Defence Science Journal, Vol. 68, No. 3, May 2018, pp. 241-250, DOI : 10.14429/dsj.68.11843

Aerodynamic Parameters Estimation Using Radial Basis Function Neural Partial Differentiation Method

Jitu Sanwale* and Dhan Jeet Singh

Aircraft Upgrade Research & Design Centre, Hindustan Aeronautics Limited, Nasik - 422 207, India
**E-mail: sanwalejitu@gmail.com*

ABSTRACT

Aerodynamic parameter estimation involves modelling of force and moment coefficients and computation of stability and control derivatives from recorded flight data. This problem is extensively studied in the past using classical approaches such as output error, filter error and equation error methods. An alternative approach to these model based methods is the machine learning such as artificial neural network. In this paper, radial basis function neural network (RBF NN) is used to model the lateral-directional force and moment coefficients. The RBF NN is trained using k-means clustering algorithm for finding the centers of radial basis function and extended Kalman filter for obtaining the weights in the output layer. Then, a new method is proposed to obtain the stability and control derivatives. The first order partial differentiation is performed analytically on the radial basis function neural network approximated output. The stability and control derivatives are computed at each training data point, thus reducing the post training time and computational efforts compared to hitherto delta method and its variants. The efficacy of the identified model and proposed neural derivative method is demonstrated using real time flight data of ATTAS aircraft. The results from the proposed approach compare well with those from the other.

Keywords: RBF Neural Network; EKF; k-means clustering; PDM; Aerodynamic parameter estimation

1. INTRODUCTION

The mathematical modelling of aerodynamics is an inevitable activity in aerospace realm. Such models find applications in multi-disciplinary activities which include performance optimisation, high-fidelity simulators for pilot training, design enhancements and additions, synthesis of flight control laws, flight envelop expansion, and upgrades of autopilot systems, etc. The mathematical model typically characterises nondimensional force and moment coefficients. Numerical values for these parameters appearing in the mathematical model are often derived from ground test methods such as wind-tunnel test and computer aided tools. However, the wind-tunnel test conditions and assumptions in software implementation typically do not replicate the actual flight environment. Consequently, it is desirable to derive estimates for these coefficients directly from flight test data.

Aerodynamic parameter estimation using classical methods both in time domain and frequency domain are well documented[1-4]. The most widely used system identification methods have been the output error method (OEM), filter error method (FEM) and equation error method (EEM). These methods require postulation of a mathematical model of the aircraft and also noise dynamics. Although highly nonlinear models can be estimated applying these methods, it may involve significant, time consuming, efforts to arrive at appropriate model structure based on the physics of the problem in such cases. Such phenomenological models often turn out to be quite complex.

Machine learning methods such as artificial neural network (ANN) provide another approach to investigate the same phenomenon, which is rather quick and easy. The artificial neural network is a computational model based structure method and functions as a biological neural network. Potential uses of ANNs in aerodynamic system identification are reported[5-20] due to its universal function approximation capability[21]. The multi layer perceptron (MLP) feed forward neural network (FFNN) is used[5-18] as a base model for nonlinear mapping between input and output spaces. The stability and control derivatives are then computed using delta method[9]. The delta method is based on finite difference approximation and successfully applied[9-13,19-20] for modelling and estimation of aerodynamic parameters. The MLP FFNN consists of input layer, hidden layers and output layer. The neurons in input and output layers are determined by input and output dimensions. The judicious choice of number of neurons in hidden layers and number of hidden layers is essential for optimal modelling. There is no analytical solution to this problem. It is an iterative procedure and poses a difficulty in finding the optimum network design parameters. There are also some drawbacks of FFNN such as slow convergence, computational memory, sensitiveness to outliers, etc. These difficulties can be avoided by using alternate neural network architecture based on radial basis functions. Broomhead & Lowe[22] proposed the radial basis function neural network (RBF NN). The RBF NN has

Received : 03 September 2017, Revised : 11 January 2018
Accepted : 23 January 2018, Online published : 16 April 2018

equivalent capabilities as those of the MLP FFNN model. The RBF NN has become a good alternative to FFNN due to its fast learning feature. The RBF NN also has universal approximation and regularisation capabilities[23]. The application of RBF NN in rotorcraft modelling is reported in[19-20].

Alternate to delta method proposed by Kuttieri[14], *et. al* is neural partial differentiation method (PDM). This method is based on the partial differentiation of FFNN output in terms of network parameters. The approach adopted[14] and applied in various aerodynamic problems[15-17] is extended and a new neural partial differentiation method using RBF NN is proposed to estimate the stability and control derivatives. In this paper, the application of RBF NN is presented to model the nondimensional force and moment coefficients. The RBF NN is trained using hybrid learning proposed by Moody and Darken[24]. In the present application, k-means clustering is used at first stage to find the centers of radial basis functions and extended Kalman filter (EKF) in the second stage to obtain the weights in the output layer.

2. RADIAL BASIS FUNCTION NEURAL NETWORK

The Radial Basis Function neural network (RBF NN) is a layered structure as shown in Fig. 1. The RBF NN is represented by input layer, hidden layer and output layer[22]. The input layer constitute by the input vector $\boldsymbol{x}$, where $\boldsymbol{x} \in R^m$ and m is number of inputs. The hidden layer consists of K number of computational units (neurons) and each unit is described by a radial basis function ψ. The RBF NN differs from the feed forward neural network architecture in respect of number of hidden layers. The RBF NN is constructed by single hidden layer unlike multiple hidden layers in FFNN. The parameter K is a network design parameter, to be selected to yield the best performance in terms of specified cost function.

