Performance Comparison of Different Neural Augmentation for the NASA Gen_2 IFCS F-15 Control Laws

M.G. Perhinschi*, J. Burken#, M.R. Napolitano*, G. Campa*, M. L. Fravolini+
* Department of Mechanical and Aerospace Engineering,
West Virginia University, Morgantown, WV 26506/6106
NASA Dryden Flight Research Center
+ University of Perugia, Perugia, Italy

Abstract

This paper describes the results of a study focused on comparing the performance of three different neural augmentations of the dynamic inversion-based control laws used for fault tolerant purposes on the NASA IFCS F-15 aircraft. The performance of the specific neural algorithms, the Extended Minimal Resource Allocating Networks algorithm, the Single Hidden Layer neural network, and the SigmaPi neural network have been compared. The comparison has been conducted in terms of specific parameters relative to the tracking of desirable handling qualities following the injection of simulated failures on the actuators of the right stabilator and the left canard of the NASA F-15 aircraft. The simulation results have shown that all three neural networks have promising performance with the Extended Minimal Resource Allocating Networks algorithm slightly outperforming the other algorithms.

List of Acronyms

EMRAN	Extended Minimal Resource Allocating Networks
IFCS	Intelligent Flight Control System
NLDI	Non Linear Dynamic Inversion
NN	Neural Network
SHL	SingleHidden Layer
WVU	West Virginia University

1. Introduction

The main goal of the Intelligent Flight Control System (IFCS) F-15 program[1] is to develop and evaluate in flight control schemes allowing the pilot to cope with the occurrence of a primary control surface failure. The control laws considered in this study - which are a general form of the control laws actually implemented in the IFCS F-15 flight computer - are based on a non linear dynamic inversion (NLDI) scheme augmented with a neural network (NN) to compensate inversion errors and changes in aircraft dynamics due to damage or failure of primary control surfaces. Starting from a well known architecture[2-6], this paper investigates the performance of three different NNs: the Extended Minimal Resource Allocating Networks[7] (EMRAN) algorithm, the Single Hidden Layer[4] (SHL) NN, and the SigmaPi[8]. Their performance is compared in different simulations involving nominal flight conditions, as well as stabilator and canard failures.

2. The Nonlinear Dynamic Inversion-Based Control Laws

The general scheme, named *Gen-2* within the IFCS program, is based on an adaptive neural controller canceling the errors associated with the dynamic inversion of the model. This control strategy[2-4] has been selected to provide consistent handling qualities without requiring the level of computational effort associated with gain-scheduling and/or system identification. In addition, desired handling qualities are achieved with *ad hoc* reference models.

Flight commands are generated by the pilot through longitudinal and lateral stick ($\delta_{lon_{stick}}$, $\delta_{lat_{stick}}$) and pedals ($\delta_{dir_{pedal}}$). These displacement commands are converted[2-6] into corresponding roll, pitch, and yaw rate commands (p_{com}, q_{com}, r_{com}). The reference model provides filtered rate commands (p_{ref}, q_{ref}, r_{ref}) and acceleration commands ($\dot{p}_{ref}$, $\dot{q}_{ref}$, $\dot{r}_{ref}$) using first order roll rate and second order pitch and yaw rate transfer functions.

The inputs to dynamic inversion ($\dot{p}_c$, $\dot{q}_c$, $\dot{r}_c$) are computed using the expression:

$$\begin{bmatrix} \dot{p}_c \\ \dot{q}_c \\ \dot{r}_c \end{bmatrix} = \begin{bmatrix} U_p \\ U_q \\ U_r \end{bmatrix} - \begin{bmatrix} U_{p_{ad}} \\ U_{q_{ad}} \\ U_{r_{ad}} \end{bmatrix} \quad (1)$$

where ($U_{p_{ad}}$, $U_{q_{ad}}$, $U_{r_{ad}}$) are augmentation commands generated by adaptive NNs in order to compensate for the estimated errors e_p, e_q, e_r from the difference between reference and plant angular rates. Furthermore, the pseudo control acceleration commands (U_p, U_q, U_r) are computed with the following expressions:

