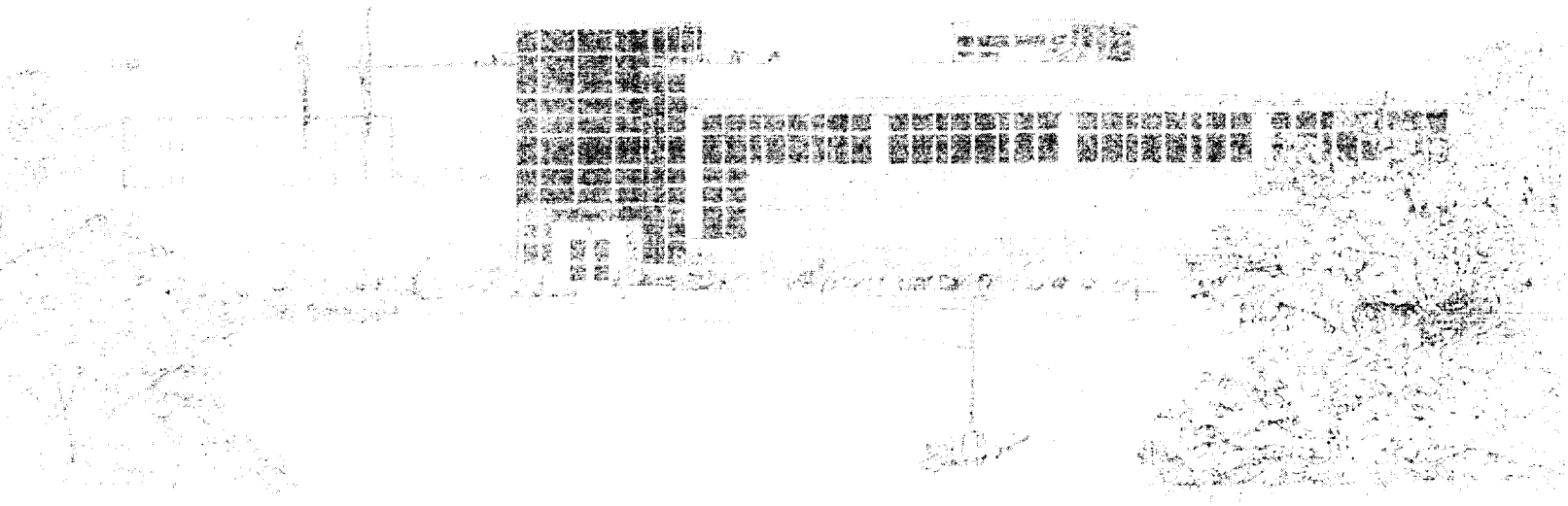
NASA-IVV-97-002
WVU-IVV-97-002
WVU-CS-TR-97-005

NASA/WVU Software IV & V Facility
Software Research Laboratory
Technical Report Series

p 10

Verifying Stability of Dynamic Soft-computing Systems

By Wu Wen, John Callahan and Marcello Napolitano



National Aeronautics and Space Administration

West Virginia University

NASA-IVV-97-002

NASA IV&V Facility, Fairmont, West Virginia

Verifying Stability of Dynamic Soft-computing Systems

Wu Wen, John Callahan, and Marcello Napolitano

June 10, 1997

This technical report is a product of the National Aeronautics and Space Administration (NASA) Software Program, an agency wide program to promote continual improvement of software engineering within NASA. The goals and strategies of this program are documented in the NASA software strategic plan, July 13, 1995.

Additional information is available from the NASA Software IV&V Facility on the World Wide Web site http://www.ivv.nasa.gov/

This research was funded under cooperative Agreement #NCC 2-979 at the NASA/WVU Software Research Laboratory.

To appear, Int. Joint Conf for AI, August 1997, Japan

Verifying Stability of Dynamic Soft-computing Systems

Wu Wen and **John Callahan**
NASA/WVU Software Research Laboratory
Knapp Hall, West Virginia University
Morgantown, WV 26506-6330, USA

Marcello Napolitano
Department of Aerospace Engineering
West Virginia University
Morgantown, WV 26505-6106, USA

Abstract

Soft computing is a general term for algorithms that learn from human knowledge and mimic human skills. Example of such algorithms are fuzzy inference systems and neural networks. Many applications, especially in control engineering, have demonstrated their appropriateness in building intelligent systems that are flexible and robust. Although recent research have shown that certain class of neuro-fuzzy controllers can be proven bounded and stable, they are implementation dependent and difficult to apply to the design and validation process. Many practitioners adopt the trial-and-error approach for system validation or resort to exhaustive testing using prototypes. In this paper, we describe our on-going research towards establishing necessary theoretic foundation as well as building practical tools for the verification and validation of soft-computing systems. A unified model for general neuro-fuzzy system is adopted . Classic non-linear system control theory and recent results of its applications to neuro-fuzzy systems are incorporated and applied to the unified model. It is hoped that general tools can be devloped to help the designer to visualize and manipulate the regions of stability and boundedness, much the same way Bode plots and Root locus plots have helped conventional control design and validation.

