Direct Comparison of Neural Network, MPVSC and Fuzzy Logic Controllers

Praveen Joshi and John Valasek
Texas A&M University

AIAA Atmospheric Flight Mechanics Conference
Portland, OR
10 August 1999
Introduction

- For better maneuverability and agility, high angle-of-attack flights are desired.
- Stability of fighter aircraft is greatly diminished in this flight regime due to:
  1. Unstable pitch break
  2. Adverse propulsion/airframe integration
  3. Loss of static and dynamic lateral/directional stability
  4. Reduced control effectiveness
- Figure 1 shows available and needed control power for typical yaw maneuvers for a generic fighter aircraft
- Various techniques have been suggested for control augmentation
TYPICAL YAW CONTROL REQUIREMENTS FOR MANEUVERING

A promising solution for stability augmentation.

Vortices are blown tangentially from aircraft nose.

Affects the vortex structure on forebody, resulting in side force.

Left wing blowing creates +ve side force and vice-versa.

Available

Required

Maximum lift

Angle of attack

Yaw control

Joshi and Valasek

AIAA-99-4279-3

Aerospace Engineering
Control Power Augmentation

- Passive Techniques
  1. Wing Fillets
  2. Winglets
- Active Techniques
  1. Rotating Strakes
  2. Tangential Slot Blowing
- Tangential slot blowing is the most promising of all these techniques
- Stability is reduced due to asymmetric vortex shedding known as Von Karman vortex shedding
- Blowing vortices from the aircraft nose counters this effect
Tangential Slot Blowing

- Vortices are blown from the nose of the aircraft in asymmetric fashion.
- Left side blowing results in increased vortex strength over right side wing and vice-versa.
- Hence, for a nose-left divergence, a vortex is introduced on the right side and vice-versa.
- Figure 2 shows the effect of tangential slot blowing on vortices.
X-29A Aircraft

- X-29A aircraft has been modified to test the effectiveness of tangential slot blowing method
- Figure 3 shows the modified X-29A aircraft
Mathematical Description of the Problem

- **Flight Conditions**
  - $M = 0.35$
  - $H = 38000$ ft
  - $\alpha = 40$ deg
  - $V = 338$ ft/sec
  - $q_{\text{bar}} = 37$ psf
  - $\delta_{\text{canard}} = -24.4$ deg
  - $\delta_{\text{strake}} = 12.8$ deg
  - $\delta_{\text{symmetric flaps}} = 20.7$ deg

- The plant model is the generic X-29A LTI lateral/directional model fitted with VFC nozzles, in the normal digital up-and-away mode.

- The control power of VFC is calculated using scale factor which relates control power of VFC nozzles to the control power of the rudder.
Lateral/directional dynamics of the X-29A aircraft and design requirements

\[
\begin{bmatrix}
\dot{\beta} \\
\dot{\rho} \\
\dot{\psi} \\
\dot{\phi}
\end{bmatrix}
= 
\begin{bmatrix}
0.034 & 0.64 & -0.77 & 0.073 & 0 \\
-8.39 & 0.78 & -0.69 & -0.74e-5 & 0 \\
0.14 & 0.063 & -0.11 & -0.17e-5 & 0 \\
0 & 0 & 1 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\beta \\
\rho \\
\psi \\
\phi
\end{bmatrix}
+ 
\begin{bmatrix}
0.0013 & -0.034 & 0.013 \\
0.86 & -0.17 & -0.12 \\
0.092 & -0.12 & 0 \\
0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\delta_{df} \\
\delta_{r} \\
\delta_{fvm}
\end{bmatrix}
= 
\begin{bmatrix}
0.0013 \\
0 \\
-0.12 \\
0
\end{bmatrix}
\delta_{fvm}
\]
Open loop poles of the system are:

\[ \lambda_{1,2} = 0.42 \pm 2.32j, \quad \omega_{\text{DR}} = 2.36 \text{ rad/sec and } \zeta_{\text{DR}} = -0.18 \]
\[ \lambda_{3,4} = -0.069 \pm 0.12j \quad \omega_{\text{ph}} = 0.14 \text{ rad/sec and } \zeta_{\text{ph}} = 0.49 \]
and \( \lambda_5 = 0 \)

Objective is to design a controller which will drive the system from a specified initial sideslip angle to zero value.