$$U_p = \left(K_{p_p} + \frac{K_{i_p}}{s} \right) \cdot e_p + s \cdot p_{ref} \quad (2)$$

$$U_q = \left(K_{p_q} + \frac{K_{i_q}}{s} \right) \cdot e_q + s \cdot q_{ref} \quad (3)$$

$$U_r = \left(K_{p_r} + \frac{K_{i_r}}{s} \right) \cdot e_r + s \cdot r_{ref} \quad (4)$$

where K_p and K_i are, respectively, proportional and integral constants.

Dynamic inversion, is then used to determine the necessary control surface deflections ($\delta_a, \delta_e, \delta_r$). Initially, control surface commands ($\delta_{a_{com}}$, $\delta_{e_{com}}$, $\delta_{r_{com}}$) are obtained with the following equation:

$$\begin{bmatrix} \delta_{a_{com}} \\ \delta_{e_{com}} \\ \delta_{r_{com}} \end{bmatrix} = B^{-1} \begin{bmatrix} \dot{p}_c - L_1 \\ \dot{q}_c - M_1 \\ \dot{r}_c - N_1 \end{bmatrix} \quad (5)$$

where B is the state space system control matrix and the terms ($\dot{p}_c - L_1$, $\dot{q}_c - M_1$, $\dot{r}_c - N_1$) are the differences between input acceleration commands and actual plant acceleration contributions (L_1, M_1, N_1). These plant contributions are function of inertial and geometric chareacteristics, aerodynamic derivatives, angular rates, and aerodynamic angles. Finally, the control surface actual deflections are computed from $\delta_{a_{com}}$, $\delta_{e_{com}}$, $\delta_{r_{com}}$ in order to include the modeling of the actuator dynamics (first and second order transfer functions).

3. The Extended Minimal Resource Allocating Networks Algorithm

The first NN considered in this study is the *Extended Minimal Resource Allocating Networks*[7] (EMRAN) algorithm. The EMRAN algorithm is a more powerful version of the standard MRAN. The main architecture[9] features growing and pruning mechanisms; moreover, the parameters are updated following a "winner takes it all" strategy. The extended algorithm allows only the parameters of the most activated neurons to be updated, while all the others are left unchanged. In other words, the algorithm allocates resources (neurons) in order to decrease the estimation error in regions of the state space where the mapping accuracy is poor. This strategy implies a significant reduction of the number of parameters to be updated on-line; thus, it is particularly suitable for on-line applications[10].

For Gaussian basis functions the estimate is computed with the expression:

$$\hat{y}(x,\theta) = \sum_{i=1}^{M} w_i e^{\left(\frac{|x-\mu_i|^2}{2\sigma_i^2} \right)} \quad (6)$$

where x is the input vector, θ is the set of parameters to be tuned by the learning algorithm including the weight w, the gaussian center positions μ, and the variances σ. A new neuron is initiated if 3 distinct criteria are satisfied: the estimation error, the windowed estimation error, and the distance from the input to the nearest center must be larger than selected thresholds:

$$e_1(k) = y(k) - \hat{y}(k) > E_1 \quad (7)$$

$$\frac{1}{N} \sqrt{\sum_{i=1}^{M} e(k-i)} > E_2 \quad (8)$$

$$d_{BMU} = \min_{j=1,\ldots,neu} \left(\left\| c(k) - x(k) \right\| \right) > E_3 \quad (9)$$

If this is the case, the center, variance, and weight of the new neuron at iteration k are given by, respectively:

$$\mu_{M+1}(k) = x(k) \quad (10)$$

$$\sigma_{M+1} = \lambda \inf_{j=1}^{M} \left\| x(k) - \mu_j(k) \right\| \quad (11)$$

$$w_{M+1}(k) = e(k) = y(k) - \hat{y}(k) \quad (12)$$

When one of the criteria is not met the tunning parameters are updated using the relationship:

$$\theta(k+1) = \theta(k) - \eta \left. \frac{\partial \hat{y}(k)}{\partial \theta(k)} \right|_{(k)} \cdot e(k) \quad (13)$$

whre $e(k)$ is the estimation error and η is the learning rate.