1 Introduction

Control systems are intrinsically dynamic and their stability are of primary concern in design. In conventional control system design, the system is carefully modeled. Frequency domain methods and state-space methods are applied to verify the stability of the system for different operation profiles and ranges. Some non-linear systems can be verified through linearization at the point of interest. On the other hand, control systems that employ soft-computing technology such as fuzzy inference system or neural networks lack such established theory as well as tools for comprehensive verification and testing. In fact the prevalent approach in neuro-fuzzy community towards the V & V of such system is summarized by Prof. Mamdini in his 1993 paper[Mamdini, 1993]:

> Stability is still an important issue but a different way has to be found to study it. In the final analysis all one may be able to do is to build prototypes for the purpose of approval certification. This is a well tried and tested approach used in industry and there is no reason why it may not suffice with control system as well.

There are reasons to believe that the above approach is too conservative and furthermore it may not be feasible in certain situations.

Recent results[Levin and Narendra, 1996; Nordgren and Meckl, 1993; Vidyasagar, 1993; Tanaka, 1995; 1996; Fang and Kincaid, 1996; Wang, 1993] have shown that, for certain class of neuro-fuzzy control systems, it is possible to ascertain the stability and bounds of the system with careful choice of parameters or through certain measurements of weights. These new results are mostly based on classic non-linear system theory of Liapounov stability and asymptotic stability and are well founded.

For large complex systems with embedded soft-computing components, it is not clear whether only testing the prototype of the soft-computing components would be sufficient. Finally, any tests conducted on the prototype short of an exhaustive testing may not be enough for safety critical systems such as those used in aerospace industry. An exhaustive testing of a continuous dynamic system may not be practical or even possible. Some of the problems of verification of soft-computing systems have been discussed in [Wen and Callahan, 1996b; 1996a; Wen *et al.*, 1996].

Our approach is based on the methodology of treating neuro-fuzzy control system as any general non-linear

system. A fuzzy-neuro model called ANFIS(Adaptive Network-based Fuzzy Inference System)[Jang, 1996]-is used to model a given neuro-fuzzy system. The control surface is divided into regions where a predominant set of linear rules appliy at the core of the region. Boundary conditions are carefully modeled. Stability analysis algorithms developed for general non-linear system and some class of neuro-fuzzy systems are then used to verify the stability of the soft-computing system in each region and the joining boarders. An example of how this can be achieved is illustrated using a neuro-fuzzy cart-pole controller.

The ultimate goal of our work is to develop practical tools for analyzing and verification of large, distributed control systems that employ soft-computing components. Currently the stability verification tool is applied to project AIRNET[Napolitano and Kincheloe, 1995], which aims at the designing verifiable neural network based auto-pilot for a model Boeing 747 airplane. We hope to improve the tool through application in this reasonably complex real-world situation.

The paper is organized as follows: Section 2 describes elements of classic non-linear system dynamics and stability theory that are relevant to general neuro-fuzzy systems. Section 3 describes a general model for an arbitrary neuro-fuzzy system. Section 4 describes how methods described in Section 2 can be applied to the generalized neuro-fuzzy model. Section 5 shows an example of applying these methods to an simple cart-pole neuro-fuzzy controller. In Section 6 we conclude and present future works needed.

2 Dynamic non-linear system and stability

Neuro-fuzzy systems aim at mimicking human expertise and adaptiveness to achieve difficult control tasks. The building blocks of neuro-fuzzy systems are non-linear functions such as the logistic function for many neuro-networks and generalized bell function for fuzzy controllers. These non-linear functions are applied to or are combined by linear weighting mechanisms to achieve the required complex functional mapping which describes the control task at hand. In this section, we introduce the essentials of non-linear dynamic systems and stability criteria that are relevant to the general neuro-fuzzy systems.

2.1 Non-linear system dynamics

The general form of non-linear system dynamics can be described by the following differential equation:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, t) \qquad (1)$$

where $\mathbf{x}$ is the state vector, $\mathbf{u} = g(\mathbf{x})$ is the control vector. Notice that higher order differentiable terms are expanded into vectors of first order differentiables. The aim of control system design is to find appropriate $\mathbf{u}$ so that $\mathbf{x}$ follows certain pre-defined trajectory.

When the time variable t doesn't appear explicitly in the right hand side of equation (1), the system is called an autonomous system. Neuro-fuzzy based control systems are examples of autonomous systems since the control actions $\mathbf{u}$ is only a function of the state variable $\mathbf{x}$.

2.2 Stability criteria

There is a large amount of theoretic work in the area of ascertaining stability and instability of dynamic non-linear system. A good introduction into these theories can be found in standard textbooks[Jordan and Smith, 1977; Glendinning, 1994]. It's beyond the scope of this paper to introduce any of these theories. However, we will summarize some practical methods as applications of those theories.

