DOI: 10.13009/EUCASS2019-679

Aerodynamic Database Simulation Implementation Based On Neural Network and Neural Network Parameter Selection Using Genetic Algorithms

Hüseyin Burak KURT, Murat MILLIDERE** and Emrah SEZER****
** Flight Control Systems, Turkish Aerospace (TA), Ankara, Turkey*
huseyinburak.kurt@tai.com.tr
*** Middle East Technical University (METU), Ankara, Turkey*
muratmillidere@gmail.com
****Flight Control Systems, Turkish Aerospace (TA), Ankara, Turkey*
emrah.sezer@tai.com.tr

Abstract
Simulation implementation of aerodynamic database is performed using lookup tables in traditional approach. This approach is slow, when number of aerodynamic dimensions increases. In this study, we suggest that neural network model for simulation implementation of aerodynamic database, instead of lookup tables. Suggested approach is faster than lookup table, when number of aerodynamic dimension increases and estimation of neural network model is promising. However neural network model has many parameters which have to be tuned for optimum result. People who have solid experience with neural network can find optimum parameter in short time. However people don't have solid experience with neural network spend a lot of time to find optimum parameter values. In addition, in this study, in order to find optimum parameter values for neural network structure, parameters are optimized using genetic algorithm for given initial parameters. Also F-16 aerodynamic database is used for implementation of neural network model.

1. Introduction

Aerodynamic database implementation is one of the important part of aircraft simulation procedure. In traditional approach, lookup tables are used in implementation of aerodynamic database. Application of this approach make simulation procedure slow, when aerodynamic database dimension (AoA, AoS, Mach, etc.) increases. On the other hand, development of new machine learning techniques, estimation of function from dataset is possible and give very accurate and reliable results. Neural network is one of the machine learning techniques. Neural network is very popular and uses many areas of industries because of its accuracy. Artificial neural networks work as general function approximation, and are capable of approximating any continuous function to any desired accuracy by appropriate network architecture [1]. This ability of neural network is used to model aircraft aerodynamics. However neural network is black-box models, so it has no physical understanding for designer. Neural network is combination of weight matrix, input vector and bias vector [2].

The recent years, researchers are interested in artificial neural networks and deep neural network because of their successful results in large application areas. Their performance is improved with parallel to microprocessor industries development and available huge amount of data [3]. Some fields of used artificial neural network and deep neural networks; signal processing, pattern recognition, system identification, control, object detection, natural language processing, autonomous vehicle, unmanned air vehicle, image processing etc. Also, aircraft aerodynamic modelling, and estimation of aerodynamic coefficient and parameter (stability and control derivatives) [4-8].

Neural network don't have any physical understanding, it is black-box and complex model. In order to understand neural network model and use it, many parameter are used with neural network. So that, there are many parameter should be tuned by user. If user doesn't have solid experience with machine learning and neural network, it is hard process to obtain good model with neural network and takes long time. In order to solve this problem, some researcher

DOI: 10.13009/EUCASS2019-679

considers neural network as optimization problem and solve it using genetic algorithm, when generating aerodynamic database using neural network instead of CFD [9].

In this study, aerodynamic coefficients are estimated using artificial neural network and obtained model is used in aircraft simulation environment instead of lookup tables. After constructing the structure of neural network model, genetic algorithm is used to optimize neural network parameters to find optimum values.

2. Nomenclatures

AoA : Angle of Attack
AoS : Angle of Sideslip
CX : X-axis Aerodynamic Force Coefficient
CY : Y-axis Aerodynamic Force Coefficient
CZ : Z-axis Aerodynamic Force Coefficient
Cm : X-axis Aerodynamic Moment Coefficient
Cl : X-axis Aerodynamic Moment Coefficient
Cn : X-axis Aerodynamic Moment Coefficient
p : Aircraft roll rate about X-axis
q : Aircraft pitch rate about Y-axis
r : Aircraft yaw rate about Z-axis
V : Aircraft resultant velocity
$\bar{c}$: Wing mean aerodynamic cord
α : Angle of attacks
β : Angle of Sideslip
b : Wing span
η : Horizontal Tail effectiveness factor
δ_h : Horizontal Tail Deflection
δ_{lef} : Leading Edge-Flap Deflection
δ_a : Aileron Deflection
δ_r : Rudder Deflection
sb : Increment in variable produced by deflection of speed brake
ds : Deep stall
ANN : Artificial Neural Network
GA : Genetic Algorithm
CFD : Computational Fluid Dynamics
MSE : Mean Square Error

3. Aerodynamic Database

Aircraft simulation model is required for motion and performance analysis of an aircraft vehicle which is on ground or during flight in six degrees of freedom. Aircraft simulation model have multiple sub-models which includes flight environment (Earth, Gravity and Atmosphere), Flight Mechanics (Flight dynamics and Flight kinematics), Mass-Inertia-CM, Propulsion, Aerodynamics models.

In Aircraft simulation model, all sub-models works together. Some of these sub-models are formed as database like Aerodynamics database. Aerodynamic database of aircraft is combinations of theoretical and empirical methods; CFD analysis database, wind tunnel and flight test database. In simulation model, aerodynamic database is used to obtain aircraft's stability and control characteristics, flight performance and handling qualities. It contains aerodynamic force and moment coefficients and hinge moment of control surfaces. These component of the database are computed for various flight conditions and control surface deflections. It means aerodynamic database depends on many variables which includes Mach number, angle of attack, sideslip angle, angular rates and deflection of surfaces.

DOI: 10.13009/EUCASS2019-679

Generated aerodynamic database is used for training neural network model. Generated database should be specific format for neural network training. In this format, each breakpoint variable in each dimension should be combined each other. This combination increases the size of dataset and neural network training time. Although training time increases, estimation time is small. This provide us faster simulation models. Example of neural network dataset format is given in figure (1). As seen in figure (1), Mach number has breakpoints [0.3, 0.6], angle of attack has breakpoints [-15, 0, 15] and sideslip angle has breakpoints [-5, 0, 5]. Each combination of breakpoints as seen in figure (1) and corresponding aerodynamic force coefficients C_X, C_Y, C_Z and aerodynamic moment coefficients C_l, C_m, C_n are calculated.

Mach	*AoA*	*AoS*	C_X/C_m
0.3	-15	-5	-
0.3	-15	0	-
0.3	-15	5	-
0.3	0	-5	-
0.3	0	0	-
0.3	0	5	-
0.3	15	-5	-
0.3	15	0	-
0.3	15	5	-
0.6	-15	-5	-
0.6	-15	0	-
0.6	-15	5	-
0.6	0	-5	-
0.6	0	0	-
0.6	0	5	-
0.6	15	-5	-
0.6	15	0	-
0.6	15	5	-

Figure 1: Example of Neural Network dataset format

F-16 aerodynamic dataset based on low-speed wind tunnel tests of subscales model is used for this study [10]. Because F-16 dataset is open source and provides us to test our approach on real aircraft aerodynamic dataset which is generated from wind tunnel data.