A combination of an ADALINE[11] and an EMRAN network (A+EMRAN) working in parallel has been implemented on each of the three channels. This approach has been adopted to achieve good performance in the presence of large non-linearities without the computational burden on operation in areas without non-linearities.

4. The Single Hidden Layer Neural Network Algorithm

The second neural algorithm considered is the Single Hidden Layer[2,3,4] (SHL) NN. The output of the network is given by the relationship:

$$y_i = \sum_{j=1}^{m} \left[w_{ij} \sigma \left(\sum_{k=1}^{n} v_{jk} x_k + \theta_{vj} \right) + \theta_{wi} \right], \quad i = 1, 2, \ldots p \quad (14)$$

where w_{ij} are the interconnection weights between the hidden layer and the output layer, v_{jk} are the interconnection weights between the input layer and the hidden layer, and θ_{vj}, θ_{wi} are bias terms. The activation potential a is used to compute the activation function of the form:

$$\sigma(\xi) = \frac{1}{1+e^{-a\xi}} \tag{15}$$

Based on a Lyapunov analysis the weights updating laws are given by[2,3,4]:

$$\begin{aligned} \dot{W} &= -\gamma_W \left[\left(\sigma - \sigma' V^T x \right) e_x^T + \lambda_W \left\| e_x \right\| W \right] \\ \dot{V} &= -\gamma_V \left[x e_x^T W^T \sigma + \lambda_V \left\| e_x \right\| V \right] \end{aligned} \tag{16}$$

where e_x are state errors and $\gamma_W, \gamma_V, \lambda_W, \lambda_V$ are design parameters (learning rates and e-modification factors).

5. The SigmaPi Neural Network

The third type of NN investigated is a two-layer Sigma-Pi NN[5,6,8] for each angular acceleration ($\dot{p}, \dot{q}, \dot{r}$). These NNs use proportional and integral acceleration errors (U_{p_error}, U_{q_error}, U_{r_error}) for on-line learning purpose:

$$U_{x\_error} = \left(K_{p_{x\_error}} + \frac{K_{i_{x\_error}}}{s} \right) \cdot e_x \qquad x = p, q, or\ r \tag{17}$$

Inputs to the networks are pseudo control acceleration commands (U_p, U_q, U_r), bias terms, and sensor feedback. For each channel three terms C_1, C_2 and C_3 are computed as functions of input variables and previous-step network outputs ($U_{p_{ad}}$, $U_{q_{ad}}$, $U_{r_{ad}}$):

$$C_{1_x} = bias, V_t, V_t^2, h \tag{18}$$

$$C_{2_p} = bias, \frac{1-e^{-\left(U_p - U'_{p_{ad}}\right)}}{1+e^{-\left(U_p - U'_{p_{ad}}\right)}}, p, r, p_{ref} \tag{19}$$

$$C_{2_q} = bias, \frac{1-e^{-\left(U_q - U'_{q_{ad}}\right)}}{1+e^{-\left(U_q - U'_{q_{ad}}\right)}}, q, q_{ref} \tag{20}$$

$$C_{2_r} = bias, \frac{1-e^{-\left(U_r - U'_{r_{ad}}\right)}}{1+e^{-\left(U_r - U'_{r_{ad}}\right)}}, p, r, r_{ref} \tag{21}$$

$$C_{3_x} = bias, \alpha, \beta \qquad x = p, q, r \tag{22}$$

All the neuron outputs are summed and multiplied to each other - hence the name of the NN topology. The outputs of the NNs are the control augmentation commands defined as:

$$U_{ad} = W^T f\left(C_1, C_2, C_3\right) \tag{23}$$

where *f* is computed from each signal inputs using a nested Kronecker product. The network weights W are determined by an adaptation law:

$$\dot{W} = -G\left(U_{error} \cdot f + L\left|U_{error}\right| W\right) \tag{24}$$

where G and L are user selected specific gains.

