Toward Verification and Validation of Adaptive Aircraft Controllers

Johann Schumann† Pramod Gupta‡ Stephen Jacklin⋆
†RIACS / NASA Ames, Moffett Field, CA, schumann@email.arc.nasa.gov
‡QSS Inc. NASA Ames, Moffett Field, CA, pgupta@email.arc.nasa.gov
⋆NASA Ames, Moffett Field, CA, sjacklin@mail.arc.nasa.gov

Abstract—Traditional fixed-gain control has proven to be unsuccessful to deal with complex, strongly nonlinear, uncertain, and changing systems such as a damaged aircraft. Control systems with components that can *adapt* toward changes in the plant, e.g., using a neural network, have been actively investigated as they offer many advantages (e.g., better performance, controllability of aircraft despite of a damaged wing). However, neuro-adaptive controllers have not been used in safety-critical applications, because performance and safety guarantees cannot be provided at development time—a major prerequisite for safety certification (e.g., by the FAA or NASA). In this paper, we will describe our approach toward V&V of neuro-adaptive controllers. We have developed tools which dynamically estimate the neural network performance and safety envelope, using a Bayesian approach. We will discuss our V&V approach, the tool architecture and simulation experiments within NASA's IFCS (Intelligent Flight Control System) project.

Table of Contents

1 Introduction
2 Background: Neuroadaptive Control
3 V&V of Adaptive Control
4 V&V Tools
5 Conclusions
6 Acknowledgments

1. Introduction

Emerging aerospace system operational goals, such as autonomy, will require advanced safety-critical control systems consisting of unconventional requirements, systems architectures, software algorithms, and hardware implementations. These control systems will make use of sophisticated models and controllers (adaptive), the behavior of which cannot always be simply characterized. Due to the operational uncertainties and communication delays, on board systems need to be dynamic, providing for rapid reconfiguration and modification in order to respond to unanticipated mission problems. The control system may need to make decisions in an unstructured environment, the functions being performed in the past by human operators of mundane tasks. To protect the aircraft from hazards, a high degree of autonomy is desired. Truly autonomous operations will require air and space vehicle safety-critical control system enhancements to achieve required safety levels without reliance on human intervention. Flight critical system requirements assert that the occurrence of any failure condition that would prevent the continued safe flight and landing of the airplane shall be extremely improbable. Applicability of advanced and adaptive control systems strongly depends on affordable V&V strategies that reduce costs and compress schedules for flight certifications.

Despite the scrutiny that adaptive control has received in a theoretical context, complex controllers that incorporate adaptation and intelligent decision making have not yet been used in mission or safety-critical applications, simply because the "right" tools for verification and validation have not been applied to the adaptive controller, frequently leading to incorrect judgments of the control performance. While theory and concepts of adaptive systems and intelligent control have been studied in depth over the past decade or so, very little attention has been paid to the issue of validating the correctness and safety of their operation. In particular, controllers for safety and mission-critical applications must satisfy a stringent verification and validation program which assigns quantitative bounds to their output error under all operating conditions, and guarantee that no combination of inputs will result in an undesirable output.

Hence, verification and validation technologies and tools are needed that can assure the reliable and safe operation during all operating conditions. In this paper, we will describe our approach to verification and validation of adaptive controllers, which are based upon neural networks (NN). We have developed tools which support the automation of verification, validation, and dynamic monitoring of system performance for reliable and safe operation. The time and cost for certifying new safety-critical systems could potentially be radically reduced through the application of new software technologies. Our primary goal is to enable affordable development of future safety-critical systems with prescribed levels of safety and reliability. The primary benefit is enabling cost-effective, rapid development of safe and reliable autonomous safety-critical systems.

In Section 2, we will give a brief description of the IFCS (Intelligent Flight Control System) adaptive controller, which is

0-7803-7231-X/01/$10.00/©2002 IEEE
IEEEAC paper # XXX

used as our example throughout the paper. Then, we discuss our layered approach to V&V, before we describe our tools, the Confidence Tool and the Envelope Tool, and present experimental results.