Initial angle of 5 degrees is chosen because it is large enough to excite dutch roll without the aircraft going unstable.

Design specifications are:

- 5% settling time in sideslip angle - 5 seconds
- maximum sideslip angle - 10 degrees
- maximum body axis yaw rate - 30 degrees/sec
- maximum bank angle - 30 degrees
- maximum heading angle - 10 degrees
Design Procedure (1)

- There are 2 continuous and one bang-bang control inputs:
  1. Differential flaps (continuous)
  2. Rudder (continuous)
  3. VFC nozzle (bang-bang)
- Design two different controllers for continuous effectors and bang-bang effector separately.
- Begin with constructing a quadratic cost function to weight different states and control activities as dictated by the designer. The cost function is of the following form:

\[
J = \frac{1}{2} \int_{0}^{T} \{X^T Q X + U^T R U\} dt
\]
Design Procedure (2)

- It is important to note at this point that the subsequent design procedure is applicable to a general class of systems with both continuous and bang-bang effectors.
- To illustrate the design procedure more clearly, X-29A lateral/directional dynamics will be used as bench-mark problem for the subsequent treatment.
- First task is to design a controller to control continuous effectors.
- Based on design specifications, following elements are selected for weighting matrices:
  \[ Q_{11} = 32.8 \quad Q_{22} = 2 \quad Q_{33} = 3.65 \quad Q_{44} = 3.65 \]
  \[ Q_{55} = 20 \quad R_{11} = 85 \quad R_{22} = 2.5 \]
On X-29A flight control system, the controller is implemented as a digital controller with a sampling rate of 10 Hz.

This changes the problem from continuous cost function optimization to sampled data regulator. Accordingly, the cost function will change from an integral to a summation:

\[ J_{new} = \sum_{k=1}^{N} X_k^T \hat{Q} X_k + U_k^T \hat{R} U_k + 2 X_k^T M U_k \]

Assuming a full state feed-back, the control law is of the form:

\[ U_k = -K X_k \]

Solving Riccati equation for this problem, we get K matrix as:

\[
K = \begin{bmatrix}
-0.92 & 2.05 & -2.06 & 0.23 & 0.081 \\
5.22 & -2.59 & -4.22 & -0.49 & -2.62
\end{bmatrix}
\]
This controller is implemented in MATLAB-SIMULINK environment and the simulation is run for 10 seconds. Figure 4 shows the time history for the simulation.

- It can be seen that both differential flap and rudder are saturated for more than one and half cycles.
- Also 5% settling time requirement for sideslip angle is not satisfied which shows the limited control power of the continuous effectors and the need for control power augmentation.
- Cost function for 10 second run is 29.875.
- Next task is to design a control methodology for bang-bang (VFC) effector.
- Three different methods will be presented. Neural controller will be discussed at length.
Figure 4 - Design 1 Cost function = 29.875
Only three discrete values \{-1, 0, +1\} are possible for the bang-bang effector.

In MPVSC, a *sub-interval* is selected by the designer and the states are propagated forward in time at the beginning of each interval for the three possible values of bang-bang effector.

The sub-interval is selected by the designer and is usually equal to one sample period \(T\).

Three cost functions arising out of each situation are calculated for the sub-interval and the operation state which generates the lowest cost is selected by the controller for the upcoming sample interval.

Figure 5 shows the block-diagram implementation of this scheme.
Figure 6 Block diagram representation of MPVSC Controller
It is important to note that the continuous effectors are NOT affected by this selection process.

Since the output of bang-bang effectors are constant between two sample intervals, the additional cost because of using them is accounted for by including them in $U_k$ vector and adding one more element to the $R$ matrix.