- Phase diagram method: plot the trajectory of $\mathbf{x}$ for some given initial starting points. For differential equations with analytic forms, regions of stable and unstable equilibrium can be quickly identified. Difficult for high dimension problems.
- Liapounov function method: construct a Liapounov function around the point of interest based on the differential equations. There is no guarantee that this Liapounov function can be found and the failure to find one can not be used as the evidence for instability.
- Perturbation methods: introduce a perturbation into a well-known system to obtain results for the class of problems described by the perturbed system.
- Linearization methods: linearize the non-linear system at the point of interest and apply linear system theory to non-linear system locally.

More recently there are some interesting results in this field that are directed towards neuro-fuzzy systems:

- Linear differential inclusion[Tanaka, 1995; 1996]: neuro-fuzzy systems are represented as linear combination of some class of special nonlinear functions. Based on the properties of the class of non-linear functions and the coefficiency of linear summation, stability of these type of neuro-fuzzy systems can be verified.
- Matrix measurement method[Fang and Kincaid, 1996]: a matrix measurement is introduced from the (matrix) differential equations. This measurement can be used to ascertain the stability of the underlying system. Certain neuro-fuzzy implementation can be shown to have a matrix measurement that could guarantee stability.

Some of the methods were developed to ascertain stability of neural networks during the learning process. These methods are concerned with the convergence and stability property of neural networks in the learning process. However, the results applies to the stability of any dynamics system that can be described by differential equations. The overall system stability can be ascertained if we can obtain similar measurements for the system dynamics differential equations.

The above methods provide us with the building blocks for developing a tool to help visualizing and manipulate the stability and boundaries of arbitrary neuro-fuzzy systems.

3 Neuro-fuzzy control system

Neural networks and fuzzy logic have been studied for decades, mostly in separation. Neural networks were mainly used to learn complex mapping between known input-output pairs. It requires a "teacher" to provide data for the "learning". It mimics human or other "teacher" by repeating exactly what the "teacher" did in exact the same situation.

On the other hand, fuzzy logic emphasizes on rules that map situations to actions. It does not try to mimic exactly what the "teacher" does but aim at extracting the essence of decision making process of the "teacher".

Recently, researchers have realized that by combining the rule-extracting and adaptive learning, more powerful systems can be built that incorporates human knowledge and skill by learning from what humans think and what humans do at the same time. Moreover, the layered propagation structure of the neural networks and fuzzy rule firing structure are very similar. They can be combined naturally to form what is called the neuro-fuzzy system.

3.1 ANFIS model for general neuro-fuzzy system

Various neuro-fuzzy systems have been proposed[Lin and Lee, 1991; Takagi and Sugeno, 1991] in recent years. The basic idea is to construct a fuzzy logic system that can adapt its membership function or rules based on back-propagation or other optimization methods. The ANFIS(Adaptive Network-based Fuzzy Inference System) tool proposed by Jang[Jang and Sun, 1995] is probably the simplest one with an implementation on a popular platform, MATLAB[Jang, 1996]. It is used in our work as the unified model for arbitrary neuro-fuzzy system. To justify this property, we will introduce briefly the relevant elements of the ANFIS system.

ANFIS is the network implementation of the Sugeno fuzzy inference system[Takagi and Sugeno, 1985]. The network topology and inference rules are shown in Figure 1. x, y are inputs to the controller and g is the controller output. A_i and B_i are square nodes that represent the membership functions which have three adaptive parameters: the center, the width and steepness. The first layer of circle nodes(indicated by a $*$) are fixed nodes which perform the fuzzy MIN operation. The next layer of nodes are also fixed nodes(indicated by circles with /+) which perform fuzzy MAX operation. The next layer of square nodes(indicated by f_1 and f_2) are the linear rules to be fired. They each has three or more adaptive parameters which uniquely determine a straight line or plane(the linear rule).