2. BACKGROUND: NEUROADAPTIVE CONTROL

We will illustrate our approach with the adaptive flight control system (FCS) which has been developed within the IFCS project at NASA and give a brief description of its adaptive control architecture [11]. The target aircraft for this controller is a specifically configured F15 jet aircraft. It is equipped with additional actuator surfaces (canards) that are located in front of the wings. By moving them, the airflow over the wing can be modified in a wide range. Thus, this aircraft can be used to simulate failures like wing-damage during test flights. The FCS (Fig. 1) is a straight-forward dynamic inverse controller: the pilot steering commands are mixed with the current sensor readings (airspeed, angle of attack, altitude) to form the desired behavior of the aircraft (measured as roll-rate, pitch-rate, and yaw-rate). The dynamic inverse model then calculates the required actuator movements (e.g., of aileron or rudder) to bring the aircraft into the desired state.

If the aerodynamics of the aircraft changes (e.g., due to a broken surface), there is a deviation between desired and actual state. The neural network is trained during operation to produce a correction signal U_{AD} to minimize this deviation. The inputs of the neural network are typically the current state of the aircraft (i.e., the sensor signals), the commanded input, and the correction signal of the previous time frame.

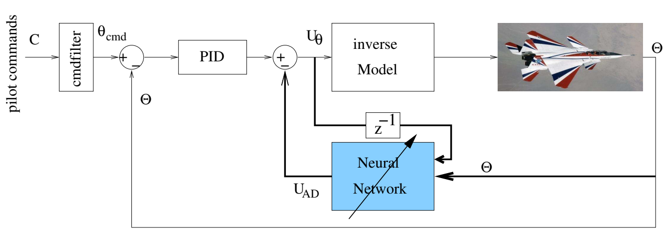


Figure 1. IFCS Adaptive Control Architecture

The controller (IFCS Gen-II, [8], [2]) uses a Sigma-Pi neural network [7]. In this network (Figure 2), the inputs are subjected to basis functions (e.g., square, scaling, logistic function). Then products (Π) of these function values are calculated. The final output of the network is a weighted sum (Σ) of these products—hence the name of this architecture.

The network is trained according to a given update rule [8]. The weight update rule is derived from a Lyapunov stability analysis of the entire controller. In this paper, we will not discuss this weight update rule (see [8] for details), because our approach is independent of the learning rule. In fact, our algorithm can easily be adapted for other network types.

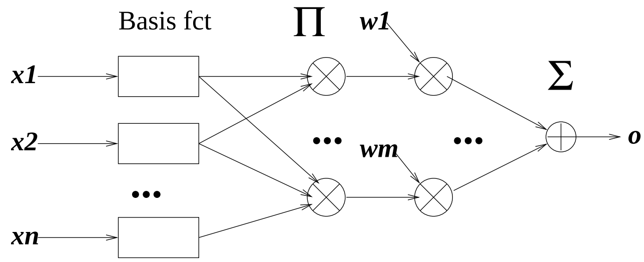


Figure 2. Architecture of $\Sigma\Pi$ network

3. V&V OF ADAPTIVE CONTROL

All safety-critical software applications require careful verification and validation of the software components, ranging from extended testing to full-fledged certification procedures (e.g., DO178-B). Many incidents (e.g., Ariane 5, Mars Polar Lander, Mars Climate Orbiter) illustrate the importance of this issue.

A number of prototypical/experimental application of neural networks in safety-critical areas have demonstrated superior behavior and practical usefulness. Unless, however, methods and techniques have been developed which are capable of assuring the correctness and performance of a neural-network based system, NN applicability in safety-critical areas is substantially limited. The adaptive nature of neural networks requires a significantly different approach to V&V than used for traditional software. Although traditional V&V and certification practices have historically produced sufficiently safe and reliable aircraft control systems, they will not be cost effective for next-generation autonomous control systems due to inherent size and complexity increases from added functionality. Current V&V methods, which rely heavily on testing, make up a large fraction of development costs of a system, yet they do not strictly guarantee performance. Dynamic adaptation of parameters, iterative numerical algorithms, and complex control architectures renders traditional approaches to V&V impracticable.