MPVSC controller was implemented in MATLAB-SIMULINK environment and the system was simulated for $R_{33} = 0.0005$.

Figure 7 shows the time history for 10 second run. The value of cost function is 13.875 which is 54% increase over design 1.

Differential flaps and rudder are saturated only during initial half cycle. Also 5% settling time criterion for sideslip angle is satisfied. Four different pulses are fired.
Figure 9, MPVSC controller time history, cost = 12.875
Fuzzy Logic based Controller (1)

- Fuzzy logic imulates human linguistic process using the degree of truth for the linguistic terms
- Fuzzification, decision making, defuzzification and tuning are the four important steps in fuzzy logic controller design
- Fuzzy logic based controller was designed to control the activity of the bang-bang effectors
- Based on the response analysis of a single pulse, sideslip angle, sideslip angle rate and yaw rate were chosen as the inputs
- Design of membership functions (MF) is done based on the analysis of the existing systems of two controllers, i.e. conventional feedback gain controller and MPVSC
- 7 MF’s were applied to each input placed at equal distance
Fuzzy Logic based Controller (2)

- Figure 8 shows the construction of MF’s
- Fuzzy rules were constructed from response analysis. Figure 9 shows the fuzzy rules expressed in the form of two FAM tables
- Mean of Maxima (MOM) is used as the defuzzification method
- Fuzzy Logic Toolbox provided in the MATLAB-SIMULINK environment was used to construct the controller
- Figure 10 shows the state and control trajectories for a 10 second run
- All the design specification are satisfied and also the VFC nozzle fires only 2 pulses
- The cost function is 12.67 for 10 second run which is 8% improvement over MPVSC
Figure 8, Construction of Membership Functions

Figure 9, Construction of fuzzy rules
Figure 10, Fuzzy Logic Controller time history, cost function = 13.67
REVIEW OF NEURAL NETWORK CONCEPTS

- Taking inspiration and guidance from human information processing, Neural Networks have been successfully used in many areas like:
  1. Signal Processing
  2. Controller Design
  3. Pattern recognition
  4. Medicine
  5. Speech Production and Recognition
  6. Business and Market Trend Predictions
- All neural networks are built using a single neuron as the basic building block
- Since a biological neuron is the model for artificial neurons, we will start with a brief introduction to biological neurons
BIOLOGICAL NEURONS (1)

- Figure 11 shows a typical biological neuron.
- Three components of a neuron are of particular interest, dendrite, soma and axon. Dendrite receives signals from other neurons. These signals are transmitted to soma after multiplying with a scaling factor. Soma sums the signal and after a particular value of the input signal, fires, i.e. sends its own signal through the axons.
- Important characteristics of a neuron can be summed up as:
  1. The processing element receives many signals
  2. Signals may be modified by a weight at the receiving synapse
  3. The processing element sums the weighted inputs
  4. Under appropriate conditions (sufficient input), the neuron transmits a single output
  5. The output of a particular neuron may go to many other neurons
BIOLOGICAL NEURONS (2)

- Other important features are:
  1. Memory is distributed
     a. Long term memory resides in/corresponds to the neurons’s weights
     b. Short term memory corresponds to the signal sent by the neurons
  2. A synapse’s strength may be modified by experience (learning)

Figure 11, A generic biological neuron

Figure 1.3 Biological neuron.
ARTIFICIAL NEURAL NETWORKS

- A single neuron is the basic unit of a neural network.
- A single neuron receives signals from each input, multiplies them by appropriate weights and then gives its own output (firing) based on the total input it receives. Mathematically:
  \[ \hat{y} = f(\vec{y}); \quad \vec{y} = b_0 + \sum_{j=1}^{n} v_j x_j \]

- \( f(.) \) is the activation function of the neuron, which will be discussed later. In the above equation, \( \hat{y} \) is the output of the neuron and \( \vec{y} \) is the input to the neuron. Each \( x_j \) denotes component of input vector and \( v_j \) is weight multiplying each input.

- Several neurons are arranged in a row to build a single layer network.