Many types of radial basis functions and their desired properties are given[25]. Popularly used radial functions are thin plate spline by Duchon[26], multiquadrics and inverse multiquadrics by Hardy[27], Gaussian function by Schagen[28], In this paper, a Gaussian function $\psi(r) = \exp(-r^2)$ is used as a radial basis function for implementation of RBF neural network. The hidden layer outputs are computed as $\psi_i = \psi(d_i)$;

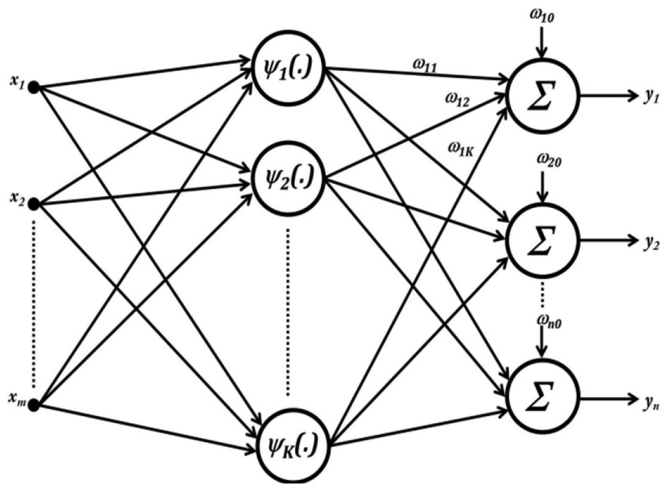

Figure 1. Radial basis function neural network architecture.

$i = 1, 2, \ldots\ldots, \mathrm{K}$, where $d_i = \left\| \boldsymbol{x} - \boldsymbol{c}^i \right\|^2$. The vector $\boldsymbol{c} \in R^m$ is called center of the radial basis function.

The function approximation model describing the mapping of input space to output space, $R^m \rightarrow R^n$ is given by

$$y_k = w_{k0} + \sum_{i=1}^{\mathrm{K}} w_{ki} \psi(d_i)\ ;\ k = 1, 2, \ldots\ldots\ldots, n \tag{1}$$

The complete model response for multi input, multi output (MIMO) system is given in matrix form as

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} w_{10} & w_{11} & w_{12} & \cdots & w_{1\mathrm{K}} \\ w_{20} & w_{21} & w_{22} & \cdots & w_{2\mathrm{K}} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ w_{n0} & w_{n1} & w_{n2} & \cdots & w_{n\mathrm{K}} \end{bmatrix} \begin{bmatrix} 1 \\ \psi_1 \\ \vdots \\ \psi_{\mathrm{K}} \end{bmatrix} \tag{2}$$

The closed form system model is

$$y = W\psi \tag{3}$$

where, output vector $[y]_{n\times 1} = \{y_k\}_{k=1,2,\ldots\ldots,n}$, weight matrix $[W]_{n\times(\mathrm{K}+1)} = \{w_{ki}\}_{k=1,2,\ldots\ldots\ldots,n}^{i=0,1,2,\ldots\ldots\ldots,\mathrm{K}}$, and $[\psi]_{(\mathrm{K}+1)\times 1} = \begin{bmatrix} 1 & \{\psi_i\}_{i=1,2,\ldots\ldots,\mathrm{K}} \end{bmatrix}^T$.

The input and output layers in RBF NN operate in different ways and, therefore, corresponding parameters (RBF centers and weights) have different meaning and properties. Hence, it is appropriate to use different learning algorithms for them. Accordingly, function approximation using RBF NN becomes a two stage optimisation problem. The first stage optimisation is performed at input layer for selection of radial basis function centers, $\boldsymbol{c}$. The second stage optimisation is used to find weight matrix, W.

Hybrid learning process used[24] consists of two different stages:

(i) Self-organised learning stage which estimates appropriate locations for the centers of the RBF in the hidden layer, and

(ii) Supervised learning stage for estimating the weights of the output layer.

Self-organised learning process is based on the unsupervised learning and need a clustering algorithm. Clustering is a technique in which input data points are partitioned into clusters in such a way that the measure of similarity between any pair of data points to each cluster minimises a specified cost function. There are many clustering techniques available in literature such as SoC, fuzzy C-means, EM clustering, GG clustering, IMC and GK clustering[29-34], etc. In this paper, k-means clustering algorithm is used due to its simple implementation and yet effective performance. Once, the radial basis function and its centers are fixed, the network is linear in the weight parameters. Various derivative based methods have been used to train the RBF NN including gradient descent[35], Kalman filter[36], and the back propagation[37]. Derivative free methods such as learning automata[38], simulated annealing[39] and genetic algorithms[40], have also been used to train the neural networks. Derivative free methods are advantageous because they do not require derivatives of the cost function with respect to network parameters and also provides global minimum solution. The drawback of these methods is the slow rate of convergence. The derivative based

methods are advantageous in fast convergence, but they suffer by convergence to local minima. In this paper, we formulate RBF NN training method based on extended Kalman filter (EKF). Standard EKF is reformulated for multi input- multi output system to estimate the weights in hidden layer.

2.1 k-means Clustering

The k-means clustering, proposed[32] and applied in this paper, is a representative of one of the unsupervised learning used to compute the centers of radial basis functions.