Used aerodynamic data taken from NASA technical report in which aerodynamic data were derived from low-speed static and dynamic (force oscillation) wind-tunnel test conducted with subscale models of the F-16 in wind tunnel facilities at NASA Ames and Langley Research Center [10]. In this report, static aerodynamic coefficients are given in tabular form of AoA and AoS over range of $-20^o \leq \alpha \leq 90^o$ and $-30^o \leq \beta \leq 30^o$. In this study, AoA and AoS are taken in range of $-10^o \leq \alpha \leq 30^o$ and $-15^o \leq \beta \leq 15^o$ because of that F-16 aircraft generally fly in this range. Total coefficient equation are calculated as sum of various aerodynamic force and moment expressions as mentioned below equations;

For X-axis force coefficient:

$$C_X = C_X(\alpha, \beta, \delta_h) + \Delta C_{X,lef}\left(1 - \frac{\delta_{lef}}{25}\right) + \Delta C_{X,sb}(\alpha)\left(\frac{\delta_{sb}}{60}\right) + \frac{\bar{c}q}{2V}\left[C_{X_q}(\alpha) + \Delta C_{X_{q,lef}}(\alpha)\left(1 - \frac{\delta_{lef}}{25}\right)\right] \quad \textbf{(1)}$$

Where:

$$\Delta C_{X,lef} = C_{X,lef}(\alpha, \beta) - C_X(\alpha, \beta, \delta_h = 0^o), \quad C_{X_q}(\alpha) = \frac{\partial C_X}{\partial \frac{\bar{c}q}{2V}} \quad \textbf{(2)}$$

For Z-axis force coefficient:

$$C_Z = C_Z(\alpha, \beta, \delta_h) + \Delta C_{Z,lef}\left(1 - \frac{\delta_{lef}}{25}\right) + \Delta C_{Z,sb}(\alpha)\left(\frac{\delta_{sb}}{60}\right) + \frac{\bar{c}q}{2V}\left[C_{Z_q}(\alpha) + \Delta C_{Z_{q,lef}}(\alpha)\left(1 - \frac{\delta_{lef}}{25}\right)\right] \quad \textbf{(3)}$$

DOI: 10.13009/EUCASS2019-679

Where:

$$\Delta C_{Z,lef} = C_{Z,lef}(\alpha,\beta) - C_Z(\alpha,\beta,\delta_h = 0^o), \quad C_{Z_q}(\alpha) = \frac{\partial C_Z}{\partial \frac{\bar{c}q}{2V}} \tag{4}$$

For the pitching-moment coefficient:

$$C_m = C_m(\alpha,\beta,\delta_h)\eta_{\delta_h}(\delta_h) + C_{Z,t}\left(x_{cg,ref} - x_{cg}\right) + \Delta C_{m,lef}\left(1 - \frac{\delta_{lef}}{25}\right) + \Delta C_{m,sb}(\alpha)\left(\frac{\delta_{sb}}{60}\right) + \frac{\bar{c}q}{2V}\left[C_{m_q}(\alpha) + \Delta C_{m_{q,lef}}(\alpha)\left(1 - \frac{\delta_{lef}}{25}\right)\right] + \Delta C_m(\alpha) + \Delta C_{m,ds}(\alpha,\delta_h) \tag{5}$$

Where:

$$\Delta C_{m,lef} = C_{m,lef}(\alpha,\beta) - C_m(\alpha,\beta,\delta_h = 0^o), \quad C_{m_q}(\alpha) = \frac{\partial C_m}{\partial \frac{\bar{c}q}{2V}} \tag{6}$$

For Y-axis force coefficient:

$$C_Y = C_Y(\alpha,\beta) + \Delta C_{Y,lef}\left(1 - \frac{\delta_{lef}}{25}\right) + \left[\Delta C_{Y,\delta_{a=20^o}} + \Delta C_{Y,\delta_{a=20^o},lef}\left(1 - \frac{\delta_{lef}}{25}\right)\right]\frac{\delta_a}{20} + \Delta C_{Y,\delta_{r=30^o}}\left(\frac{\delta_r}{30}\right) + \frac{b}{2V}\left\{\left[C_{Y_r}(\alpha) + \Delta C_{Y_{r,lef}}(\alpha)\left(1 - \frac{\delta_{lef}}{25}\right)\right]r + \left[C_{Y_p}(\alpha) + \Delta C_{Y_{p,lef}}(\alpha)\left(1 - \frac{\delta_{lef}}{25}\right)\right]p\right\} \tag{7}$$

Where:

$$\Delta C_{Y,lef} = C_{Y,lef}(\alpha,\beta) - C_Y(\alpha,\beta), \qquad \Delta C_{Y,\delta_{a=20^o}} = \Delta C_{Y,\delta_{a=20^o}}(\alpha,\beta) - C_Y(\alpha,\beta),$$

$$\Delta C_{Y,\delta_{a=20^o},lef} = \Delta C_{Y,\delta_{a=20^o},lef}(\alpha,\beta) - C_{Y,lef}(\alpha,\beta) - \left[\Delta C_{Y,\delta_{a=20^o}}(\alpha,\beta) - C_Y(\alpha,\beta)\right],$$

$$\Delta C_{Y,\delta_{r=30^o}} = \Delta C_{Y,\delta_{r=30^o}}(\alpha,\beta) - C_Y(\alpha,\beta)$$

$$C_{Y_p}(\alpha) = \frac{\partial C_Y}{\partial \frac{pb}{2V}}, \qquad C_{Y_r}(\alpha) = \frac{\partial C_Y}{\partial \frac{rb}{2V}}, \tag{8}$$

For the yawing-moment coefficient:

$$C_n = C_n(\alpha,\beta,\delta_h) + \Delta C_{n,lef}\left(1 - \frac{\delta_{lef}}{25}\right) + C_{Y,t}\left(x_{cg,ref} - x_{cg}\right)\frac{\bar{c}}{b} + \left[\Delta C_{n,\delta_{a=20^o}} + \Delta C_{n,\delta_{a=20^o},lef}\left(1 - \frac{\delta_{lef}}{25}\right)\right]\frac{\delta_a}{20}$$
$$+\Delta C_{n,\delta_{r=30^o}}\left(\frac{\delta_r}{30}\right) + \frac{b}{2V}\left\{\left[C_{n_r}(\alpha) + \Delta C_{n_{r,lef}}(\alpha)\left(1 - \frac{\delta_{lef}}{25}\right)\right]r + \left[C_{n_p}(\alpha) + \Delta C_{n_{p,lef}}(\alpha)\left(1 - \frac{\delta_{lef}}{25}\right)\right]p\right\}$$
$$+\Delta C_{n_\beta}(\alpha)\beta \tag{9}$$

Where:

$$\Delta C_{n,lef} = C_{n,lef}(\alpha,\beta) - C_n(\alpha,\beta,\delta_h = 0^o), \qquad \Delta C_{n,\delta_{a=20^o}} = \Delta C_{n,\delta_{a=20^o}}(\alpha,\beta) - C_n(\alpha,\beta,\delta_h = 0^o),$$

$$\Delta C_{n,\delta_{a=20^o},lef} = \Delta C_{n,\delta_{a=20^o},lef}(\alpha,\beta) - C_{n,lef}(\alpha,\beta) - \left[\Delta C_{n,\delta_{a=20^o}}(\alpha,\beta) - C_n(\alpha,\beta,\delta_h = 0^o)\right],$$

$$\Delta C_{n,\delta_{r=30^o}} = \Delta C_{n,\delta_{r=30^o}}(\alpha,\beta) - C_n(\alpha,\beta,\delta_h = 0^o)$$

$$C_{n_p}(\alpha) = \frac{\partial C_n}{\partial \frac{pb}{2V}}, \qquad C_{n_r}(\alpha) = \frac{\partial C_n}{\partial \frac{rb}{2V}}, \qquad C_{n_\beta}(\alpha) = \frac{\partial C_n}{\partial \beta} \tag{10}$$