6. Results of the Comparative Study

The simulations have been performed using the WVU IFCS F-15 flight simulator[12] at Mach = 0.75 and altitude = 20000 ft.

Six test cases have been considered:

Case #1: nominal conditions, no failure, doublets on all three channels;

Case #2: right stabilator locked at -4 degree at t=40 s, no pilot input;

Case #3: right stabilator locked at -4 degree at t=40 s, doublets on all three channels, before and after the failure;

Case #4: right stabilator locked at +8 degree at t=40 s, no pilot input;

Case #5: left canard locked at -4 degree at t=40 s, no pilot input;

Case #6: left canard locked at -4 degree at t=40 s, doublets on all three channels, before and after the failure.

To assess the performance of the different approaches, the mean values and the standard deviations of the errors between reference model angular rates and actual aircraft angular rates have been computed. Results for the roll, pitch, and yaw channels are presented in Tables 1-3, respectively.

The NLDI-based controller is coping with the failure in all the cases investigated except Case #4. A similar behaviour is recorded for the SigmaPi augmented version. The EMRAN and SHL augmented versions can handle all the cases investigated.

On the roll and pitch channel, the best performance is achieved using the EMRAN NN. SHL ranks second, followed by the SigmaPi.

On the yaw channel, the performance of the SHL NN augmented control laws is the best.

As an example, Figure 1 shows the time histories of the square of the pitch rate tracking errors for the three NNs on and with the NN off for the stabilator failure case (case #2). A similar chart is presented in Figure 2 for the canard failure case (case #5). The time histories of the square of the roll rate tracking errors for the same two cases are presented in Figure 3 and 4 respectively.

7. Conclusions

Three different NNs have been compared as alternatives for the augmentation of fault tolerant NLDI-based control laws.

Different failure scenarios of the stabilator and the canard have been investigated.

Each of the considered neural algorithms provide successful augmentation of the NLDI-based control laws.

The EMRAN algorithm slightly outperforms the other two algorithms in terms of the angular rate tracking errors, while requiring a lower computational effort.

Acknowledgements

Support for the 1st, 3rd, and 4th author has been provided through the NASA Dryden grant NCC4-159.

References

1. "Intelligent Flight Control: Advanced Concept Program – Final Report", The Boeing Company, BOEING-STL 99P0040, May 1999.
2. Calise A. J., Lee S., Sharma M., "Direct Adaptive Reconfigurable Control of a Tailless Fighter Aircraft", *Proc. of the 1998 AIAA Guidance, Navigation and Control Conference*, Boston MA, August 1998, AIAA 98-4108
3. Rysdyk R.T., Calise A. J., "Fault Tolerant Flight Control via Adaptive Neural Network Augmentation", AIAA 98-4483, August 1998.
4. Calise A. J., Sharma M., "Adaptive Autopilot Design for Guided Munitions", *AIAA Journal of Guidance, Control and Dynamics, vol. 23 no 5,* 2000
5. Kaneshige J., Bull J., Totah J.J., "Generic Neural Flight Control and Autopilot System", AIAA Paper 00-4281, Denver, Co, August 2000.
6. Kaneshige J., Gundy-Burlet K., "Integrated Neural Flight and Propulsion Control System", AIAA Paper 01-4386
7. Lu Y., Sundararajan N., Saratchandran P., "Analysis of Minimal Radial Basis Function Network Algorithm for Real Time Identification of Nonlinear Dynamic Systems", IEEE *Proceedings on Control Theory and Application 2000*, vol.4, no 147, pp.476.
8. Shin Y., Ghosh J., "The Pi-Sigma Network: An Efficient Higher-Order Neural Network for Pattern Classification and Function Approximation", IEEE 7803-0164, 1991.
9. Campa G., Fravolini M.L., Napolitano M.R., "A Library of Adaptive Neural Networks for Control Purposes", IEEE International Symposium on Computer Aided Control System Design, September 18-20, 2002 Glasgow, Scotland, UK.
10. Fravolini M.L., Campa G., Napolitano M.R., La Cava M., "Comparison of Different Growing Radial Basis Functions Algorithms for Control Systems Applications", *Proc. of the 2002 American Control Conference*, Anchorage AK, May 2002
11. Polycarpou M., "On-Line Approximators for Nonlinear System Identification: A unified Approach", *Control and Dynamic Systems Series*, vol. 7, Ac. Press, Jan. 1998
12. Perhinschi M.G., Campa G., Napolitano M.R., Lando M., Massotti L., Fravolini M.L., "A Simulation Tool for On-line Real Time Parameter Identification", *Proceedings of the 2002 AIAA Modeling and Simulation Technologies Conference*, Monterey, Ca, August 2002.