Currently, there is no established way to verify and validate adaptive and intelligent flight-critical control software leading to certification. In order for any flight critical software to be certifiable by the Federal Aviation Authority (FAA), it must be developed according to a detailed and well-documented software development process, which is extremely time-consuming and costly. The specific requirements for adaptive systems, however, are not easily discernible. Moreover, as emerging safety-critical systems become more complex, system certification costs will increase exponentially due to projected increase in required testing resources, such as hardware in loop (HIL) testing labor.

To address the problem, we are developing and maturing a multi-layered approach to V&V techniques and methods for intelligent adaptive controllers (Fig. 3) [10], [11]. The core layer contains rigorous, mathematically sound results concerning robustness, stability, or Vapnik-Chervonenkis-dimension arguments to reason about the potential of learnability. These methods, however, only can provide relatively weak results, often in form of asymptotic guarantees, which are not sufficient to guarantee the required safety and reliability properties for safety-critical applications.

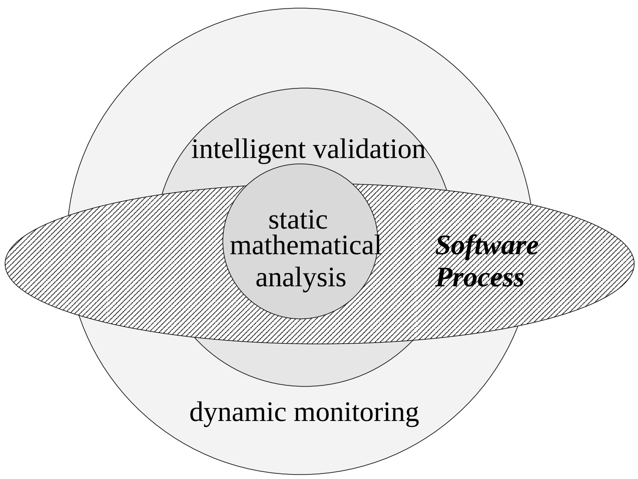


Figure 3. Layers of ANN V&V methods

Intelligent testing and error checking on the second layer provides additional performance and safety guarantees. Testing of an adaptive controller, however, is far from trivial, because testing needs to take into account the system's history (usually resulting in a prohibitively large number of test cases) and the current state of the adaptive component (nominal, in adaptation, adapted). V&V tools will be designed in such a way that they are fine-tuned to meet the demands and idiosyncrasies of adaptive or autonomous systems.

For truly adaptive systems, however, we still do not have *a-priori* guarantee for performance. Here, the third layer comes into play: methods in this layer will dynamically monitor the system and its behavior. Such a monitor will return dynamic information on how the system currently behaves and if the current state of the system is recoverable at all.

4. V&V TOOLS

In the following, we will describe two tools, which have been developed by the authors to specifically address the V&V issues discussed above. The tools are based on a Bayesian approach and produce a statistical sound estimation of the network's current performance. These tools can be used during the entire software life cycle, during design time to assess and fine-tune the network architecture, during validation tests, and to dynamically monitor the network's behavior after deployment.

Confidence Tool

The knowledge of the current quality of the model is important in order to obtain a probability under which a feedback controller exceeds it robustness limits. Any feedback controller can always handle small deviations between the model and the actual plant dynamics. However, this robustness is strictly limited by the design of the controller, so large deviations between model and plant or a very noisy model can cause severe problems. Our *Confidence Tool* (CT) [4] produces a quality measure of the neural network output, which directly can be used to assess the model quality[1]. In order to define our performance measure, we calculate the probability density function $p(o|\mathbf{x}, D)$ of the network output o when it is subject to inputs $\mathbf{x}$ and the network has been trained with training data D. Assuming a Gaussian (Normal) distribution, we use its characteristic parameter, the standard deviation σ^2 as our performance measure. A small σ^2 results in a narrow bell-shaped curve, meaning that, with a high probability, the actual value is close to the returned value. This indicates a good performance of the network. A large σ^2 corresponds to a shallow and wide curve. Here, a large deviation is probable, indicating poor performance.