DOI: 10.13009/EUCASS2019-679

For the rolling-moment coefficient:

$$C_l = C_l(\alpha, \beta, \delta_h) + \Delta C_{l,lef}\left(1 - \frac{\delta_{lef}}{25}\right) + \left[\Delta C_{l,\delta_{a=20^o}} + \Delta C_{l,\delta_{a=20^o},lef}\left(1 - \frac{\delta_{lef}}{25}\right)\right](\frac{\delta_a}{20})$$

$$+\Delta C_{l,\delta_{r=30^o}}\left(\frac{\delta_r}{30}\right) + \frac{b}{2V}\left\{\left[C_{l_r}(\alpha) + \Delta C_{l_{r,lef}}(\alpha)\left(1 - \frac{\delta_{lef}}{25}\right)\right]r + \left[C_{l_p}(\alpha) + \Delta C_{l_{p,lef}}(\alpha)\left(1 - \frac{\delta_{lef}}{25}\right)\right]p\right\}$$

$$+\Delta C_{l_\beta}(\alpha)\beta \tag{11}$$

Where:

$$\Delta C_{l,lef} = C_{l,lef}(\alpha, \beta) - C_l(\alpha, \beta, \delta_h = 0^o), \quad \Delta C_{l,\delta_{a=20^o}} = \Delta C_{l,\delta_{a=20^o}}(\alpha, \beta) - C_l(\alpha, \beta, \delta_h = 0^o),$$

$$\Delta C_{l,\delta_{a=20^o},lef} = \Delta C_{l,\delta_{a=20^o},lef}(\alpha, \beta) - C_{l,lef}(\alpha, \beta) - \left[\Delta C_{l,\delta_{a=20^o}}(\alpha, \beta) - C_l(\alpha, \beta, \delta_h = 0^o)\right],$$

$$\Delta C_{l,\delta_{r=30^o}} = \Delta C_{l,\delta_{r=30^o}}(\alpha, \beta) - C_l(\alpha, \beta, \delta_h = 0^o)$$

$$C_{l_p}(\alpha) = \frac{\partial C_l}{\partial \frac{pb}{2V}}, \qquad C_{l_r}(\alpha) = \frac{\partial C_l}{\partial \frac{rb}{2V}}, \qquad C_{l_\beta}(\alpha) = \frac{\partial C_l}{\partial \beta} \tag{12}$$

Aerodynamic force and moment is calculated using equations (1-12) using wind tunnel AoA and AoS data. Leading-edge flap, rudder and aileron deflection limits are given by user and used in given above equations. Deflection of control surface used in this study is given table (1). After calculation of aerodynamic force and moment, dataset is converted desired neural network dataset format as seen figure (1).

Table 1: Deflection Limits of Control Surface

Control Surface	**Deflection Limit**
LEF, δ_{lef}	25
Aileron, δ_a	±10
Rudder, δ_r	±15
Horizontal Tail, δ_h	±20

4. Neural Network

Neural network is combination of basic processing elements, units or nodes, which are inspired form human brain functionality. Ability of neural network is that neural network can learn functions from set of training data [11]. In this study neural network is used to learn aerodynamic coefficient functions from set of aerodynamic dataset. In rest of this section, structure of used neural network and how to genetic algorithm is applied to this neural network structure are mentioned. In neural network training process, model input variables are aerodynamic dimensions and model output is one of the aerodynamic coefficient as seen figure (3). Instead of matching the multiple outputs through a single network, multiple modules are used, each consisting of a suitable neural network characterizing just one aerodynamic coefficient. Thus, the problem is broken down to sub models, each representing a multi-input, single-output subspace. This approach provides more flexibility, and it might turn out that smaller-size neural networks are adequate for each module. Usually, the training task is also simpler [12]. Also, inputs are different for each aerodynamic coefficient, input dimensions are decided using previous equations (1-12).

DOI: 10.13009/EUCASS2019-679

4.1 Neural network structure

Artificial neural network is combination of layers and each layer is combination of many artificial neuron. Artificial neuron simply has three set of rules; multiplication, summation and activation. In general mathematical formula of artificial neuron is given in equation (13).

$$y = F\left(\sum_{i=0}^{m} w_{ij} * x_i + b_i\right) \quad \textbf{(13)}$$

Where:

j - Artificial neuron number.
m - is input number for each artificial neuron.
x_{ij} - is input value where i goes from 0 to m, j goes from 1 to neuron number.
w_i - is weight value where i goes from 0 to m.
b - is bias value where i goes from 0 to m.
F - is activation function of artificial neuron.
y_i - is output value of artificial neuron.

In entrance of artificial neuron input vector is multiplied with weight vector, this means that each input has own weight and multiplied with them. This calculation can be called multiplication. In middle section of artificial neuron is sum function that sums all weighted inputs with bias variables. In exit of artificial neuron some specific activation functions are applied to previously sum weighted inputs and bias [13]. In figure (2) shows working principle of artificial neuron.

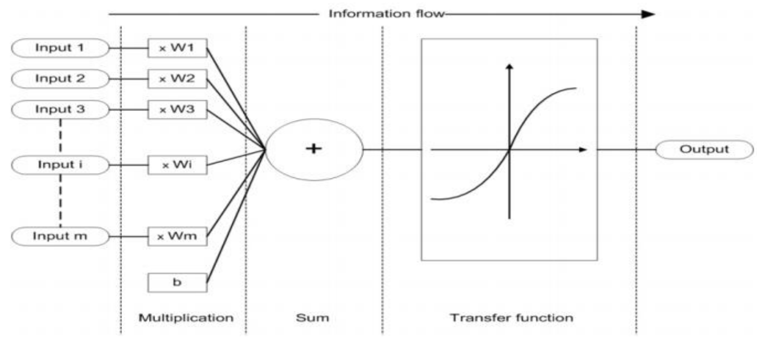


Figure 2: Working principle of artificial neuron.

Combination of these artificial neuron create layer. Combination of layers create artificial neural network. Artificial neural network has three kind of layers; input layer, hidden layer and output layer. Each neural network must has one input and output layer, but they can have many hidden layers. When number of hidden layer is greater than two or three, then we call the network as deep neural network. Simple neural network structure is given figure (3).

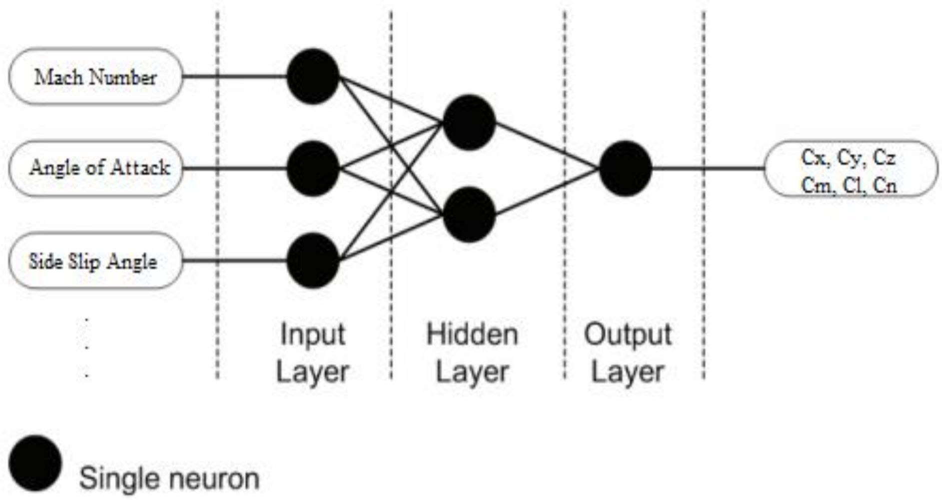


Figure 3: Simple neural network structure.