- Multiple layered network can be created by arranging several rows of neurons. Figure 12 shows a typical multi-layered neural network. A Hidden layer is any layer between input and output layer.
Figure 12, A typical multi-layered neural network
ACTIVATION FUNCTIONS

Following are the most important activation functions for neural networks:

1. **Identity function** – output of identity function is same as its input
   
   \[ f(x) = x \quad \text{for all } x \]

2. **Binary step function** (hard limit) – The output of the neuron is either 0 or 1 depending on the input value. \( \theta \) is the threshold value
   
   \[ f(x) = \begin{cases} 1 & \text{if } x \geq \theta \\ 0 & \text{if } x < \theta \end{cases} \]

3. **Logarithmic Sigmoid** – Output of this function is continuous and lies between 0 and 1
   
   \[ f(x) = \frac{1}{1 + \exp(-x)} \]

4. **Tangential Sigmoid** – Output of this function is continuous and lies between -1 and 1. This and above activation functions are very useful because they are differentiable
   
   \[ f(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \]

\[ a = \text{logsig}(n) \quad \text{Log-Sigmoid Transfer Function} \]

\[ a = \text{tansig}(n) \quad \text{Tan-Sigmoid Transfer Function} \]
A neural network is useful because of its property to generalize an unknown function based on limited data.

A few input vectors (training vectors) and corresponding output vectors (target vectors) are generated skillfully by the designer. Initially, random values are assigned to the weight and bias vectors in a network.

Learning of a neural network implies changing weight and bias values of a network in order to minimize mean square error between network output and target vector.

Several important algorithms exist to achieve the training:
1. Hebbian learning rule
2. Delta learning rule
3. Widrow-hoff learning rule
   - Generalization of Widrow-Hoff rule for multi-layered network leads to the backpropagation learning algorithm
4. Correlation learning rule
TRAINING OF NEURAL NETWORKS (2)

- All the algorithms listed on previous page are useful for a single layer network where network output and desired output (target) are known.
- For a hidden layer of a multi-layered network, the desired output (target) is not known.
- The above mentioned algorithms are modified to overcome this difficulty. This results in following important learning algorithms for a multi-layered network:
  1. Back-propagation learning
  2. Learning with momentum
  3. Levenberg-Marquardt algorithm, etc
- Next section of this talk will discuss a methodology to design a neural network to control the bang-bang effector activity. This method is applicable to a general class of systems but to illustrate the point, X-29A will be used as the bench-mark problem.
NEURAL CONTROLLER
bang-bang effectors

- There are only 3 discrete outputs, i.e. {-1, 0, +1}
- The natural choice for activation function is hardlimit function. But we will not use it as activation function for following reasons:
  1. Hardlimit is not differentiable which means that most of the existing training algorithms cannot be used
  2. As there are only 3 sharp outputs, the network will be susceptible to instabilities even for small deviations from ideal training data
  3. No stability analysis can be performed on networks using hardlimit as activation function
- We will instead use tansig and logsig as the activation functions and will use a threshold value on the output to convert network output into three crisp signals.
- A 3 layered network with symmetric hidden layer is the best choice
- With these general ideas, we now proceed to design a controller for X-29A
As discussed earlier, the first network will be designed using logsig activation function

**NETWORK ARCHITECTURE**

- A symmetric hidden layer is found to be more efficient than one hidden layer. Three different networks with 8, 12 and 16 neurons in each hidden layer were built.
- The network with 12 neurons in hidden layer gave best results hence it was used for further tuning and analysis.
- Three neurons were used in the output layer with logsig as the activation function for each. Output of each neuron lies between 0 and +1 and each neuron represents the relative importance of -1, 0 and +1 in the actual output respectively.

- The three outputs of the network were multiplied by -1, 0 and +1 respectively, added together and passed through *round* function to get a crisp output.
This converts the problem from a discrete output to a continuous output and we can take advantage of the diversity of existing training algorithms.

MATLAB environment was used to initialize and train the network.