Our confidence tool uses an algorithm, following the derivation in [1]. The CT has been implemented in Matlab (for a Simulink environment) and in C. The CT tool is currently implemented on the flight computer on a NASA F-15 aircraft and is scheduled for manned test-flights in the near future.

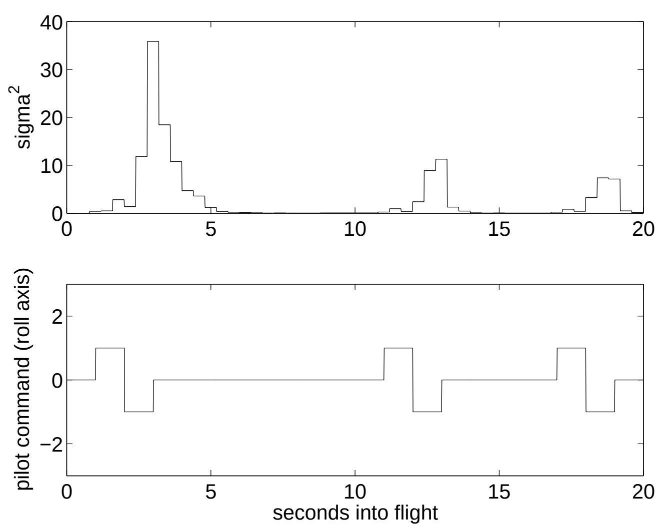


Figure 4. Confidence value σ^2 over time *(top)* and pilot commands for roll axis (*bottom*). A failure has occurred at $T = 1.5s$.

Figure 4 shows the results of a (Simulink) simulation experiment carried out with the on-line adaptive controller of Fig-

[1] In the Gen-II architecture, the network produces a control augmentation signal which is directly fed into the inverse model. In adaptive control architectures where the NN directly encodes the model (e.g., [6]), the CT directly produces the quality measure.

ure 1. In the top panel, σ^2 is shown over time (small portion of a simulated test flight). At time $T = 1.0s$, the pilot issues a specific command, a so-called doublet (fast stick movement from neutral into positive, then negative and back to neutral position; Fig. 4*(lower panel)*). Shortly afterwards ($T = 1.5s$), one control surface of the aircraft (stabilizer) gets stuck ("failure"). Because the system dynamics and the model behavior do not match any more, the neural network produces an augmentation control signal to compensate for this deviation. The network weights are updated according to the given weight update rule. Initially, the network confidence is very high, but as soon as the damage occurs, the network has to adapt. Here, σ^2 of the network output increases substantially, indicating a large uncertainty in the network output. Due to the dynamic training of the network, this uncertainty decreases very quickly.

A second and third pilot command, which is identical to the first one is executed at $T = 11s$, and $T = 17s$, respectively. During that time, the network's confidence is still reduced, but much less than before. This is a clear indication that the network has successfully adapted to handle this failure situation.

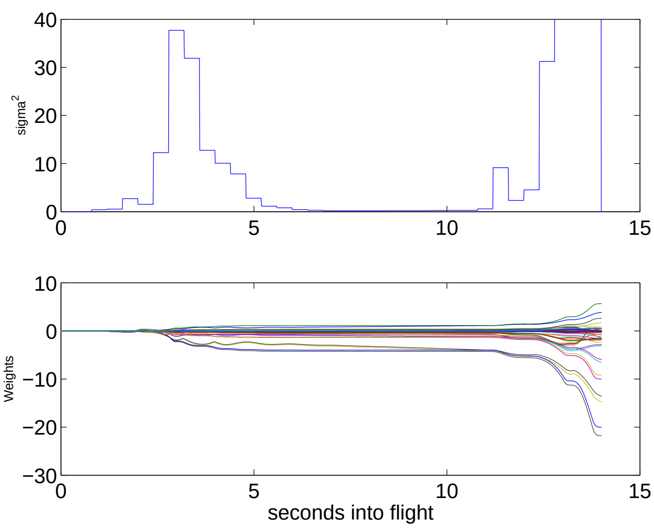


Figure 5. Confidence value σ^2 *(top)* and weights of the network over time (*bottom*). The system has been subjected to the same pilot commands as shown in Fig. 4. However, the learning rule of the neural network is "broken", leading to the inability of the network to converge (weights diverge during the second doublet). An extremely high value of σ^2 (just before the simulation terminated at $T = 14s$) clearly indicates the problem.