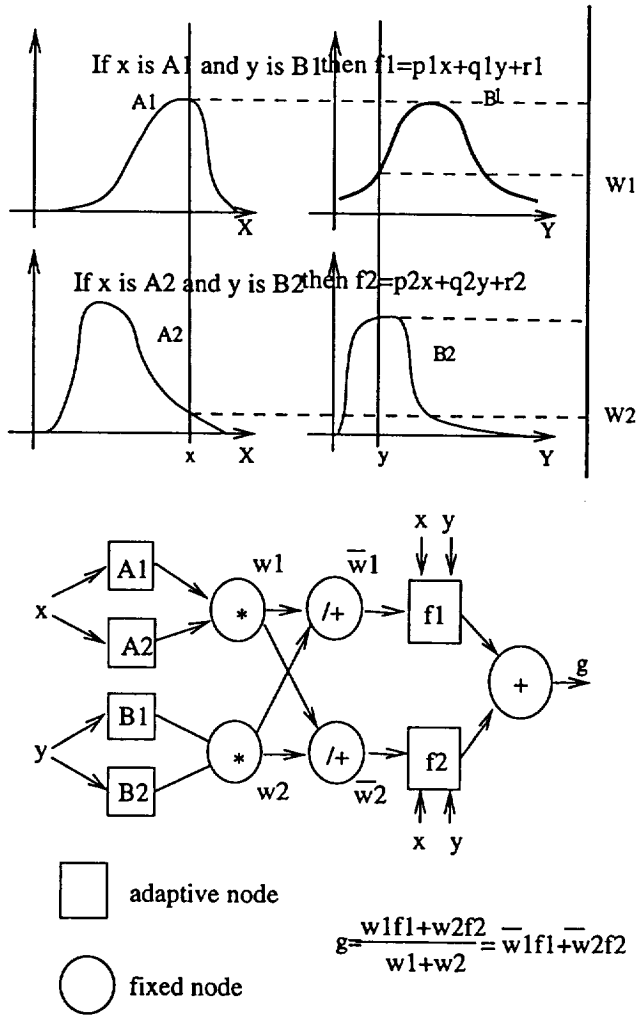


Figure 1: The Adaptive Network-based Fuzzy Inference System, courtesy to R. Jang

ANFIS can be used as a model for an arbitrary neural network or fuzzy controller. The modeling of a fuzzy inference system is straight forward by substitution of membership functions and fuzzy rules. In the case of a neural network controller, once it has been trained, the trained neural networks can be used to generate train-

ing data for ANFIS. Since ANFIS is a universal function approximator[Jang and Sun, 1995], in theory, it is possible to approximate any trained neural networks to any degree of closeness. This observation forms the basis of our justification to use ANFIS as the model for arbitrary neuro-fuzzy system. It must be noted that although ANFIS will approximate the trained neural networks of interest to any degree of closeness, it may not approximate the original training dataset well due to factors such as incomplete dataset etc. ANFIS is used here as a model of the trained neural networks, not the original systems.

3.2 Divide and conquer with ANFIS

In order to visualize and manipulate the neuro-fuzzy controller with respect to its state space, the input space must be divided into regions that share similar properties with respect to stability etc. Once ANFIS is trained using the underlying neuro-fuzzy system, rules are refined and extracted. Fuzzy membership functions are also obtained which partitions the input space into regions where one dominant rule applies.

The following example in Figure 2 shows how ANFIS can divide and conquer the input space into meaningful regions in which a dominant linear rule applies. The training data is generated from the following equation:

$$f(u) = 0.6\sin(\pi u) + 0.3\sin(\pi u) + 0.1\sin(5\pi u).$$

The final MFs shows the partitions in which a dominant linear rule applies. This ability to divide and conquer is essential to our approach of developing tools to visualize and manipulate the neuro-fuzzy system to ascertain stability and other properties. It does so by allowing us to gain some understanding of what the underlying neuro-fuzzy system actually looks like. In other word, ANFIS provides us with hints on what the neural networks have learned from training data.

It must be pointed out that this divide and conquer process is not completely automatic and requires careful human intervention. The number of partitions must be pre-determined and improper choices can lead to erroneous results. Figure 3 shows the results with four partitions. It is clear that the extracted linear rules are hardly representative of its partition. Care must be taken in selecting the number of partitions. The only way to assure correct choice is through trial-and-error.

4 Verification of dynamic neuro-fuzzy system

In Section 2.2 we have introduced a few methods for ascertaining stability of some special class of neuro-fuzzy control systems. To apply them to the ANFIS model, some modifications must be made. In this section we will describe our strategy with intention of applying it in an embedded neural network controller.