For speed and efficiency, *trainlm* algorithm provided in the MATLAB neural network toolbox was used for training.

Momentum learning was used to avoid getting stuck into a local minima.

Momentum constant of 0.9 was used for training.

Performance goal of 1E-3 and 1E-4 were used, which resulted in:
- Performance Goal of 1E-3 -- 20 to 25 epochs for training
- Performance Goal of 1E-4 -- 150 to 200 epochs for training

Performance goal of 1E-4 was selected for its better performance.

Figure 13 shows the SIMULINK implementation of the controller.

Generation is of correct training data is probably the most important thing in neural network design. Hence, special attention was given to generating correct training and target data, which is described next.
Figure 13, SIMULINK implementation of neural controller
To generate the first set of training data, the system was simulated using MPVSC controller and 250 samples of each state variable were taken, at regular interval from the generated trajectory.

- These 250 samples were used as training vectors
- To generate the target vectors, the system equations were integrated, for 0.1 seconds for 3 different cases for bang-bang input of -1, 0 and +1 respectively, starting with each training vector as the initial condition
- Position limits on flap and rudder can be implemented in this offline simulation, which increases the accuracy of all calculations
- Cost function for each of the three cases was calculated for 0.1 seconds and the input which gives lowest cost was selected as network output.

Since there are 3 output layer neurons, The target vector assigned was [1 0 0] for a desired output of -1 and target vectors assigned for desired outputs of +1 and 0 were [0 0 1] and [ 0 1 0] respectively.

Figure 14 shows time history for the system using this controller
Figure 14, time histories for neural controller 1, cost = 13.47
The design of previous neural controller proved the validity of the method. But since the training vectors were generated using a limited portion of input domain, the network cannot be expected to work for arbitrary initial condition. The input domain was scanned into 6*9*7*9 vectors and training data was generated. The target vectors were generated using the same method as above. This network gave a cost function of 13.47 for a 10 second run. It was observed that the system was sensitive to sideslip angle conditions. Hence, the resolution of sideslip angle scanning was increased gradually and networks were designed which reduced to cost function to 13.315 and then 13.251.

The increase in training time (including generation of training data) is:
- First network with 250 training vectors - ~5 minutes
- Final network with fine beta resolution - ~45 minutes

Figure 15 shows time histories for the network with cost function of 13.251.
Figure 15. Time histories for neural network 4, cost = 13.251
IMPROVED TRAINING AND TARGET DATA (1)

- It was observed that reduction in cost function due to sideslip angle resolution increase was becoming bounded.
- Also, we are simulating the system for 0.1 seconds. But the application of a pulse for 0.1 seconds will have effect on the subsequent trajectory which needs to be taken into account while calculating cost function and then deciding target vectors.
- Theoretically, we need to consider the effect of application of a pulse for 0.1 seconds for an infinite time, but the closed loop system is highly stable (as seen from all the previous time histories). Hence it was decided to simulate the trajectory for 2.5 seconds while applying a pulse for initial 0.1 seconds.
- The network designed using this method performed much better giving a cost function value of 12.395 for a 10 second run.
- Figure 16 shows the time history for the system using this controller.
Figure 16, Time histories for neural controller with cost function 12.395
It can be seen from the previous figure that the network fires 5 pulses.

- While doing robustness studies, it was found that the network is susceptible to excessive switching which is undesirable from mechanical considerations. This is probably because more weightage is given to network outputs representing -1 and +1.

A new network using tansig as the activation function was designed.

- The architecture and training method was same but the output layer now has only one neuron.
- Output of the network lies between -1 and +1 so round function was used to convert it to a crisp signal.
- The performance was slightly degraded but this network reduced excessive switching.