Figure 5 shows the results of a simulation with the same pilot inputs as before. However, the network weight update rule was corrupted on purpose, such that the network was not able to adapt toward a solution. This behavior, which leads to an uncontrollable system after the second pilot command (which also causes termination of the simulator run), can be clearly seen in the top panel of Figure 5. The increase of σ^2 during the first doublet is roughly the same as before, although the network takes longer to gain confidence there. However, during the second doublet, the weights of the network diverge (due to the corrupted update rule), leading to large and erratic outputs. The value of σ^2 reaches a peak value of approximately 140, before the simulation has to be stopped. This (albeit extreme) scenario demonstrates the tool's ability to detect—in real time—situations where the neural network is not behaving correctly. Thus, the tool output can be used to activate emergency measures (e.g., reverting to a different controller, cockpit annunciation, etc.).

Envelope Tool

The Confidence Tool calculates the current performance σ^2 of the neural network. This measure, however, gives no indication on the network performance in subsequent time steps or in the face of small perturbations of the input values. Because the NN is a non-linear function approximator, the accuracy (and thus the performance) can change drastically due to small changes in the input values. We therefore define the notion of a *safety envelope* of the NN behavior around the current point of operation (i.e., current input $\mathbf{x}$). This safety envelope is a region around $\mathbf{x}$, where the corresponding network performance $\sigma^2_{\mathbf{x}}$ is not larger than $\sigma^2_{\mathbf{x}} + \Theta$ for a given threshold Θ. Intuitively, the safety envelope is a "dish" around the current point in the input space. Figures 6, 7 show a two-dimensional example. Figure 6 is a graphical representation of the underlying "true" function, which the neural network has to approximate. This function has two modes, based upon the value of one input parameter. A typical example for such a behavior is the change of the system dynamics when the sound barrier is reached. In our example, the network has been trained only on mode 1 (the front part of the surface). Figure 7 shows the performance σ^2 over the two inputs. The curvature of this surface strongly depends on the underlying process and the training of the network. It is easy to see that for the area, where the network has been trained, σ^2 is small. σ^2 increases in areas, where the NN has not been trained (in our case, mode 2). The safety envelope is shown as the area in black at the bottom of this figure.

The safety envelope can provide valuable information about the *estimated* performance of the NN, since it provides performance values for possible future situations. For example, if Figure 7 depicts a situation of a climbing aircraft, then we can expect a similar performance in the near future for a slightly higher altitude. Here, we assume that the network is fixed. This is a reasonable assumption, because the online learning processes are slow compared to the basic update cycle. The shape and size of the safety envelope around the current point of operation thus provides a distance from the current point to the border of the safety envelope. To ensure reliable operation, this distance should be reasonably large. Combined with the value of $\sigma^2_{\mathbf{x}}$, we can get one of the three situations, shown in Figure 8 for a one-dimensional representation. The current point of operation and the safety envelope (an interval) are

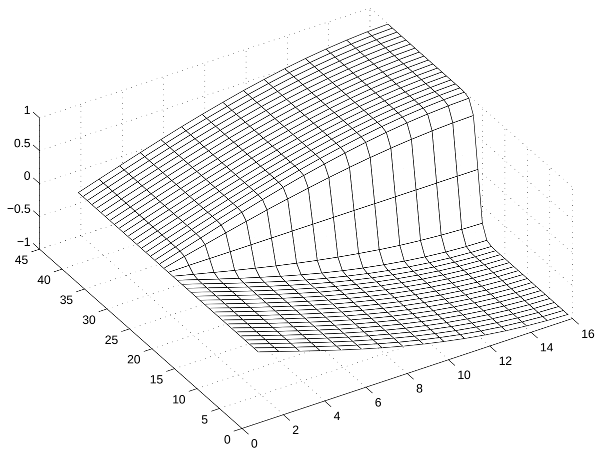


Figure 6. Example function with two inputs (airspeed and altitude) and to modes. The network has been trained on mode 1 (front) only.