Consider $\{\boldsymbol{x}_j\}_{j=1}^{N}$ denote a set of multidimensional measurements that is to be partitioned into a set of K clusters. The objective of k-means clustering is to find cluster centers $\boldsymbol{c}$ such as to minimise total intra-cluster variance as the cost function:

$$J = \sum_{i=1}^{\mathrm{K}} \sum_{j=1}^{N} \left\| x_j - c_i \right\|^2 \tag{4}$$

Defining the binary membership matrix M as

$M = \{m_{ij}\}$ for $i = 1, 2, ..., \mathrm{K}$ and $j = 1, 2, ..., N$

where $m_{ij} = 1$ if $\left\| x_j - c_i \right\|^2 \le \left\| x_j - c_k \right\|^2$ for each $k \neq i$
$= 0$ otherwise

Since the measurement vector x_j fall in any one of the K clusters, the binary membership matrix M possesses the following properties

$$\sum_{i=1}^{\mathrm{K}} m_{ij} = 1, \forall j = 1, 2, ..., N \qquad \sum_{i=1}^{\mathrm{K}} \sum_{j=1}^{N} m_{ij} = n$$

Now, if m_{ij} is fixed, the cluster centres that minimised Eqn. (4) are calculated as

$$c_i = \frac{\sum_{k} x_k}{\sum_{j=1}^{N} m_{ij}} \tag{5}$$

Here k refers the number of measurement vectors (input points) in i[th] cluster.

Algorithm:

1. Select K points at random from the measurement data as cluster centers
2. Form K clusters by assigning each point to its closest center according to the Euclidean distance function
3. Compute the centroid or mean of each cluster using Eqn. (5)

Loop on steps 2 and 3 until Centroids do not change

There is no analytical method to find the optimal value of K . A rule of thumb is to compare the outcomes of multiple iterations with different K and choose the best one.

2.2 Extended Kalman Filter

The extended Kalman filter (EKF) algorithm is used to estimate state vector from measurements. In this paper, the state vector is considered to be the weights of RBF neural network. Derivations of the EKF are available in literature[41-42]. In this section a brief and how it is applied to RBF neural network to obtain the weights of the output layer are presented.

Consider the system model given as

State equation:

$\theta_{k+1} = f(\theta_k) + \vartheta_k$, where $\vartheta_k \to \mathbb{N}(0, Q_k)$ (6)

Output equation:

$y_k = h_k(\theta_k) + \nu_k$, where $\nu_k \to \mathbb{N}(0, R_k)$ (7)

where ϑ_k and ν_k are noise parameters represented by normal distribution with zero mean and Q and R as covariance matrices.

The computational steps of EKF are as follows:

For $k = 1, 2, ...N$, perform the following

1. Compute the system (state derivate) matrix as

$$A_{k-1} = \left. \frac{\partial f_{k-1}}{\partial \theta} \right|_{\hat{\theta}^{+}_{k-1}} \tag{8}$$

2. Perform the state estimate and estimation error covariance as follows

$$P_k^{-} = A_{k-1} P_{k-1}^{+} A_{k-1}^{T} + Q_{k-1} \tag{9}$$

$$\hat{\theta}_k^{-} = f_{k-1}(\hat{\theta}_{k-1}^{+}) \tag{10}$$

3. Compute the output matrix as

$$C_k = \left. \frac{\partial h_k}{\partial \theta} \right|_{\hat{\theta}_k^{-}} \tag{11}$$

4. Execute the following Kalman filter equations

$$K_k = P_k^{-} C_k^{T} (C_k P_k^{-} C_k^{T} + R_k)^{-1} \tag{12}$$

$$\hat{\theta}_k^{+} = \hat{\theta}_k^{-} + K_k (y_k - h_k(\hat{\theta}_k^{-})) \tag{13}$$

$$P_k^{+} = (I - K_k C_k) P_k^{-} \tag{14}$$

Consider the RBF neural network of Fig. 1 with m inputs, K hidden nodes with $\boldsymbol{c}$ centers, and n outputs. We use vector $\boldsymbol{y} = \begin{bmatrix} y_1 & y_2 & \cdots & y_n \end{bmatrix}^T$ to denote the true/measured value for the RBF NN outputs and $h(\hat{\theta}_k) = \begin{bmatrix} \hat{y}_1 & \hat{y}_2 & \cdots & \hat{y}_n \end{bmatrix}^T$ to denote the actual output of the network at the k^{th} time instant.

$$\begin{pmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{pmatrix} = \begin{pmatrix} w_{10} & w_{11} & w_{12} & \cdots & w_{1K} \\ w_{20} & w_{21} & w_{22} & \cdots & w_{2K} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ w_{n0} & w_{n1} & w_{n2} & \cdots & w_{nK} \end{pmatrix} \begin{pmatrix} 1 \\ \psi_1 \\ \psi_2 \\ \vdots \\ \psi_{\mathrm{K}} \end{pmatrix} \tag{15}$$

The states of the system can be represented as

$$\theta = \begin{pmatrix} w_{10} & w_{11} & \cdots & w_{1K} & w_{20} & w_{21} & \cdots & w_{2K} & \cdots & w_{n0} & w_{n1} & \cdots & w_{n\mathrm{K}} \end{pmatrix}^T$$

The system model to which the Kalman filter can be applied is

$$\theta_{k+1} = \theta_k + \vartheta_k \tag{16}$$

$$y_k = h(\theta_k) + \nu_k \tag{17}$$

The ϑ_k and ν_k are artificially added process and measurements noise to execute a stable Kalman filter algorithm.

3. NEURAL PARTIAL DIFFERENTIATION METHOD

The first order partial derivatives form the basis for linear combination for modelling of force and moment coefficients.