DOI: 10.13009/EUCASS2019-679

Activation function used in artificial neuron is key element of neural network. Without activation function neural network cannot learn nonlinear function. Activation function provide neural network some nonlinearities. There are many activation function used in literature, but commonly used functions are given below table (2).

Table 2: Commonly used Activation functions

Activation Function Names	**Activation Functions**
Sigmoid Activation Function	$F(x) = \frac{1}{1+e^{-x}}$
Linear Activation function	$F(x) = x$
Tanh activation function	$F(x) = \frac{2}{1+e^{-2x}} - 1$
ReLU activation function	$F(x) = \max(x, 0)$

Although there are many activation function in literature, there is no best activation function for estimation all function. So that user should try common activation functions and see how much the neural network model estimate real function. According to some experience in literature sigmoid and linear activation function is best solution for aerodynamic coefficient estimation with neural network [9]. For manual selection of activation function needs some experience or searching literature, in our solution, genetic algorithm is used to select activation function and other neural networks parameters automatically. In next section we mention how to implement genetic algorithm to optimize the neural network parameters.

Mean square error (MSE) method is used for loss calculation of neural network and MSE is given in equation (14)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y_i})^2 \qquad \textbf{(14)}$$

Where:
n - Number of data
Y_i - Real Data value
$\widehat{Y_i}$ - Estimated Data value

4.2 Neural network model validation

It is important concept to validate trained neural network, whether it is good enough or not for estimation of desired function. In neural network model validation process, we test our trained model with real aerodynamic coefficient data. In order to validate trained model, we should test this model with real aerodynamic dataset which is different from model trained aerodynamic dataset. Main idea is that model should not be trained with validation dataset. So that all aerodynamic dataset is shuffled for providing randomness to dataset. If dataset is random, generated model can learn the function better. After shuffling procedure, dataset is divided three part; training (70%), validating (%15) and testing (15%) part with given percentages. Given percentages are user defined, they can be changed but in machine learning literature recommended division percentages like this. Training dataset is used to train model. Validation dataset is used to validate model performance in same time when model is training. If model validation loss goes to bad situation training procedure is terminated. We use validation dataset to check model training procedure is going to get better or not. Test dataset is used to decide trained model is good enough or not. Trained dataset don't see any part of test dataset, when training procedure.

4.3 Neural network parameter optimization with genetic algorithm

In previous section, general neural network structure is mentioned. In this section, user defined neural network parameters and how genetic algorithm is integrated to neural network parameter optimization will be mentioned. Neural network structure is complicated, especially when the neuron and hidden layer number increase. There are

DOI: 10.13009/EUCASS2019-679

many parameter which should be tuned by user for finding optimum result. If user does not have solid experience in this area, it is hard process to find optimum solution and takes long time to find. Using genetic algorithm optimization method is suggested in order to find optimum parameters for desired cost function.

In order to use genetic algorithm, firstly we decide which parameters should be optimized, then decide how to construct the cost function.

There are many parameter which can be optimized in neural network, in this study we choose following parameters;

- Number of neurons in hidden layers
- Activation functions (Sigmoid, Linear, Logarithmic Sigmoid)

Number of hidden layer is very effective for learning functions. However if number of layer is too much, it result in overfitting and other side, if number of layer is too low, neural network cannot learn desired function. In literature, two hidden layer for estimating aerodynamic coefficient is used and recommended [9]. So in this study neural network with two hidden layers is used. Number of neurons in each hidden layer is just like behave number of layers. We used three activation function (Sigmoid, linear and Logarithmic Sigmoid). Each activation function perform different performance for different problem.
In literature, some people initialize the weight matrix with same value in every iteration of optimization procedure. For example, Equation (14) is neural network weight function between input and first hidden layer. There are two input in input layer and two hidden neuron in hidden layer. So we obtain 2x2 weight matrix. Each element of 2x2 weight matrix is 0.5 as seen equation (14);

$$W = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix} \tag{15}$$

In this case, randomness of weight matrix of neural network is disappeared, randomness of network weights is important concept, we don't want to lose randomness of weight matrix, when initialize it. So that we initialize weight matrix randomly. But, this time each weight matrix is random and different with each other, when initialize each iteration of optimization procedure. This is unacceptable for view of optimization. As a result, we decide initialize neural network weight with same random matrix in each optimization iteration. Example of fixed random weight which initialize in each optimization iteration is given equation (15).

$$W = \begin{bmatrix} 0.2 & -0.4 \\ -0.5 & 0.8 \end{bmatrix} \tag{16}$$

Each element of matrix is initialize in range of [-1, 1] randomly and initialized weight matrix stay fixed at each iteration of optimization.

Another important parameter of neural network is used backpropagation algorithm. There many backpropagation algorithm. In this study, Levenberg-Marquardt backpropagation algorithm is used. Levenberg-Marquardt backpropagation is chosen because it is fast and give good results, when compared other backpropagation methods. It is combination of Gradient Descent and Gauss-Newton optimization methods and more robust than both of them.

In this study, neural network should be deterministic except parameters are optimized by genetic algorithm, because of view of optimization. So that Levenberg-Marquardt backpropagation algorithm should also be deterministic. This means that when we give neural network with same parameters with Levenberg-Marquardt backpropagation algorithm, we should obtain same weight matrix in result of same epoch number of neural network training. We tried this concept, neural network trained many times with same parameters with Levenberg-Marquardt backpropagation algorithm and training procedure is terminated after same epoch number, we obtained same neural network weights. This means that everything in neural network is deterministic. This is necessary condition for optimization procedure and for this problem this condition is provided to neural network training procedure.

In beginning of genetic optimization procedure, parameters which will be tuned and loss (cost) function should be decided. Previous of this section, we mentioned which parameter stay free and which parameter stay fixed. After decision of parameters, genetic algorithm needs a cost function to be minimized. Mean square error (MSE) between estimated value and real value of aerodynamic coefficient is decided as cost function. Equation (14) gives MSE. Loss (cost) of genetic algorithm is obtained with calculation of MSE using %80 percent of train data and %20 of test data in equation (17). Genetic algorithm loss is calculated using both train and test dataset, while neural network loss is calculated just using train dataset. Because, we need to validate neural network performance using dataset which is different from train dataset. In the case of genetic algorithm, loss of model using test data is important for genetic algorithm performance.