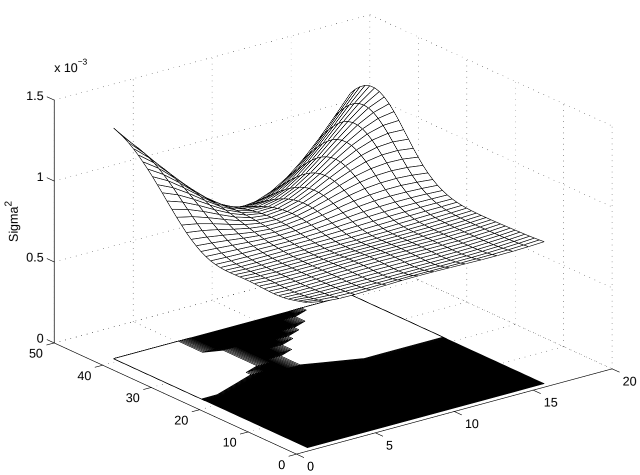


Figure 7. Performance surface and safety envelope for a neural network with two inputs (e.g., airspeed and altitude): Performance measure σ^2 and safety-envelope (black). The network has been trained with selected data points from the function in Fig. 6.

marked. In situation (A, left), we have a large $\sigma^2_{\mathbf{x}}$, i.e., a relatively bad performance. The large safety envelope indicates that in the vicinity of $\mathbf{x}$, no improvement (or degradation) of performance is to be expected. This situation typically shows up in scenarios, where the network still has to adapt toward a solution. However, small disturbances and changes in the plant behavior will not cause a catastrophic loss in network performance. Situation (B, middle) is a much better scenario. Here, the current performance is good, and the safety envelope is wide. Situation (C, right) shows a very good network performance at the current point of operation. However, small disturbances can result in a drastic loss of network performance (large σ^2) which could lead to dangerous (unstable) situations. Such a situation is a clear indication that the network either cannot learn the current situation, or the network has been overtrained. The latter situation usually results from a design error.

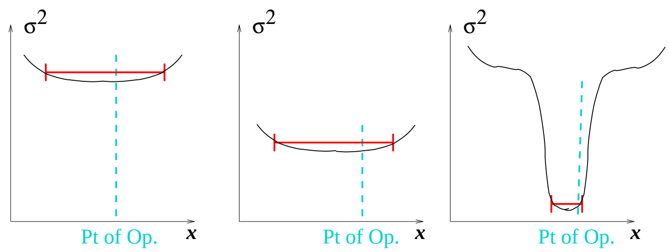


Figure 8. Three scenarios for the safety envelope

5. CONCLUSIONS

Research has shown that neuro-adaptive controllers exhibit many advantages (e.g., better performance, controllability of aircraft despite a structural damage) and can substantially save development costs. However, their practical applicability in safety-critical areas is limited, because traditional V&V methods fail, and there exist no methodology for providing safety and performance guarantees. In this paper, we have discussed the specific requirements for V&V of adaptive systems. Because mathematical analysis often produces results which are too weak, analysis has to be augmented by tests, which take the process of adaptation into account. Due to the requirement of being able to adapt toward unforeseen events, monitoring the performance after deployment is necessary. To this end, we have developed two tools: the Confidence Tool calculates the current network performance, the Envelope Tool estimates a safety envelope with adequate performance. Although the confidence tool will be flight tested in the near future, this approach only marks the beginning of novel V&V approaches for adaptive control. We will extend our work to address the relationship between the network performance (i.e., the outputs of the confidence and envelope tool) and the handling quality of the controlled system. An estimate of the expected aircraft handling quality provides a kind of feedback that can be directly assessed by the pilot. Our work indicates that Bayesian statistical approach for V&V is an effective approach and further research in this field is worth pursuing.