	NN off		EMRAN		SigmaPi		SHL	
	Mean [deg/s]	STD	Mean [deg/s]	STD	Mean [deg/s]	STD	Mean [deg/s]	STD
Case #1	0.0182	4.4121	-0.0006	3.4229	-0.0086	6.2877	-0.0003	4.0883
Case #2	-0.1369	1.3007	-0.0002	0.8416	-0.0411	1.6936	-0.0003	0.9167
Case #3	-0.0974	4.9750	-0.0028	4.0092	-0.0086	6.2877	-0.0000	5.3922
Case #4	unstable	unstable	-0.0004	1.3014	unstable	unstable	-0.0024	1.5490
Case #5	0.0133	0.1694	-0.0001	0.1640	0.0087	0.1893	-0.0001	0.1549
Case #6	0.0303	4.2208	0.00002	3.4009	0.0054	3.7050	-0.0002	3.0629

Table 1. Performance Parameters on the Roll Channel for the Three NNs

	NN off		EMRAN		SigmaPi		SHL	
	Mean [deg/s]	STD	Mean [deg/s]	STD	Mean [deg/s]	STD	Mean [deg/s]	STD
Case #1	0.0399	0.5059	0.0055	0.5948	0.0100	0.9903	-0.00002	0.6785
Case #2	-0.0651	0.8689	-0.00001	0.4959	-0.0180	0.7122	-0.00003	0.5031
Case #3	-0.0446	1.0112	-0.0334	0.8536	0.0100	0.9903	0.0000	0.9599
Case #4	unstable	unstable	0.9113	0.8593	unstable	unstable	0.5288	0.8285
Case #5	0.0502	0.2589	-0.0000	0.1813	0.0075	0.2806	-0.0000	0.1900
Case #6	0.0037	0.6640	0.0005	0.4781	0.0001	0.6267	-0.0001	0.5094

Table 2. Performance Parameters on the Pitch Channel for the Three NNs

	NN off		EMRAN		SigmaPi		SHL	
	Mean [deg/s]	STD	Mean [deg/s]	STD	Mean [deg/s]	STD	Mean [deg/s]	STD
Case #1	-0.0040	1.6987	0.0077	2.5394	-0.0033	1.7476	-0.0000	0.8324
Case #2	-0.0178	0.2168	-0.0000	0.1577	-0.0088	0.2506	-0.0001	0.1676
Case #3	-0.0093	1.6516	0.0023	2.5565	-0.0033	1.7476	-0.0001	1.0185
Case #4	unstable	unstable	-0.00003	0.2715	unstable	unstable	0.00006	0.3673
Case #5	-0.0156	0.1559	-0.0001	0.1531	-0.0097	0.1603	-0.0001	0.1518
Case #6	-0.0192	1.7560	0.0049	2.5586	-0.0112	1.8180	-0.0001	1.0121