Figure 17 shows time histories for system using this network.
Figure 17, Time histories for neural controller with tansig, cost = 12.65
CONTROLLER STABILITY (1)

Stability

- Stability of a system is generally thought of as the property of the system which ensures bounded output when bounded input is given to the system.
- Lyapunov Stability - A solution $u(t)$ of a system of differential equations is said to be Lyapunov stable if, given a small number $\varepsilon > 0$, there exists a number $\delta = \delta(\varepsilon) > 0$ such that any other solution $v(t)$ for which $\|u-v\| < \delta$ at time $t = t_0$ satisfies $\|u-v\| < \varepsilon$ for all $t > t_0$.
- In the first method, the system is linearized about an equilibrium point and conclusions are drawn based on linearized system analysis.
- In the second method, a scalar function called Lyapunov function is constructed for stability analysis.
Lyapunov Second Method

- Let $V(X)$ be a scalar function with continuous first partial derivatives for all $X$. If,
  1. $V(X) > 0$ for all $X \neq 0$
  2. $\dot{V}(X) < 0$ for all $X \neq 0$
  3. $V(X) \to \infty$ as $||X|| \to \infty$

- If $V(X)$ satisfies all 3 conditions listed above, then it is called Lyapunov function

- A system is globally asymptotically stable if a Lyapunov function exists for the systems

- Lyapunov function for a system is not unique
STABILITY OF SYSTEMS
Continuous and bang-bang effectors

Stability Analysis

The stability analysis begins with the selection of Lyapunov function

\[ V(X) = X'PX \quad \text{where } P \text{ is a positive definite matrix} \]

Differentiating the above equation, we get

\[ \dot{V}(X) = \dot{X}'PX + X'PX \]

Substituting for \( \dot{X} \) from the system equation,

\[ \dot{X} = A_{cl}X + B_{bb}U_2 \quad \text{where } A_{cl} \text{ is the closed loop } A \text{ matrix and } U_2 \text{ is the bang – bang control value} \]

After simplification and substituting \( U_2 = 1 \), we get

\[ \dot{V}(X) = X'[A_{cl}'P + PA_{cl}]X + X'PB_{bb} + B_{bb}'PX \]

For \( V \) to be a Lyapunov function, \( \dot{V} \) must be negative definite

The term in the square brackets is negative for closed loop system is stable

So the condition for \( \dot{V} \) to be negative definite is

\[ |X'[A_{cl}'P + PA_{cl}]X| > |X'PB_{bb} + B_{bb}'PX| \]
STABILITY ANALYSIS

Neural Controller

- Starting with the condition stated on previous page, the steps to prove the stability of the neural controller are listed below:
  - Starting with zero value, each input variable is increased independently till the network value crosses 0.5. This gives 4 different vectors, i.e., \([\beta_0, 0 0 0]\), \([0 p_0 0 0]\), \([0 0 r_0 0]\) and \([0 0 0 \phi_0]\) after which the controller starts firing a pulse
  - A suitable P matrix is then sought to obtain which will satisfy following conditions
    - Stability condition is satisfied for \(X = [\beta_1 0 0 0]\) where \(|\beta_1| < |\beta_0|\)
    - Stability condition is satisfied for \(X = [0 p_1 0 0]\) where \(|p_1| < |p_0|\)
    - Stability condition is satisfied for \(X = [0 0 r_1 0]\) where \(|r_1| < |r_0|\)
    - Stability condition is satisfied for \(X = [0 0 0 \phi_1]\) where \(|\phi_1| < |\phi_0|\)
  - This means that we are trying to find a P matrix which will ensure that the system is stable outside a 4-D box bounded by \(|\beta_1|, |p_1|, |r_1|\) and \(|\phi_1|\).
  - If we can show that inside the this 4-D box, if the controller does not fire any pulse, then because of stability of the closed loop system, we can conclude that the system is stable with bang-bang effectors
Using the knowledge about system dynamics and some algebraic simplifications, we get a following P matrix

\[
P = \begin{bmatrix}
150 & 0 & 4 & 0 \\
0 & 5.5 & 1 & 0 \\
0.2 & 0.2 & 0.008 & 2.0
\end{bmatrix}
\]

This P matrix satisfies all the conditions stated above.

Now we have to prove that inside the 4-D box, the network does not fire a pulse. The best way is to take the derivative of the output with respect to each input, find maxima and minima and show that the |output| < 0.5.