6. ACKNOWLEDGMENTS

This work is in part sponsored by the NASA IFCS project (SMART-T) at NASA DFRC and the NASA Office of Safety and Mission Assurance (OSMA).

REFERENCES

[1] Ch. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon-Press, Oxford, 1995.

[2] A. Calise and R. Rysdyk. Nonlinear adaptive flight control using neural networks. *IEEE Control Systems Mag-*

azine, 21(6):14–26, 1998.

[3] P Gupta, K. Loparo, J. Schumann, and F. Soares. Verification and validation methodology of real-time adaptive neural networks for aerospace applications. In *International Conference on Computational Intelligence for Modeling, Control, and Automation*, 2004.

[4] P. Gupta and J. Schumann. A tool for verification and validation of neural network based adaptive controllers for high assurance systems. In *Proceedings High Assurance Software Engineering (HASE)*. IEEE, 2004.

[5] S. Jacklin, M. Lowry, J. Schumann, P. Gupta, J. Bosworth, E. Zavala, J. Kelly, K. Hayhurst, C. Belcastro, and C. Belcastro. Verification, validation and certification challenges for adaptive flight control systems software. In *AIAA Guidance Navigation and Control Conference and Exhibit*, 2004.

[6] M. Norgaard, O. Ravn, N.K. Poulsen, and L. K. Hansen. *Neural Networks for Modeling and Control of Dynamic Systems*. Springer, 2002.

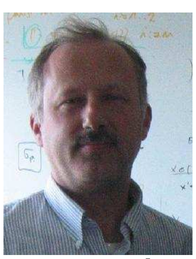
[7] Rumelhart, McClelland, and the PDP Research Group. *Parallel Distributed Processing*. MIT Press, 1986.

[8] R. Rysdyk and A. Calise. Fault tolerant flight control via adaptive neural network augmentation. *AIAA American Institute of Aeronautics and Astronautics*, AIAA-98-4483:1722–1728, 1998.

[9] J. Schumann and P. Gupta. On verification & validation of neural network based controllers. In *Proceedings of the Conference on Engineering Applications in Neural Networks (EANN 03)*, pages 40–47, 2003.

[10] J. Schumann and S. Nelson. Toward V&V of neural network based controllers. In *Proceedings WOSS (Workshop on Self-Healing Systems, 2002*, pages 67–72. ACM Press, 2002.

[11] J. Schumann and P. Gupta. Monitoring the performance of a neuro-adaptive controller. In *Proceedings of the 24th International Workshop on Bayesian Inference and Maximum Entropy Methods in Sciences and Engieering, MAXENT 2004*, 2004.



Dr. Johann Schumann *(PhD 1991, habilitation degree 2000) is a Senior Scientist in the Automated Software Engineering Group, NASA Ames. He is engaged in research on verification and validation of adaptive controllers and learning software. He is also working on automatic generation of navigation and state estimation code. Dr. Schumann is author of a book on theorem proving in software engineering and has published more than 60 articles on automated deduction and its applications, automatic program generation, and neural network oriented topics.*

Dr. Pramod Gupta *(PhD), is a member of Automated Software Engineering Group at NASA Ames Research Center. Presently, he is involved in verification and validation of neural network based adaptive controllers. His research interests are neural networks, modeling, control and robotics and he has published in these area.*

Steven A. Jacklin *received the B.S. degree from the University California, Berkeley in 1978, and the M.S.M.E and Eng.D degrees from Stanford University in 1980 and 1985, respectively. He was a research assistant at Stanford University and joined NASA in 1980. He has authored more than 40 publications on system identification, vibration control, and the reduction of helicopter noise using individual blade control. He has served as project director of 5 major NASA programs in the National Aerodynamics Research Complex, has been the recipient of 11 NASA awards, 3 NASA Group Achievement Awards, other awards including the Agusta Gruppo International Fellowship Award, and American Helicopter Society Velkoff Award for Best Technical Paper. He is a senior member of the AIAA, member of the IEEE, and member of the IEEE Control Systems Society.*