The derivatives are calculated numerically using different methods such as delta and zero methods[9], neural partial differentiation using MLP FFNN[14].

The proposed first order partial differentiation method is explained using RBF neural network in this section. This methodology is based on the partial differentiation of the RBF neural network output and is exactly computed at every training data point. Therefore, it reduces the post-training efforts for partial derivative computation unlike the delta method[9] and its variants. The k^{th} output of RBF NN described in Eqn. (1) can be rewritten as

$$y_k = w_{k0}.1 + w_{k1}\psi_1 + w_{k2}\psi_2 + \ldots\ldots\ldots\ldots + w_{k\mathrm{K}}\psi_{\mathrm{K}} \tag{18}$$

Defining the following vector calculus used to compute the partial derivative of k^{th} output, y_k with respect to input $\boldsymbol{x}$ as

$$\frac{\partial y_k}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial y_k}{\partial x_1} & \frac{\partial y_k}{\partial x_2} & \cdots & \frac{\partial y_k}{\partial x_m} \end{bmatrix} \tag{19}$$

where

$$\frac{\partial y_k}{\partial x_p} = w_{k1}\frac{\partial \psi_1}{\partial x_p} + w_{k2}\frac{\partial \psi_2}{\partial x_p} + \ldots\ldots\ldots\ldots + w_{k\mathrm{K}}\frac{\partial \psi_{\mathrm{K}}}{\partial x_p};$$

$$p = 1, 2, \ldots\ldots\ldots, m \tag{20}$$

$$\frac{\partial y_k}{\partial x_p} = \sum_{i=1}^{\mathrm{K}} w_{ki}\frac{\partial \psi_i}{\partial x_p} \tag{21}$$

$$\frac{\partial y_k}{\partial x_p} = -2\sum_{i=1}^{\mathrm{K}} (x_p - c_p^i)\psi_i w_{ki} \tag{22}$$

$$\frac{\partial y_k}{\partial \boldsymbol{x}} = \begin{bmatrix} -2\sum_{i=1}^{\mathrm{K}} (x_1 - c_1^i)\psi_i w_{ki} \\ -2\sum_{i=1}^{\mathrm{K}} (x_2 - c_2^i)\psi_i w_{ki} \\ \cdots -2\sum_{i=1}^{\mathrm{K}} (x_m - c_m^i)\psi_i w_{ki} \end{bmatrix} \tag{23}$$

Equation (23) is the required relation to compute the partial derivatives of k^{th} output, y_k with respect to input $\boldsymbol{x}$ for multi input-single output (MISO) system. The same result can be extended for multi input-multi output (MIMO) system.

To do this, define the vector calculus of a vector quantity as

$$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_m} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_m} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial y_n}{\partial x_1} & \frac{\partial y_n}{\partial x_2} & \cdots & \frac{\partial y_n}{\partial x_m} \end{bmatrix} \tag{24}$$

Now, putting the values of $\frac{\partial y_k}{\partial x_p}$ in the above expression we get

$$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = -2\begin{bmatrix} \sum_{i=1}^{\mathrm{K}} (x_1 - c_1^i)\psi_i w_{1i} & \sum_{i=1}^{\mathrm{K}} (x_2 - c_2^i)\psi_i w_{1i} & \cdots & \sum_{i=1}^{\mathrm{K}} (x_m - c_m^i)\psi_i w_{1i} \\ \sum_{i=1}^{\mathrm{K}} (x_1 - c_1^i)\psi_i w_{2i} & \sum_{i=1}^{\mathrm{K}} (x_2 - c_2^i)\psi_i w_{2i} & \cdots & \sum_{i=1}^{\mathrm{K}} (x_m - c_m^i)\psi_i w_{2i} \\ \vdots & \vdots & \vdots & \vdots \\ \sum_{i=1}^{\mathrm{K}} (x_1 - c_1^i)\psi_i w_{ni} & \sum_{i=1}^{\mathrm{K}} (x_2 - c_2^i)\psi_i w_{ni} & \cdots & \sum_{i=1}^{\mathrm{K}} (x_m - c_m^i)\psi_i w_{ni} \end{bmatrix} \tag{25}$$

This is the required relation for computing the first order derivatives in the framework of radial basis function network.

4. FLIGHT TEST DATA

The flight data of ATTAS aircraft of DLR is used to demonstrate the aerodynamic parameter estimation capability using the proposed neural partial differentiation method. The flight data of three different manoeuvers performed at altitude 16000 ft and at 200 kts nominal speed are analysed here. The three manoeuvers are: (i) short period motion excited by multistep elevator, (ii) bank-to-bank motion by aileron input, and (iii) Dutch roll motion by the doublet rudder input. The elevator input is considered in lateral-directional dynamics to extract the cross coupling effects. The three manoeuvers are 25 s, 30 s, and 30 s long respectively. The flight data is recorded at a sampling rate of 25 Hz.

The time history of motion variables (β, p, r) and applied control inputs (δ_a, δ_r) during these three flight manoeuvers are as shown in Fig. 2.

5. PARAMETER ESTIMATION FROM FLIGHT DATA

The lateral-directional nondimensional coefficients C_Y, C_l and C_n are derived from analytically computed side force (Y), rolling moment (L) and yawing moment (N) respectively. The Y, L and N are computed using flight measured linear accelerations and angular rate gyros. All the relevant equations are well explained[2,40].

