NAUCK and KRUSE, 1993; NAUCK et. al., 1995; NAUCK et. al., 1996] we present the simulation results concerning a conventional PT_2-System (see [LEONHARD, 1992; KNAPPE, 1994]). The simulation used the NEFCON I algorithm for rulebase learning and the NEFCON II algorithm for optimization. Three trapezoidal fuzzy sets were used for each input and five for the conclusion. The fuzzy error was described using 'fuzzy intervals' (see Fig. 3). The simulation environment is shown in Fig. 4. The algorithm used a noisy reference signal during rule learning to improve the coverage of the system state space (see cycle 1-3 in Fig. 6). The noise was produced by a signal generator included in the implementation. The system was able to find an appropriate rulebase within 3 rule learning and 3 optimization cycles (with 167 iteration steps each cycle). The rulebase consists of 25 rules.

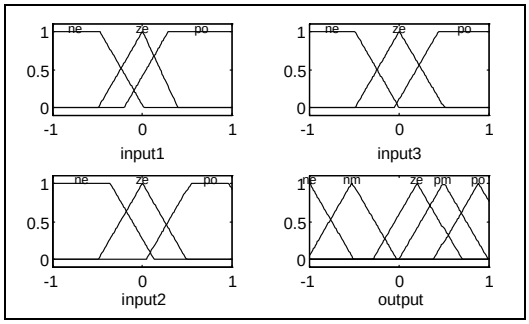

Fig. 5: Fuzzy Sets of Optimized Rulebase

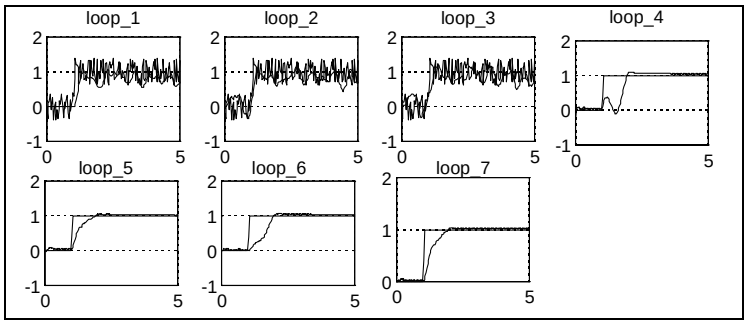

Fig. 6: Simulation Results for a PT_2-System

4 Conclusion

By the implementation of the updated NEFCON model under MATLAB/SIMULINK it is possible to use the model conveniently for the design of fuzzy controllers for different dynamic systems. Changes to system configuration can be carried out in a simple way. The implementation was designed to be used as an interactive development tool.

In case of dynamic systems with less temporal dependence, the rules of the controller will be learned and optimized within a small number of runs. The obtained fuzzy controller will be able to control the dynamic system appropriately. In addition the rulebase is always interpretable. In case of complex systems the quality of the results greatly depends on the definition of the error measure. This is caused by the fact that the NEFCON algorithms use only a simple approach to include the dynamics of the controlled system in the optimization process (see the credit assignment problem [BARTO et. al., 1983]). Some variations of reinforcement strategies [BARTO et. al., 1995; LIN, 1994] have to be analyzed in order to determine if it will be possible to integrate them into the optimization phase of the presented algorithms. It has to be checked whether they improve the quality of the controller without increasing the number of runs for learning significantly.

References

[BARTO et. al., 1983] Barto, A.G., Sutton R. S., Anderson, C. W. (1983): Neuronlike adaptive elements that can solve difficult learning control problems, IEEE Transactions on Systems, Man and Cybernetics, 13:834-846

[BARTO et. al., 1995] Barto, A. G.; Bradtke, S. J.; Singh, S. P.(1995): Learning to act using real-time dynamic programming, Artificial Intelligence, Special Volume: Computational Research on Interaction and Agency, 72(1): 81-138, 1995

[BARTO, 1992] Barto, A.G. (1992): Reinforcement Learning and Adaptive Critic Methods, In [White and Sofge, 1992

[KNAPPE, 1994] Knappe, Heiko (1994): Comparison of Conventional and Fuzzy-Control of Non-Linear Systems, in [KRUSE et. al. 1994]

[KRUSE et. al. 1994] Kruse, Rudolf; Gebhardt, Jörg; Palm, Rainer (Eds.) (1994): Fuzzy Systems in Computer Science, Friedr. Vieweg & Sohn Verlagsgesellschaft mbH, Braunschweig, Wiesbaden

[LEONHARD, 1992] Leonhard, Werner (1992): Einführung in die Regelungstechnik, Friedr. Vieweg & Sohn Verlagsgesellschaft mbH, Braunschweig, Wiesbaden

[LIN, 1994] Lin, C.T. (1994): Neural Fuzzy Control Systems with structure and Parameter Learning, World Scientific Publishing, Singapore

[MAMDANI and ASSILIAN, 1975] Mamdani, E. H.; Assilian S. (1973): An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller, International Journal of Man-Machine Studies, 7:1-13

[NAUCK and KRUSE, 1993] Nauck, Detlef and Kruse, Rudolf (1993): A Fuzzy Neural Network Learning Fuzzy Control Rules and Membership Functions by Fuzzy Error Backpropagation, In Proc. IEEE Int. Conf. on Neural Networks 1993, San Francisco

[NAUCK, 1994] Nauck, Detlef (1994): A Fuzzy Perceptron as a Generic Model for Neuro-Fuzzy Approaches, In Proc. of the 2nd German GI-Workshop Fuzzy-Systeme '94, München

[NAUCK et. al., 1995] Nauck, Detlef; Kruse, Rudolf; Stellmach, Roland (1995): New Learning Algorithms for the Neuro-Fuzzy Environment NEFCON-I, In Proceedings of Neuro-Fuzzy-Systeme '95, 357-364, Darmstadt

[NAUCK et. al., 1996] Nauck, Detlef; Klawonn, Frank; Kruse, Rudolf (1996): Neuronale Netze und Fuzzy-Systeme, 2. Auflage, Friedr. Vieweg & Sohn Verlagsgesellschaft mbH, Braunschweig, Wiesbaden

[NÜRNBERGER, 1996] Nürnberger, Andreas (1996): Entwurf und Implementierung des Neuro-Fuzzy-Modells NEFCON zur Realisierung Neuronaler Fuzzy-Regler unter MATLAB/SIMULINK, Diplomarbeit, Technische Universität Braunschweig, www: http://fuzzy.cs.uni-magdeburg.de/nefcon

[RIEDMILLER and JANUSZ, 1995] Riedmiller, Martin; Janusz, Barbara (1995): Using Neural Reinforcement Controllers in Robotics, In Xian Yao, editor, Proceedings of the 8th. Australian Conference on Artificial Intelligence, Singapore, 1995, World Scientific Publishing, Singapore

[TSCHICHOLD-GÜRMAN, 1995] Tschichold-Gürman, Nadine (1995): RuleNet - A new Knowledge-based Artificial Neural Network Model with Application Examples in Robotics, Dissertational Thesis, ETH Zürich

[TSUKAMOTO, 1979] Tsukamoto, Y. (1979): An Approach to Fuzzy Reasoning Method, In M. Gupta, R. Ragade and R. Yager, Hrsg.: Advances in Fuzzy Set Theory, North-Holland, Amsterdam

[WHITE and SOFGE, 1992] White, D. A., Sofge, D. A., Publ. (1992): Handbook of Intelligent Control. Neural, Fuzzy and Adaptive Approaches, Van Nostrand Reinhold, New York