Because the network uses three layers of tansig functions, it is not possible to analytically differentiate the output.
STABILITY ANALYSIS
Neural Controller

- So, we divide the 4-D box into a grid of $5 \times 5 \times 5 \times 5 = 625$ elements and simulate the network at these 625 points.
- We check the maximum and minimum output of the network and then we divide the corresponding grid elements again into 625 smaller grid elements.
- With this fineness of the grid and the knowledge that the tansig function is continuous and bounded, we can use the maximum and minimum values thus found.
- Maximum and minimum values lie between -0.5 and +0.5 which means the network does not fire any pulse inside the box and hence the system is stable.
- The existence of P matrix proves stability of the system outside the box, hence the system is stable.
- It is important to note that this analysis does not consider limitations on stability due to limited control authority (position limits for flap & rudder).
Robustness analysis is important because operating conditions are different than ideal design conditions.

Robustness can be broadly classified as:

- Robustness to parametric uncertainties - This covers inaccuracies in plant and controller models.
- Robustness to non-parametric uncertainties - This covers high frequency unmodelled dynamics, Coulomb friction, stiction, measurement noise, computation round-off errors and sampling delays.

For this controller, we will only investigate robustness to parametric uncertainties:

- Due to errors in plant modeling.
- Due to errors in controller dynamics.
- Combination of above two cases.
ROBUSTNESS ANALYSIS

errors in plant modeling

- Important difficulties encountered during high angle-of-attack flight
  1. Sign reversal of yawing moment due to sideslip angle and change in magnitude
  2. Change in magnitude of rolling moment due to sideslip angle
  3. Lack of sideslip angle feedback

- Robustness test case - Plant model, for each of the controller includes:
  - Sign reversal of yawing moment due to sideslip and 15% increase in magnitude
  - 15% increase in magnitude of rolling moment due to sideslip angle
  - Lack of sideslip angle feedback
  - Figure 18, 19, 20 show the results for each controller
Figure 18, Worst case - plant modeling for MPVSC, cost = 19.827
Figure 19, Worst case - plant modeling for Fuzzy controller, cost = 16.36
Figure 20, Worst case - plant modeling for Neural controller, cost = 14.32
ROBUSTNESS ANALYSIS
Controller Modeling

- It is important to study the robustness of the controller with respect to the controller model itself

- WORST CASE - CONTROL POWER
  - Important parameters in the B matrix, i.e., B21, B22, B32 and B33 were changed by 15% in both directions one at a time
  - The change which leads to degraded performance was selected for robustness analysis
  - The worst case scenario for control power robustness is the combination of change in the above controller elements all at the same time
  - Hence, this case was selected as the worst case control power

- Figure 21, 22 and 23 show controller performance for MPVSC, Fuzzy and Neural controllers each, for this robustness case
Figure 21, Worst case controller modeling for MPVSC, cost = 20.135
Figure 22, Worst case controller modeling for Fuzzy controller, cost = 20.211
Figure 23, Worst case controller modeling for Neural controller, cost = 18.48
ROBUSTNESS ANALYSIS
Worst Case Scenario

- The most difficult robustness problem is encountered when both plant model and controller model are in error.
- A combination of case 1 and 2 was studied as the worst case scenario for X-29 controller.
- Figure 24 shows MPVSC performance for this case. Both differential flap and rudder are saturated for almost two cycles. 5 pulses are fired. Cost = 35.2
- Figure 25 shows Fuzzy controller performance for the same case. Both differential flap and rudder are saturated for one and half cycles. 4 pulses are fired. Cost = 29.05
- Figure 26 shows Neural controller performance. Both flap and rudder and saturated for only one cycle. 7 pulses are fired. Cost = 23.84
- This brings us to the conclusion of this work.
Figure 24, Robustness case 3 for MPVSC, cost = 35.2
Figure 25, robustness case 3 for Fuzzy controller, cost = 29.05
Figure 26, robustness case 3 for Neural controller, cost = 23.84