DOI: 10.13009/EUCASS2019-679

$$MSE\_Loss\_GA = 0.8\left[\frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2\right]_{train} + 0.2\left[\frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2\right]_{test} \quad \textbf{(17)}$$

Where:

n - Number of data in train/test data

Y_i - Real Data value in train/test data

$\widehat{Y}_i$ - Estimated Data value in train/test data

Problems where the variables can only take on integer values are called discrete optimization problems or integer programming problems. In a few applications it is sufficient to solve the problem ignoring the integrality restriction, and once a solution is obtained, to round off the variables to their nearest integer. Unfortunately rounding off of a solution does not guarantee that it is optimal, or even that it is feasible, so this approach is often inadequate. In case of our problem is discrete optimization. Firstly optimization problem try to optimize using continuous genetic algorithm, but performance of optimization algorithm is not good enough, because it is difficult to expression of discrete parameter as continuous parameter as mentioned previous reason. After that we implement discrete genetic optimization algorithm and get better performance.

In order to check whether genetic algorithm always gives us best combination of parameters, brute force test is applied. Firstly all parameters combination and their cost function is found with fix epoch number using brute search algorithm. As a result of this search, 900 result is obtained, 10 nodes for first and second layer number and 3 activation function of each layer the all results are sorted with respect to its cost values. After brute search of parameter combination, some search is done using genetic algorithm and results of genetic algorithm search gives us mostly first ten sorted result obtained with brute force search. Brute force test result is given table (3). In table, sorted first 10 result of brute force is given. Then genetic search algorithm is run ten times, %90 of genetic algorithm estimation is inside best result of brute force test. In table (3), distribution of genetic algorithm result is given. So genetic algorithm can estimate best result of given optimization problem.

Table 3: Brute Force Test

	Node Num. Layer 1	**Node Num. Layer 2**	**Act. Func. Layer 1**	**Act. Func. Layer 2**	**Cost**	**GA Result**
1	7	8	Log-sigmoid	Sigmoid	2.0235e-11	4
2	7	7	Log-sigmoid	Sigmoid	2.1788e-11	2
3	7	4	Log-sigmoid	Sigmoid	2.3015e-11	1
4	6	5	Log-sigmoid	Sigmoid	2.311e-11	1
5	9	4	Log-sigmoid	Sigmoid	2.3702e-11	0
6	7	6	Log-sigmoid	Sigmoid	2.3768e-11	0
7	8	4	Log-sigmoid	Sigmoid	2.3819e-11	0
8	8	5	Log-sigmoid	Sigmoid	2.4043e-11	0
9	7	9	Log-sigmoid	Sigmoid	2.4348e-11	0
10	6	9	Log-sigmoid	Sigmoid	2.462e-11	1

DOI: 10.13009/EUCASS2019-679

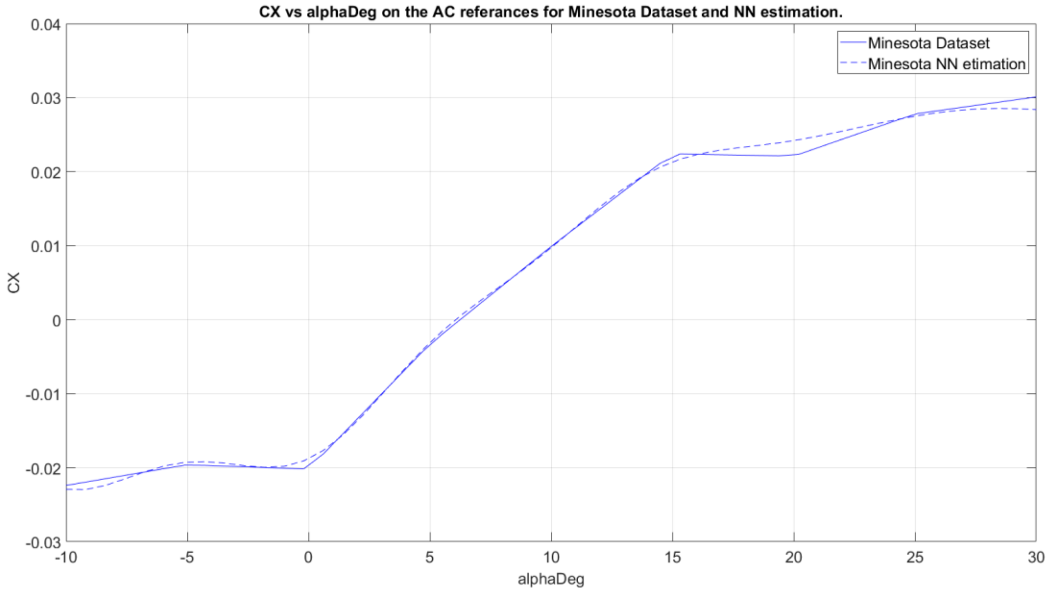
5. Results and Conclusion

In this section, aerodynamic estimation of trained neural network model with genetic algorithm is compared with real F-16 dataset for each aerodynamic coefficient.

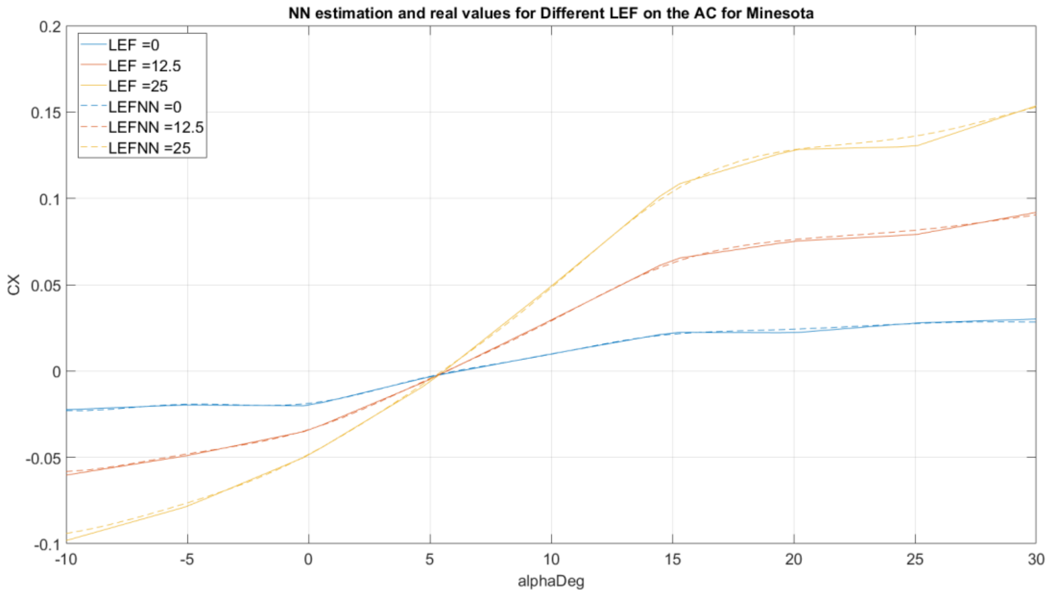
5.1 Aerodynamic coefficient C_X results

Genetic algorithm results of aerodynamic coefficient C_X is node number of layer 1 = 18, node number of layer 2 = 5, activation function of layer 1 = sigmoid and activation function of layer 2 = sigmoid.