The model for lateral-directional aerodynamic force and moment coefficients is postulated as follows:

$$C_Y = C_{Y_0} + C_{Y_\beta}\beta + C_{Y_p}p^* + C_{Y_r}r^* + C_{Y_{\delta_a}}\delta_a + C_{Y_{\delta_r}}\delta_r \tag{26}$$

$$C_l = C_{l_0} + C_{l_\beta}\beta + C_{lp}p^* + C_{lr}r^* + C_{l_{\delta_a}}\delta_a + C_{l_{\delta_r}}\delta_r \tag{27}$$

$$C_n = C_{n_0} + C_{n_\beta}\beta + C_{n_p}p^* + C_{n_r}r^* + C_{n_{\delta_a}}\delta_a + C_{n_{\delta_r}}\delta_r \tag{28}$$

where δ_a : the aileron deflection, δ_r : the rudder deflection, β : the angle of sideslip, (p^*, r^*) : the normalised angular rates.

The unknown aerodynamic stability and control derivatives are $C_{Y_\beta}, C_{Y_p}, C_{Y_r}, C_{l_\beta}, C_{l_p}, C_{l_r}, C_{n_\beta}, C_{n_p}, C_{n_r}$ and $C_{Y_{\delta_a}}, C_{Y_{\delta_r}}, C_{l_{\delta_a}}, C_{l_{\delta_r}}, C_{n_{\delta_a}}, C_{n_{\delta_r}}$ respectively.

The coefficient parameters (C_Y, C_l and C_n) are used to train the RBF neural network. The neural network input and output vectors for aerodynamic parameter estimation are defined as

Input: $\boldsymbol{x} = \begin{bmatrix} \beta & p^* & r^* & \delta_a & \delta_r \end{bmatrix}^T$

Output: $\boldsymbol{y} = \begin{bmatrix} C_Y & C_l & C_n \end{bmatrix}^T$

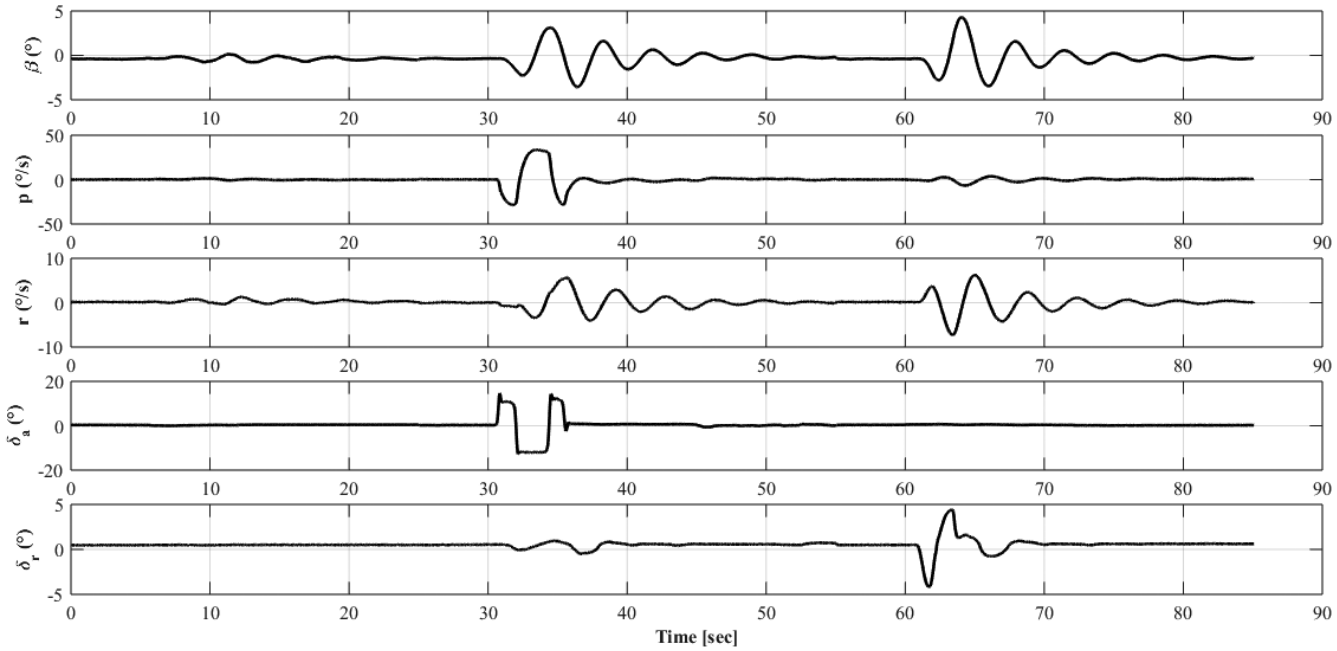

Figure 2. Time histories of motion/control inputs.

The RBF neural network applied for aerodynamic parameter modelling is illustrated in Fig. 3.