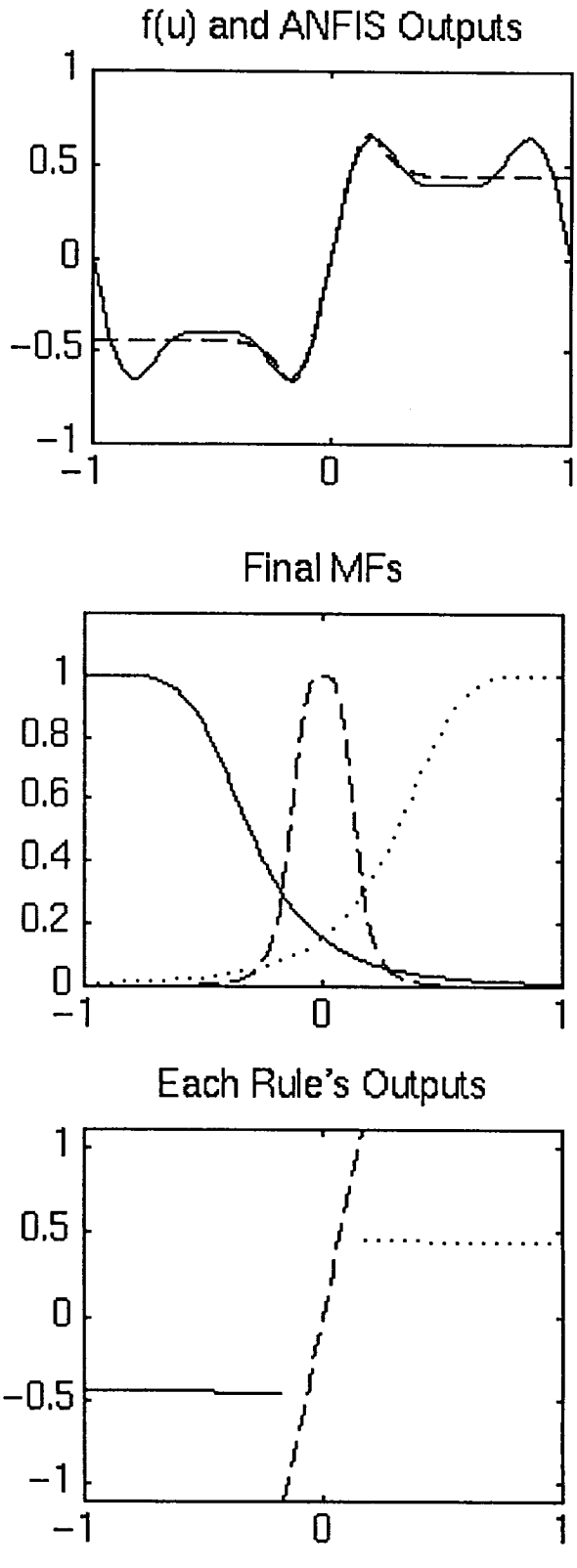


Figure 2: Input space are divided into three regions in which a dominant linear rule applies

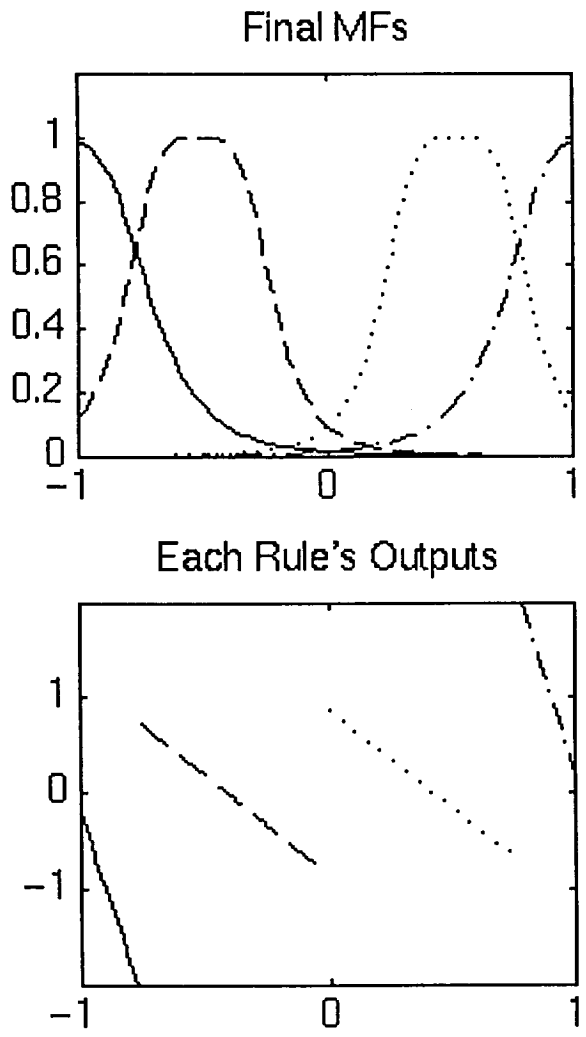


Figure 3: Choice of number of partitions are crucial to divide and conquer process

4.1 A general framework

The stability of the learning algorithm of neural networks and that of the system being controlled by the neural networks are quite different matters. However, they can both be represented by a set of general differential equations. A general dynamic system under control can be describe as follows: plant dynamics

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}), \tag{2}$$

and controller

$$\mathbf{u} = g(\mathbf{x}). \tag{3}$$

This is shown schematically in Figure 4.

The purpose of control is to find appropriate control actions $\mathbf{u} = g(\mathbf{x})$ so that $\mathbf{x}$ follows certain pre-defined trajectory. Once $\mathbf{u}$ is determined, the differential equation becomes

$$\dot{\mathbf{x}} = f(\mathbf{x}, g(\mathbf{x})).$$

In general this can be considered as an autonomous system and a number of methods mentioned in Section 2.2 can then be used to ascertain the overall system stability. However, before they can be applied directly to the ANFIS model, some modification must be made.

Usually ANFIS is used to model the control function $\mathbf{u} = g(\mathbf{x})$ only. In order to ascertain the overall system stability, it is necessary to generate appropriate representation for the overall system dynamics differential equations.

In most situations, the plant dynamics are obtained from physical properties of the system and have analytical form. Once we obtained the control laws it is possible to generate an ANFIS representation of the overall system dynamics. Given a state $\mathbf{x}$, control action $\mathbf{u}$ is determined by the neuro-fuzzy controller ouputs. Moreover, $\dot{\mathbf{x}}$ can be determined from euqation (2) by substitution of $\mathbf{u}$ with $g(\mathbf{x})$. The pair $\mathbf{x}$ and $\dot{\mathbf{x}}$ can then be used to train a new ANFIS model to represent the overall system dynamics. Divide and conquer can be used and the state space can be partitioned into regions in which dominant rule applies.