C_X vs AoA plot of Minnesota dataset and neural network estimation;

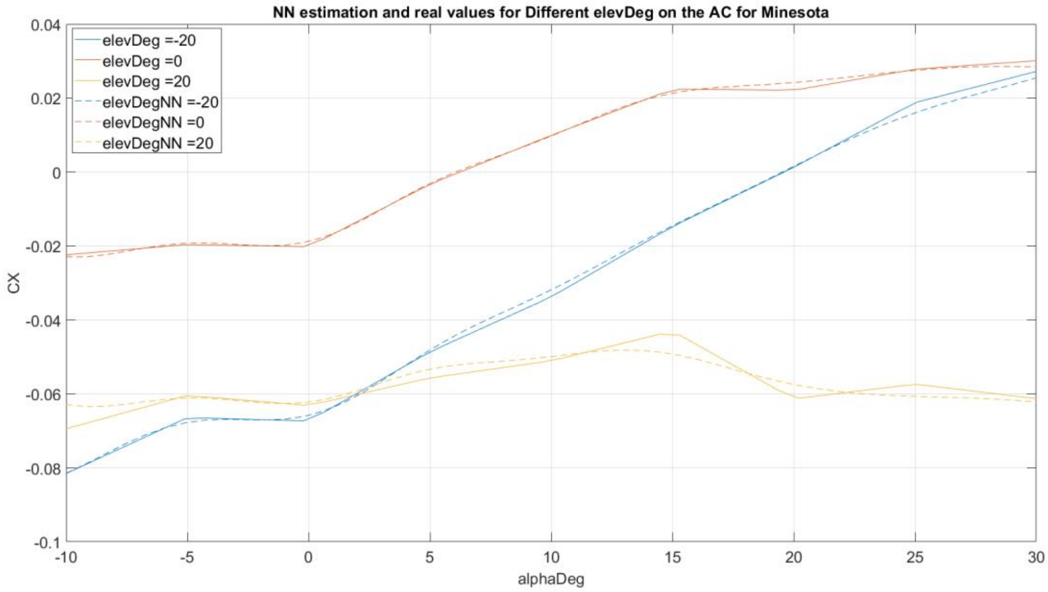


C_X vs AoA plot of Minnesota dataset and neural network estimation for various LEF values;

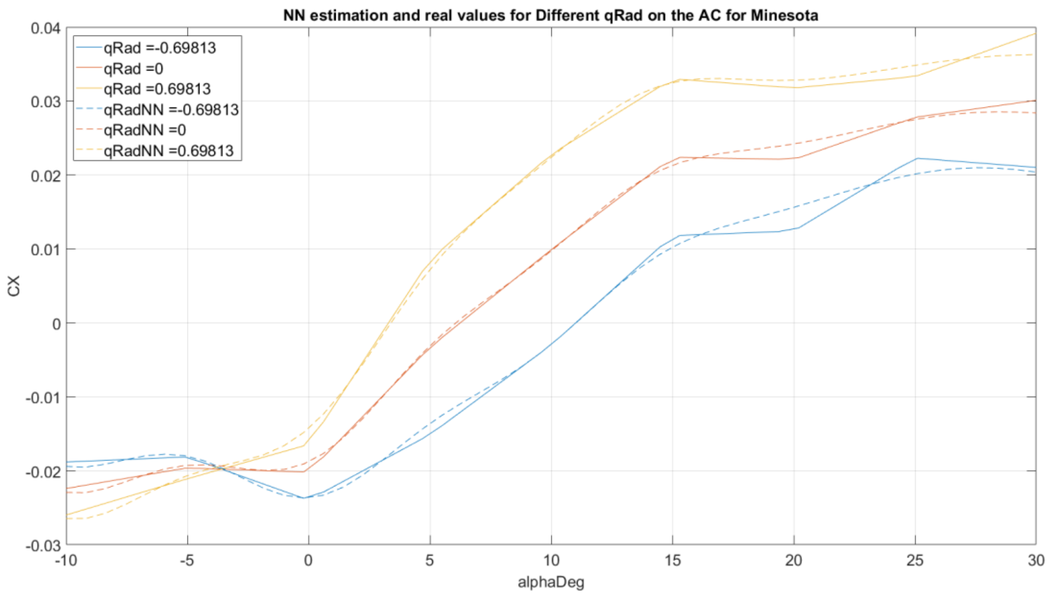


DOI: 10.13009/EUCASS2019-679

C_X vs AoA plot of Minnesota dataset and neural network estimation for various elevator values;

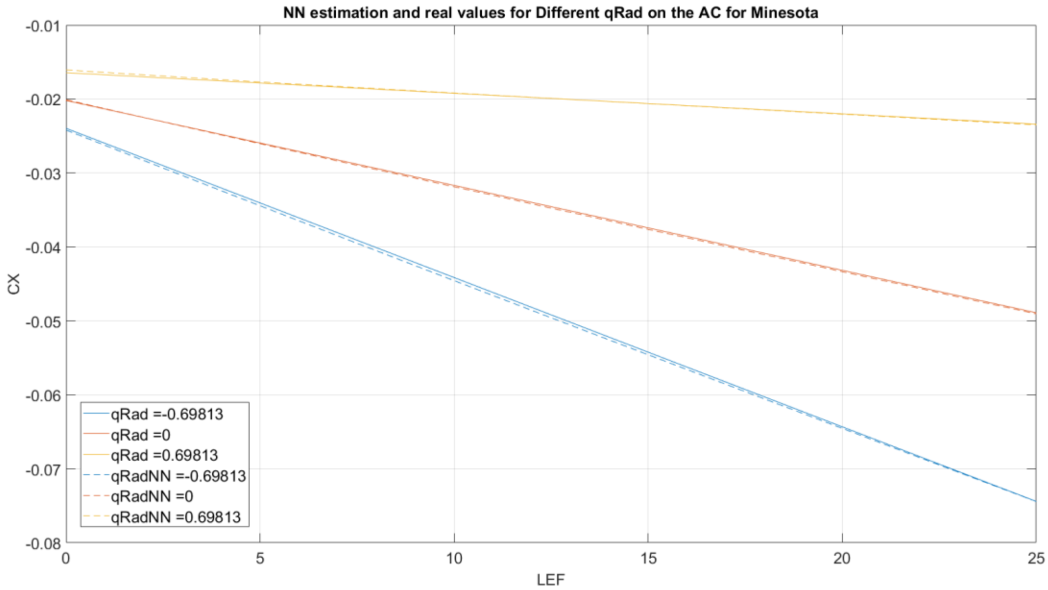


C_X vs AoA plot of Minnesota dataset and neural network estimation for various pitch rate(rad) values;



DOI: 10.13009/EUCASS2019-679

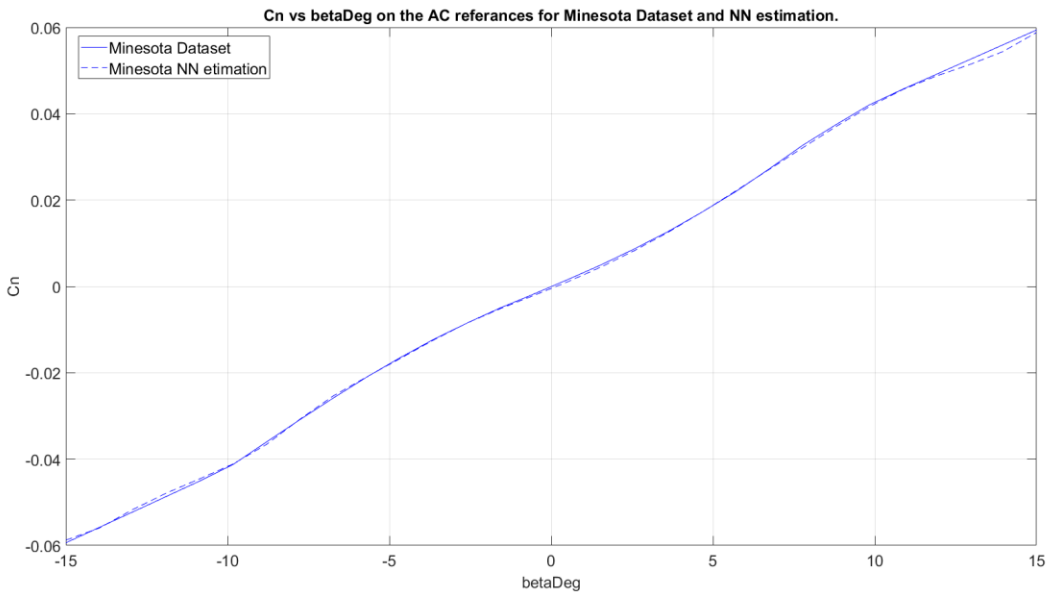
C_X vs LEF plot of Minnesota dataset and neural network estimation for various pitch rate(rad) values;