For the purpose of mathematical explanation one of the output and associated single derivative is explained. However, similar mathematics is applicable for all other outputs and their derivatives. Lateral side force coefficient (C_Y) and stability derivative (C_{Y_β}) are considered as network output and associated partial derivative respectively for mathematical derivation. Rewriting the RBF neural network output as

$$C_Y = w_{10} + w_{11}\psi_1 + w_{12}\psi_2 + \ldots\ldots\ldots\ldots + w_{1,10}\psi_{10} \qquad (29)$$

where nonlinear functions $\psi_1, \psi_2, \ldots\ldots\psi_{10}$ are defined as

$$\psi_1 = e^{-[(\beta-c_1^1)^2+(p^*-c_2^1)^2+(r^*-c_3^1)^2+(\delta_a-c_4^1)^2+(\delta_r-c_5^1)^2]}$$

$$\psi_2 = e^{-[(\beta-c_1^2)^2+(p^*-c_2^2)^2+(r^*-c_3^2)^2+(\delta_a-c_4^2)^2+(\delta_r-c_5^2)^2]}, \text{ and}$$

$$\psi_{10} = e^{-[(\beta-c_1^{10})^2+(p^*-c_2^{10})^2+(r^*-c_3^{10})^2+(\delta_a-c_4^{10})^2+(\delta_r-c_5^{10})^2]}$$

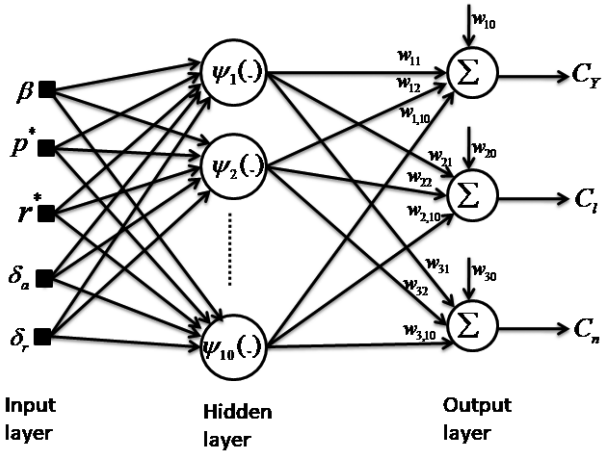

Figure 3. Aerodynamic force and moments modelling using RBF neural network.

Then lateral force stability derivative C_{Y_β} is computed by partial derivative of C_Y w.r.t. β as

$$\frac{\partial C_Y}{\partial \beta} = 0 + w_{11}\frac{\partial \psi_1}{\partial \beta} + w_{12}\frac{\partial \psi_2}{\partial \beta} + \ldots\ldots\ldots\ldots + w_{1,10}\frac{\partial \psi_{10}}{\partial \beta} \qquad (30)$$

where

$$\frac{\partial \psi_1}{\partial \beta} = -2(\beta - c_1^1)e^{-[(\beta-c_1^1)^2+(p^*-c_2^1)^2+(r^*-c_3^1)^2+(\delta_a-c_4^1)^2+(\delta_r-c_5^1)^2]} = -2(\beta - c_1^1)\psi_1$$

$$\frac{\partial \psi_2}{\partial \beta} = -2(\beta - c_1^2)e^{-[(\beta-c_1^2)^2+(p^*-c_2^2)^2+(r^*-c_3^2)^2+(\delta_a-c_4^2)^2+(\delta_r-c_5^2)^2]} = -2(\beta - c_1^2)\psi_2$$

, and so

$$\frac{\partial \psi_{10}}{\partial \beta} = -2(\beta - c_1^{10})e^{-[(\beta-c_1^{10})^2+(p^*-c_2^{10})^2+(r^*-c_3^{10})^2+(\delta_a-c_4^{10})^2+(\delta_r-c_5^{10})^2]} = -2(\beta - c_1^{10})\psi_{10}$$

$$\frac{\partial C_Y}{\partial \beta} = -2(\beta - c_1^1)\psi_1 w_{11} - 2(\beta - c_1^2)\psi_2 w_{12} - \ldots\ldots\ldots - 2(\beta - c_1^{10})\psi_{10} w_{1,10} \qquad (31)$$

As already pointed out, the RBF NN is trained by k-means clustering algorithm at input layer and weights in output layer are obtained using nonlinear Extended Kalman Filter.

Judicious choice is necessary on K, the number of neurons in the hidden layer. Simulations show that seven or more neurons are able to model the aerodynamic charactesistic of the aircraft. However, increasing the value of K in hidden layer improves the residual error performance but at the increased cost of computational efforts. In the present case, $K = 10$ is used for simulation.

The process and measurement noise covariance matrices used in output layer weight estimation using EKF are chosen as $Q = 10^{-7} I_{33\times 33}$ and $R = 10^{-2} I_{3\times 3}$ respectively. These values are determined by Monte-Carlo simulations for 10000 runs.

The EKF algorithm modified for training to RBF neural network is formulated by adding fictitious noise to the system

model with zero mean and covariance Q.

$$w_{k+1} = w_k + \vartheta_k \tag{32}$$

$$y_k = h_k(w_k) + \nu_k \tag{33}$$

$$P_k^- = A_{k-1}P_{k-1}^+A_{k-1}^T + Q_{k-1} \tag{34}$$

$$K_k = P_k^-C_k^T(C_kP_k^-C_k^T + R_k)^{-1} \tag{35}$$

$$\hat{w}_{k+1} = \hat{w}_k + K_k(y_k - h_k(\hat{w}_k)) \tag{36}$$

where $\boldsymbol{w} = \left(w_{10} \quad w_{11} \quad \cdots \quad w_{1K} \quad w_{20} \quad w_{21} \quad \cdots \quad w_{2K} \quad \cdots \quad w_{n0} \quad w_{n1} \quad \cdots \quad w_{n\mathrm{K}}\right)^T$

The function h is described by the RBF neural network outputs mentioned in Eqn. (26-28). The state Jacobian matrix A_k is the identity matrix. The measurement Jacobian matrix becomes $C_k = \left.\frac{\partial h}{\partial w}\right|_{\hat{w}_k}$.