In the case of ANFIS, these rules will be linear. LDI method, matrix measure method or linear perturbation method can be used in each region to ascertain the stability of the overall system.

4.2 The AIRNET testbed

Our long term objective is to develop tools that can help to visualize and manipulate properties such as boundedness and stability of arbitrary neuro-fuzzy systems. To this end, we are in the process of developing this tool for its use within the AIRNET project[Wen *et al.*, 1996].

The AIRNET project consists of a 1:10 scale model of a Boeing 747 fitted with a neural network based sensing and estimation subsystem and a neural network based auto-pilot subsystem. These neural network subsystems

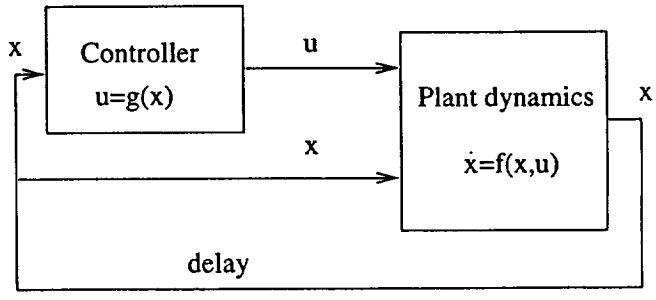


Figure 4: A general dynamic system

are distributed over seven micro-processors located on various part of the airplane body.

Currently the model airplane is controlled by remote radio controls and test flies are conducted to gather enough data to train the neural networks. Once trained, the sensor neural network subsystem will be responsible for generating robust estimation of airplane aerodynamic states. The auto-pilot neural network subsystem will be responsible to maintain certain maneuvers such as altitude hold, speed hold and climbing etc.

For systems as complicated as this, simple stable or unstable criteria is not sufficient. We must provide tools that help to visualize and manipulate these properties in the entire state space. Furthermore, these neural network subsystems are embedded in the overall system, it is essential that they can be manipulated conveniently to achieve better performance for the overall system.

4.3 Validation strategy

The sensing subsystem and the control subsystem must be validated separately. The sensing subsystem is implemented as follows: first a mathematical model of the airplane is built and neural networks is trained using data generated by the math model. Then test flights are conducted to gather real aerodynamic data for the model airplane. The real data is also used to train the neural networks. ANFIS is then applied to both neural networks to extract the rules and partitions. The results for both neural networks are then compared. The math model is modified to maintain consistency with the real data.

The next step is to train the neural network auto-pilot. This will be achieved through many cycles of coaching using remote radio control. Once the weight are fixed, we need to validate the neural network auto-pilot for stability. This again will be achieved through application of divide and conquer using ANFIS, followed by stability verification methods described in Section 2.2.

The complexity of the auto-pilot system poses great challenge to our approach. One of the major problem we are facing now is that ANFIS only allows one output. Currently we have to use one ANFIS network for each output. We are working to extend ANFIS to multiple outputs.

5 Experimentation

To illustrate the applicability of the stability verification tool, we present an simple example of neuro-fuzzy controller. The original controller is implemented in standard BP neural networks.

5.1 A cart-pole controller

Figure 5 shows a cart-pole system which has been used extensively to demonstrate various kinds of control algorithms. The objective of the controller is to maintain a balanced position for the pole through exerting a horizontal force on the cart. The plant model is as follows:

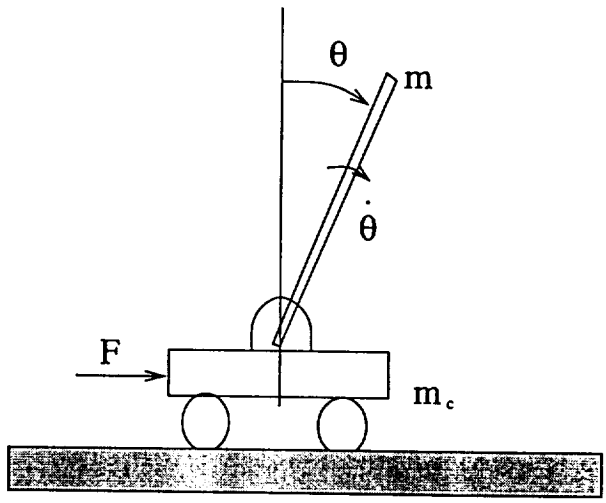
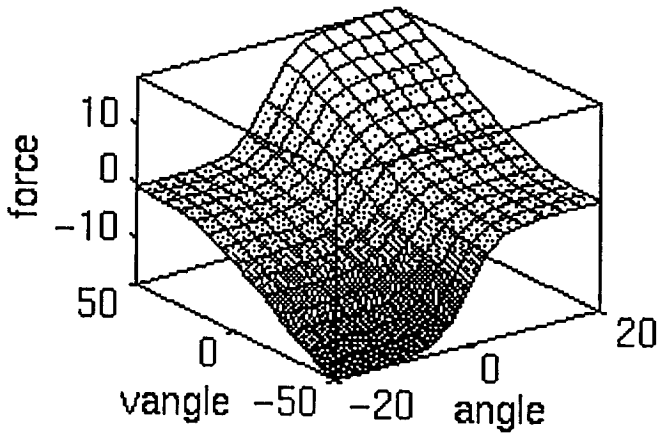