5.2 Aerodynamic coefficient C_n results

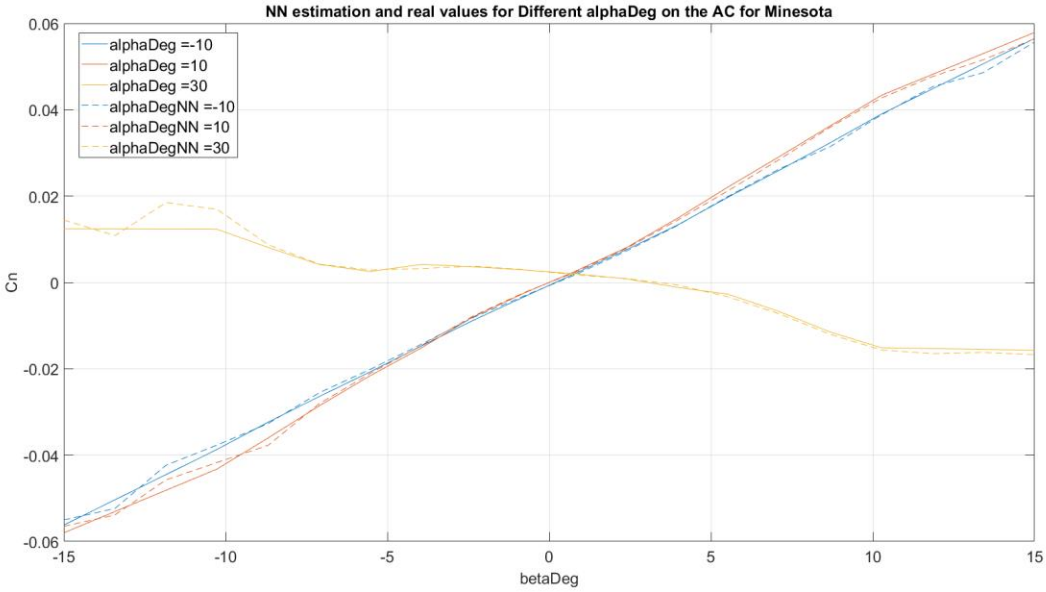
Genetic algorithm results of aerodynamic coefficient C_X is node number of layer 1 = 15, node number of layer 2 = 16, activation function of layer 1 = Logarithmic sigmoid and activation function of layer 2 = sigmoid function.

C_n vs AoS plot of Minnesota dataset and neural network estimation;

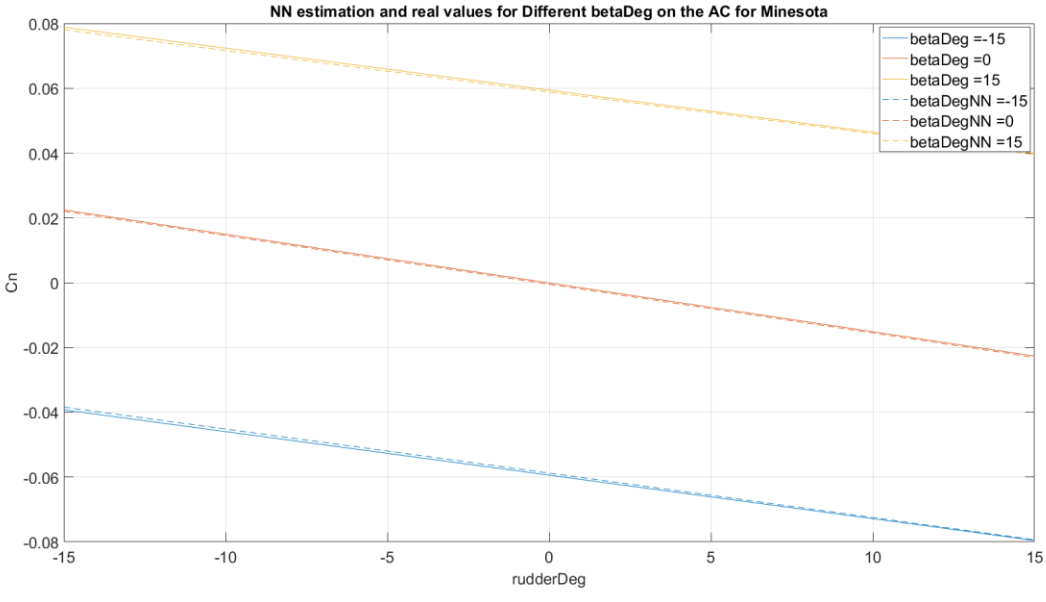


DOI: 10.13009/EUCASS2019-679

C_n vs AoS plot of Minnesota dataset and neural network estimation for various AoA values;

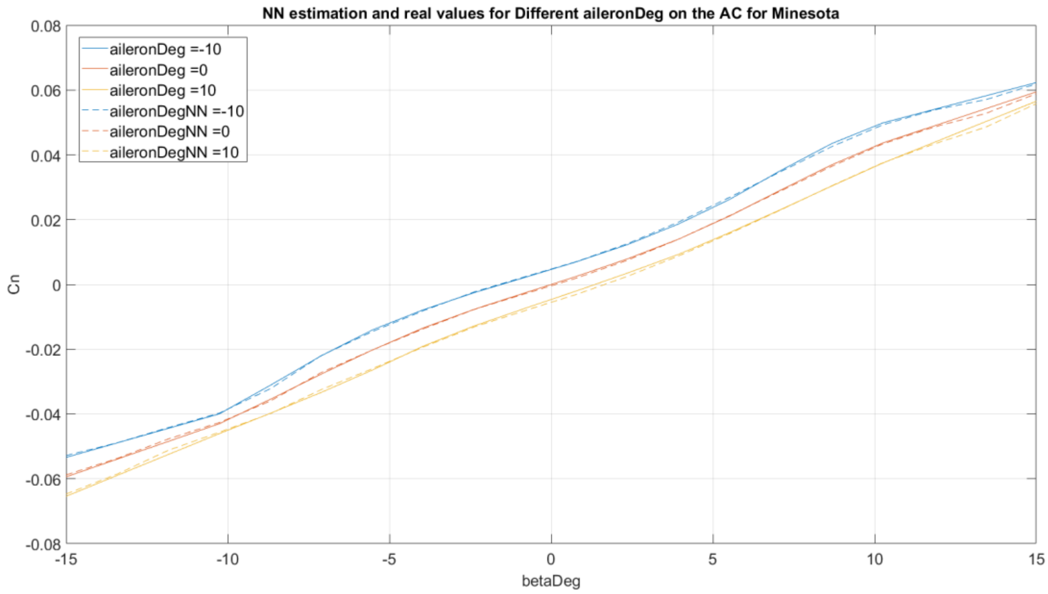


C_n vs rudder plot of Minnesota dataset and neural network estimation for various AoA values;