The training algorithm is iterative in nature and starts with any random initial weights (zero or non-zero), the EKF algorithm repeats for all data points to update the weights recursively. The mean square error is computed after each iteration defined by

$$MSE = \frac{1}{Nn}\sum_{i=1}^{N}(\boldsymbol{y} - \hat{\boldsymbol{y}})^T(\boldsymbol{y} - \hat{\boldsymbol{y}}) \tag{37}$$

In this paper, the MSE was used as stopping criterion for termination of RBF neural network learning.

Simulations were carried out for all lateral-directional force and moment coefficients. Lateral side force and associated stability and control derivatives are illustrated for the sake of brevity. Figure 4 shows a comparison between flight measured force coefficient and neural network estimate during the training phase. The result show good matching. The stability and control derivatives calculated using RBF NN neural partial differentiation method (PDM) during training at each data point is as given in Table 1. The average values of lateral-directional derivatives using RBF NN neural partial differentiation method are compared with RBF NN delta method, partial differentiation with multi-layer feedforward neural network and equation error method. It is observed that derivatives obtained by proposed method are consistent with other methods.

The side force control and stability derivatives are illustrated in Fig. 5. It is obvious that there is a little contribution of aileron in side force and same is noticed in Fig. 5. However, there is a slight disturbance in $C_{Y_{\delta_a}}$ when excited through the

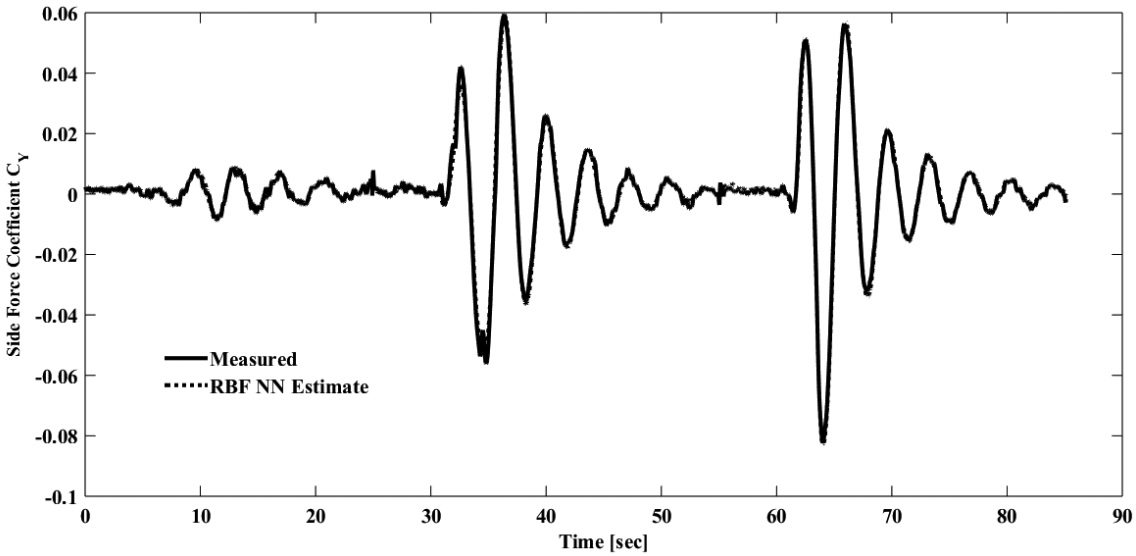

Figure 4. Comparison of RBF NN estimate and measured side force coefficient (C_Y).

Table 1. Average value of estimated stability and control derivatives using RBF neural partial differentiation, delta method for RBF Neural Network, MLP partial differentiation and equation error

Method	C_{Y_0}	C_{l_0}	C_{n_0}	C_{Y_β}	C_{l_β}	C_{n_β}	C_{Y_p}	C_{l_p}	C_{n_p}
RBF NN PDM	-0.0072	-0.0001	0.0029	-1.0620	-0.1149	0.2610	0.1786	-0.7788	-0.0843
RBF NN delta method	-0.0072	-0.0001	0.0029	-1.0599	-0.1150	0.2607	0.1784	-0.7757	-0.0844
FFNN PDM[14]	-0.0071	-0.0002	0.0029	-1.0557	-0.1134	0.2587	0.2027	-0.7584	-0.0912
EEM[14]	-0.0071	-0.0002	0.0029	-1.0483	-0.1127	0.2572	0.2058	-0.7557	-0.0921
Method	C_{Y_r}	C_{l_r}	C_{n_r}	$C_{Y_{\delta_a}}$	$C_{l_{\delta_a}}$	$C_{n_{\delta_a}}$	$C_{Y_{\delta_r}}$	$C_{l_{\delta_r}}$	$C_{n_{\delta_r}}$
RBF NN PDM	0.5823	0.2535	-0.1156	0.0046	-0.2034	-0.0108	0.1877	0.0380	-0.1432
RBF NN delta method	0.5847	0.2532	-0.1192	0.0046	-0.2028	-0.0108	0.1867	0.0380	-0.1446
FFNN PDM[14]	0.6116	0.2809	-0.1252	0.0080	-0.1934	-0.0118	0.1902	0.0442	-0.1419
EEM[14]	0.6157	0.2866	-0.1265	0.0083	-0.1928	-0.0119	0.1909	0.0439	-0.1430

aileron. This is may be due to the adverse yaw effect. Side force is majorly due to side slip angle β and yawing angular rate r and is the reason why stability derivatives C_{Y_β} and C_{Y_r} are dominant among the derivatives. Apparently remaining terms are not so effective. The RMS values of derivatives obtained using RBF NN PDM is illustrated in Fig. 6. The training of RBF NN converges within 5-10 iterations and is confirmed through result as shown in Fig. 6.