Table 3. Performance Parameters on the Yaw Channel for the Three NNs

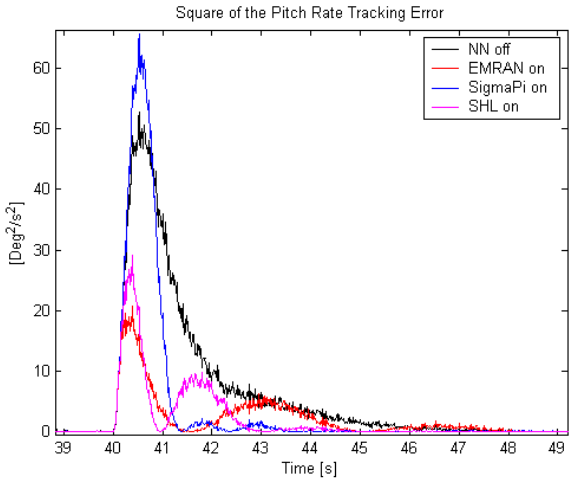

Figure 1. Comparison of the three NNs - Stabilator failure at t=40s - Pitch Channel

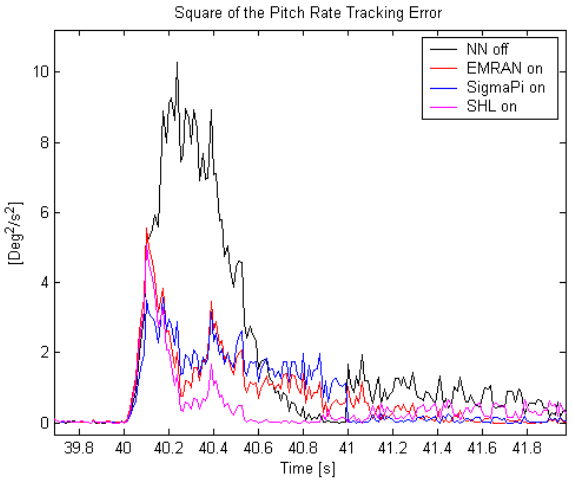

Figure 2. Comparison of the three NNs - Canard failure at t=40s - Pitch Channel

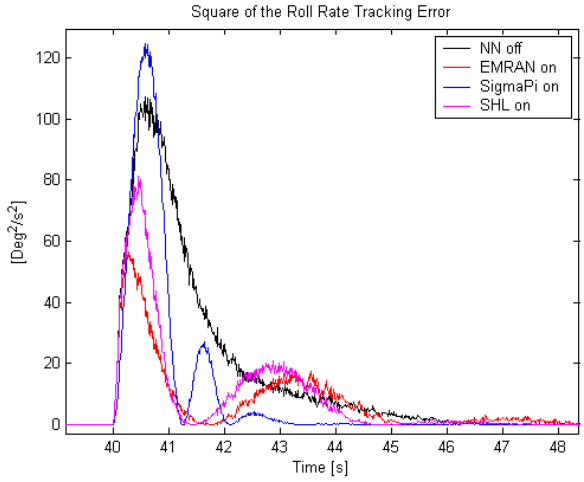

Figure 3. Comparison of the three NNs - Stabilator failure at t=40s - Roll Channel

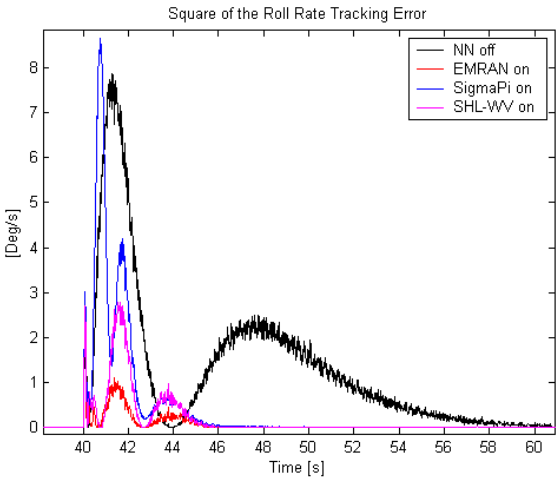

Figure 4. Comparison of the three NNs - Canard failure at t=40s - Roll Channel