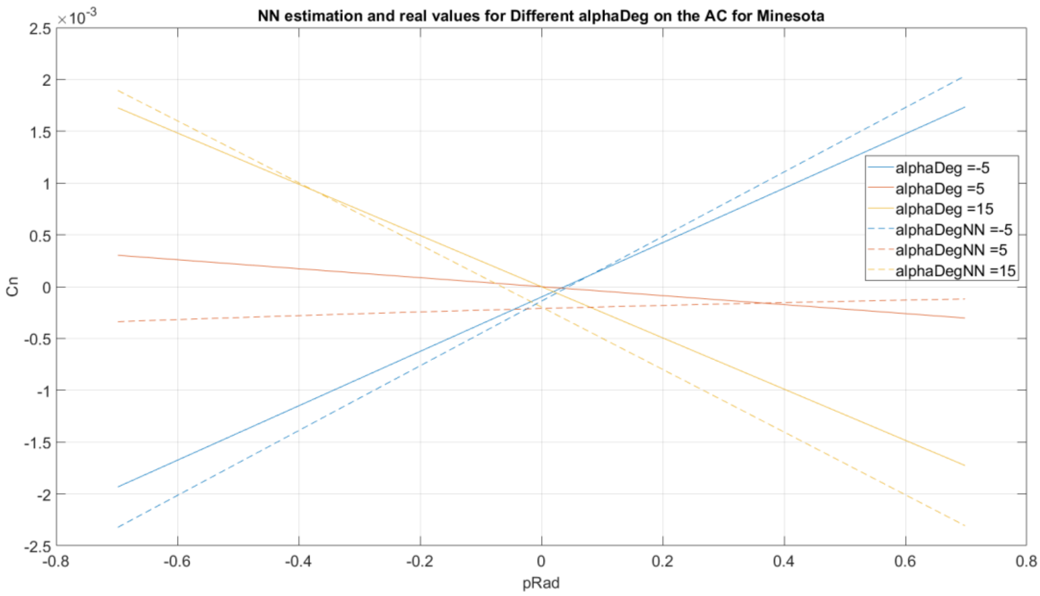


DOI: 10.13009/EUCASS2019-679

C_n vs AoS plot of Minnesota dataset and neural network estimation for various aileron values;

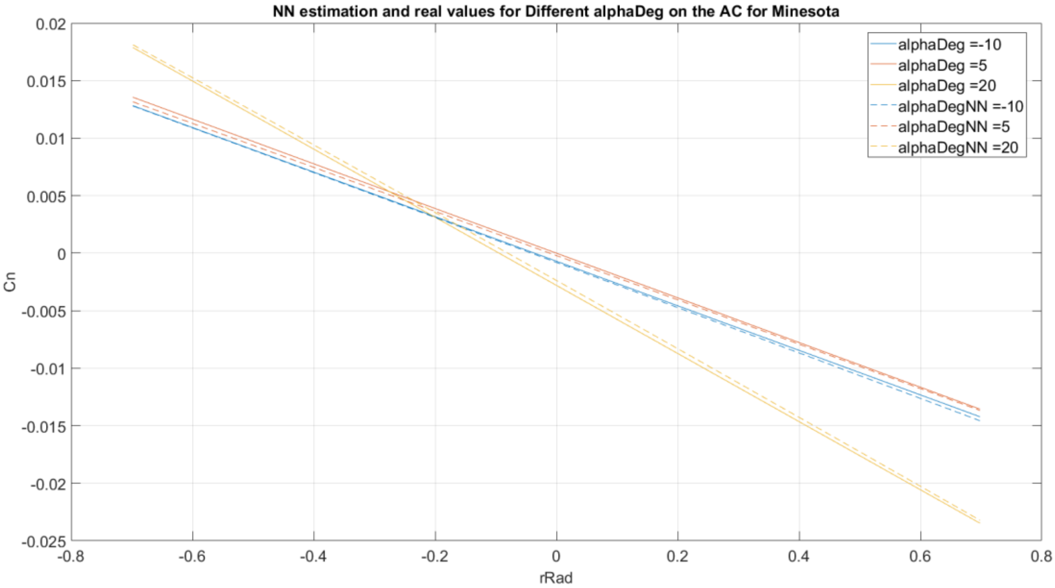


C_n vs roll rate plot of Minnesota dataset and neural network estimation for various AoA values;



DOI: 10.13009/EUCASS2019-679

C_n vs yaw rate plot of Minnesota dataset and neural network estimation for various AoA values;



References

[1] Horni, K. K., Stinchcombe, M. and White, H. “Multi-layer feed forward networks are universal approximators”, *Neural Networks*, 1989, 2, (5), pp 359-366.

[2] Basappa and Jategaonkar, R.V. “Aspects of feed forward neural network modelling and its application to lateral-directional flight data”, DLR-IB III-95/30, September 1995.

[3] Ian Goodfellow, Yoshua Begio and Aeron Courville, “Deep learning”, MIT Press, 2016.

[4] Hess, R.A. “On the use of back propagation with feed-forward neural networks for aerodynamic estimation problem”, AIAA paper 93-3638, August 1993.

[5] Linse, D.J. and Stengel, R.F. “Identification of aerodynamic coefficient using computational neural networks”, *J Guid COnt Dynam*, 1993, 16, (6), pp1018-1025.

[6] Youssef, H.M. and Juang, J.C. “Estimation of aerodynamic coefficient using neural networks”, AIAA paper 93-3639, August 1993.

[7] Raol, J.R. and Jategaonkar, R.V. “Aircraft parameter estimation using recurrent neural networks – a critical appraisal”, AIAA paper 95-3504-C, August 1995.

[8] Raisinghani, S.C., Ghosh, A.K., Kalra, P.K. “Two new techniques for aircraft parameter estimation using neural networks”, *The Aeronautical Journal*, paper no 2349, Jan. 1998.

[9] Fazil Selcuk Gomec and Murat Canibek, “Aerodynamic Database Improvement of Aircraft based on Neural Networks and Genetic Algorithms”, *7th European Conferences for Aeronautics and Space Sciences (EUCASS)*, 2017, Milan, Italy.

[10] Luat T. Nguyen, at al., “Simulator Study of Stall/Post-Stall Characteristics of a Fighter Airplane With Relaxed Longitudinal Static Stability”, National Aeronautics and Space Administration, Hampton, Virginia, NASA Technical Paper 1538, 1979.

[11] Kevin Gurney. 1997. “An Introduction to Neural Networks”. Taylor and Francis, Inc., Bristol, PA, USA.

[12] Jategaonkar, R.V. (2015). Flight Vehicle System Identification: A Time-Domain Methodology, Second Edition.

[13] Andrej Krenker, Janez Bester and Andrej Kos. “Introduction to the Artificial Neural Networks – Methodological Advances and Biomedical Applications”, Kenji Suzuki, IntechOpen.