Figure 5: A cart-pole balancing problem

Figure 6: Control surface of the neural network cart-pole controller

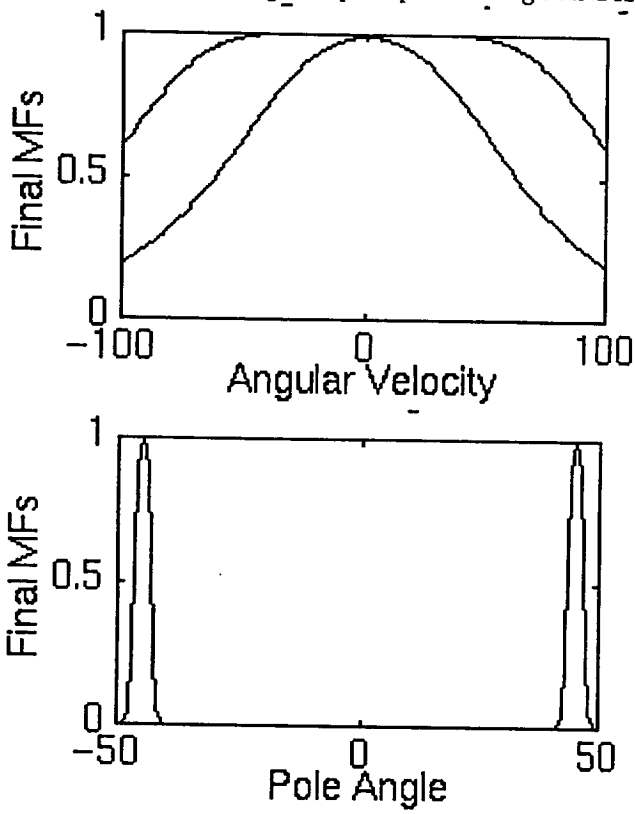


$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = f(\mathbf{x}, u) = \begin{bmatrix} x_2 \\ \dfrac{g \sin x_1 + \cos x_1 \left(\dfrac{-u - ml x_2^2 \sin x_1}{m_c + m} \right)}{l \left(\frac{4}{3} - \dfrac{m \cos^2 x_1}{m_c + m} \right)} \\ x_4 \\ \dfrac{u + ml(x_2^2 \sin x_1 - \dot{x}_2 \cos x_1)}{m_c + m} \end{bmatrix}$$

.

Figure 6 shows the control surface of the original neural network controller. First the original neural network controller is used to generate training data for ANFIS. Next ANFIS is applied to partition the input space and extract dominant linear rules in these partitions. The following is the extracted rules:

Figure 7: Partitioning of input space through ANFIS



if θ is A_1 and $\dot{\theta}$ is B_1, then

$$force = 0.05\theta + 0.165\dot{\theta} - 10.1$$

if θ is A_1 and $\dot{\theta}$ is B_2, then

$$force = 0.008\theta + 0.012\dot{\theta} - 1.1$$

if θ is A_2 and $\dot{\theta}$ is B_1, then

$$force = 0.008\theta + 0.012\dot{\theta} + 1.1$$

if θ is A_2 and $\dot{\theta}$ is B_2, then

$$force = 0.05\theta + 0.165\dot{\theta} + 10.1$$

The resulting membership function is plotted in Figure 7.

From the dynamic system differential equation and the ANFIS controller we can obtain an ANFIS representation of the system dynamic model. Figure 8 shows the phase diagram of the overall system model generated by ANFIS. The horizontal axis is the angular displacement x_1 and the vertical axis is the angular velocity x_2. It can be clearly seen that the origin is a stable equilibrium point.

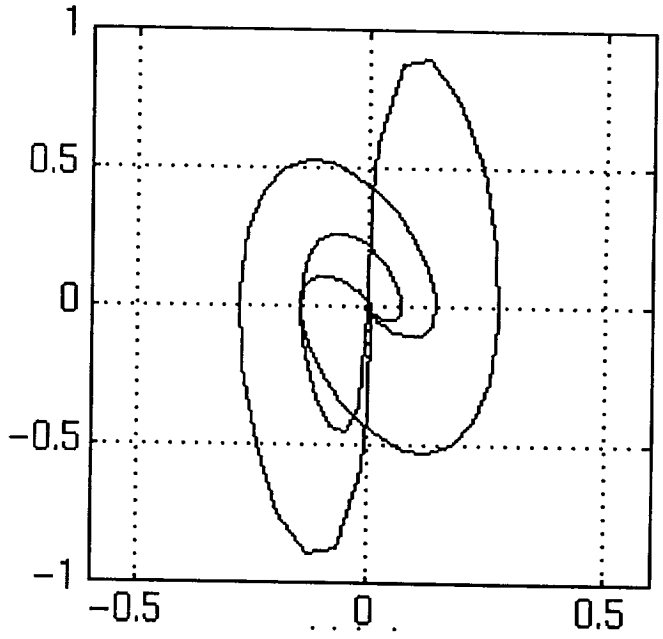


Figure 8: Phase diagram of the neural-fuzzy cart-pole controller

6 Conclusions and future work

This paper described our on-going work in developing a practical tool for help visualizing and manipulating neuro-fuzzy controllers to ascertain their stability in regions of interest. The essence of this approach is to use ANFIS as a unified model for arbitrary neuro-fuzzy system. Moreover, an ANFIS model is generated to represent the overall system dynamics. This ANFIS model thus provide us with an effective partition of the state space and linear rules that are dominant in those partitions. A number of stability methods can be applied in each of these regions and verification of the overall stability of the system can be achieved.

So far our only implementation is on a simple cart-pole problem. The effectiveness of this approach will be fully tested when we apply it to more complicated situations. Our long term objective is to develop a tool that allow us to deal with distributed neuro-fuzzy systems. Future work will include

- extend ANFIS to allow multiple outputs;
- build toolkits that use different methods such as the LDI methods, matrix measure methods and phase diagram methods to deal with different subsystems;
- build a set of interfaces to allow simulation of embedded neuro-fuzzy systems.

7 Acknowledgment

Our thanks are due to Prof. Mizoguchi and Prof. Ohwada at Tokyo Science University for many inter-

esting discussions with the first author. This work is supported by NASA through cooperative research agreement #NCC 2-979.

References

[Fang and Kincaid, 1996] Y. Fang and T. G. Kincaid. Stability analysis of dynamical neural networks. *IEEE Trans. on Fuzzy Systems*, 7(4):996–1006, 1996.

[Glendinning, 1994] P. Glendinning. *Stability, instability and chaos: an introduction to the theory of nonlinear differential equations.* Cambridge University Press, 1994.

[Jang and Sun, 1995] J. S. R. Jang and C. T. Sun. Neuro-fuzzy modeling and control. *Proceedings of IEEE*, 83(3):378–406, 1995.

[Jang, 1996] J. R. Jang. *Fuzzy logic toolbox with Matlab.* The Matworks Inc., 1996.

[Jordan and Smith, 1977] D. W. Jordan and P. Smith. *Nonlinear ordinary differential equations.* Clarendon Press, 1977.

[Levin and Narendra, 1996] A. U. Levin and K. S. Narendra. Control of nonlinear dynamical systems using neural networks—part ii: Observability, identification, and control. *IEEE Trans. on Neural Networks*, 7(1):30–42, 1996.

[Lin and Lee, 1991] C. T. Lin and C. S. G. Lee. Neural-network-based fuzzy logic control and decision system. *IEEE Trans on Computers*, 40(12):1320–1336, 1991.

[Mamdini, 1993] E. H. Mamdini. Twenty years of fuzzy control: experiences gained and lessons learned. *IEEE Magzine*, pages 339–344, 1993.

[Napolitano and Kincheloe, 1995] M. R. Napolitano and M. Kincheloe. On-line learning nueral network controllers for autopilot systems. In *95' AIAA Guidance Navigation and Control Conference*, Baltimore, Md, August 1995.

[Nordgren and Meckl, 1993] R. E. Nordgren and P. H. Meckl. An analytical comparison of a neural network and a model-based adaptive controller. *IEEE Trans. on Neural Networks*, 4(4):685–694, 1993.

[Takagi and Sugeno, 1985] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans. on SMC*, 15:116–132, 1985.

[Takagi and Sugeno, 1991] T. Takagi and M Sugeno. Nn-driven fuzzy reasoning. *Int. Journal of Approximate Reasoning*, 5(3):191–212, 1991.

[Tanaka, 1995] K. Tanaka. Stability and stabilizability of fuzzy-neural-linear control systems. *IEEE Trans. on Fuzzy Systems*, 3(4):438–447, 1995.

[Tanaka, 1996] K. Tanaka. An approach to stability criteria of neural-network control systems. *IEEE Trans. on Neural Networks*, 7(3):629–642, 1996.

[Vidyasagar, 1993] M. Vidyasagar. Location and stability of the high-gain equilibria of nonlinear neural networks. *IEEE Trans. on Neural Networks*, 4(4):660–672, 1993.

[Wang, 1993] L.-X. Wang. Stable adaptive fuzzy control of nonlinear systems. *IEEE Trans. on Fuzzy Systems*, 1(2):146–155, 1993.

[Wen and Callahan, 1996a] W. Wen and J. Callahan. Neuralware engineering: develop verifiable ann systems. In *Symposium on Intelligent Systems and Robotics*, Washington D. C., November 1996.

[Wen and Callahan, 1996b] W. Wen and J. Callahan. Verification and validation of knowledge-based systems with ann components. In *Workshop on verification and validation of KBS*, Portland, Oregon, August 1996.

[Wen *et al.*, 1996] W. Wen, J. Callahan, and M. Napolitano. Develope verifiable neural network controller. In *ICTAI Workshop on AI applications to Aviation*, Toulouse, France, November 1996.