The predictive capability of the identified lateral-directional model is proved in two ways

(i) A separate set of flight data is applied to identified RBF NN model, and

(ii) Identified stability and control derivatives are used to reconstruct the force and moment coefficients for this flight data.

Result for validation method 1 is illustrated in Fig. 7. Neural network identified model output is compared with measurement values and show good matching. The reconstructed side force coefficient is illustrated in Fig. 8. Both the model output and reconstructed coefficient values are in close agreement with the measured values. This proves the acceptance of the identified model and derivative values.

6. CONCLUSIONS

Applicability of the RBF neural network is investigated to model the aircraft dynamics. Model for lateral-directional nondimensional force and moment coefficients are identified for ATTAS aircraft. The model is validated using a different data set and reconstruction method. A new analytical method for estimation of stability and control derivatives is proposed using RBF neural network. The first order partial differentiation is performed at each training data point to obtain the stability and control derivatives. The analytically computed derivatives by the proposed method are compared with hitherto methods such as delta method, partial differential method using MLP and equation error method. The results so obtained are

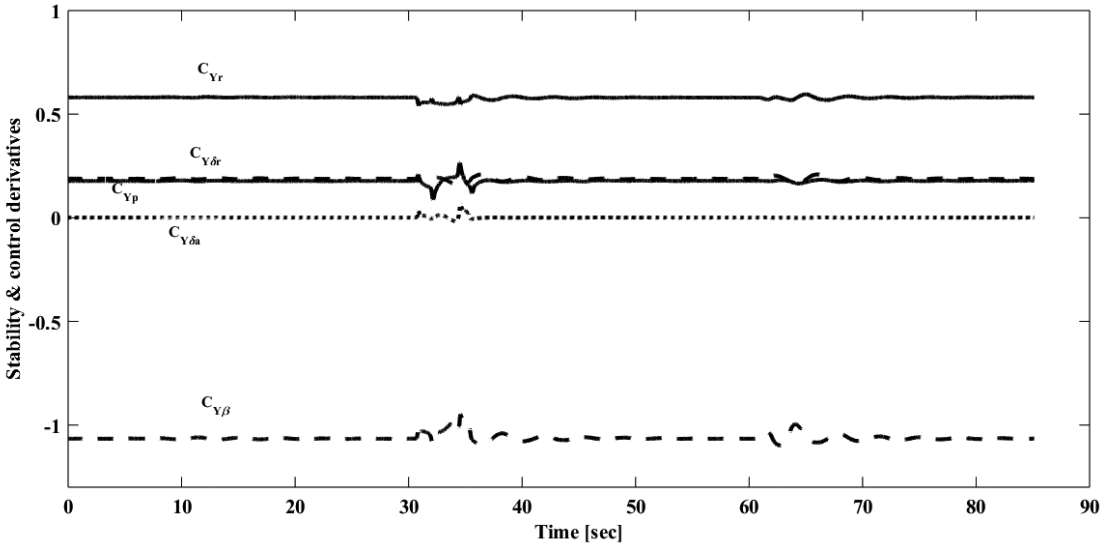

Figure 5. Stability and control derivatives associated with side force (Y) computed using RBF NN PDM.

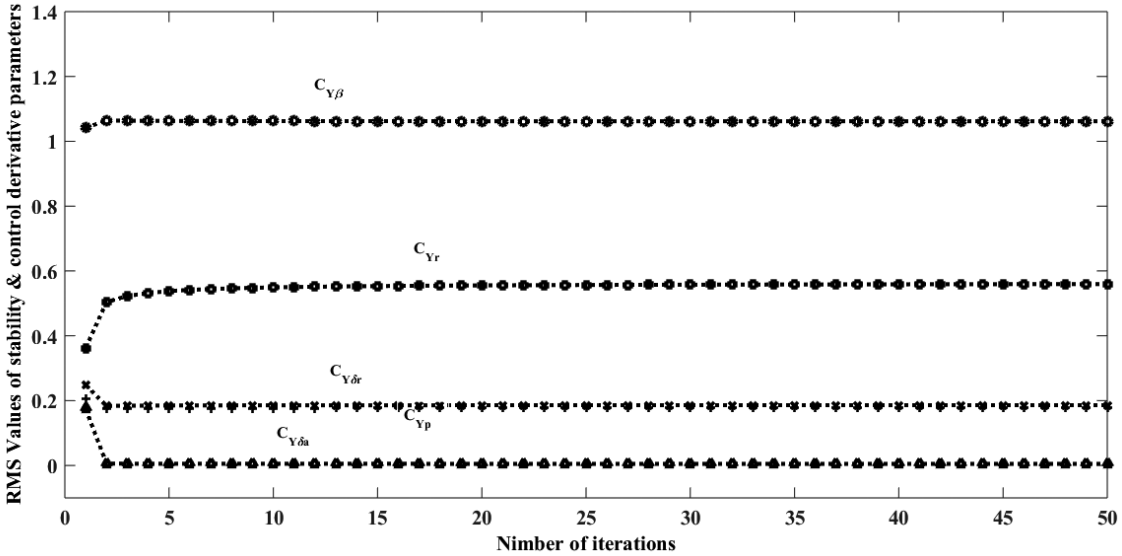

Figure 6. RMS values of stability and control derivatives of side force (Y) at each iteration.

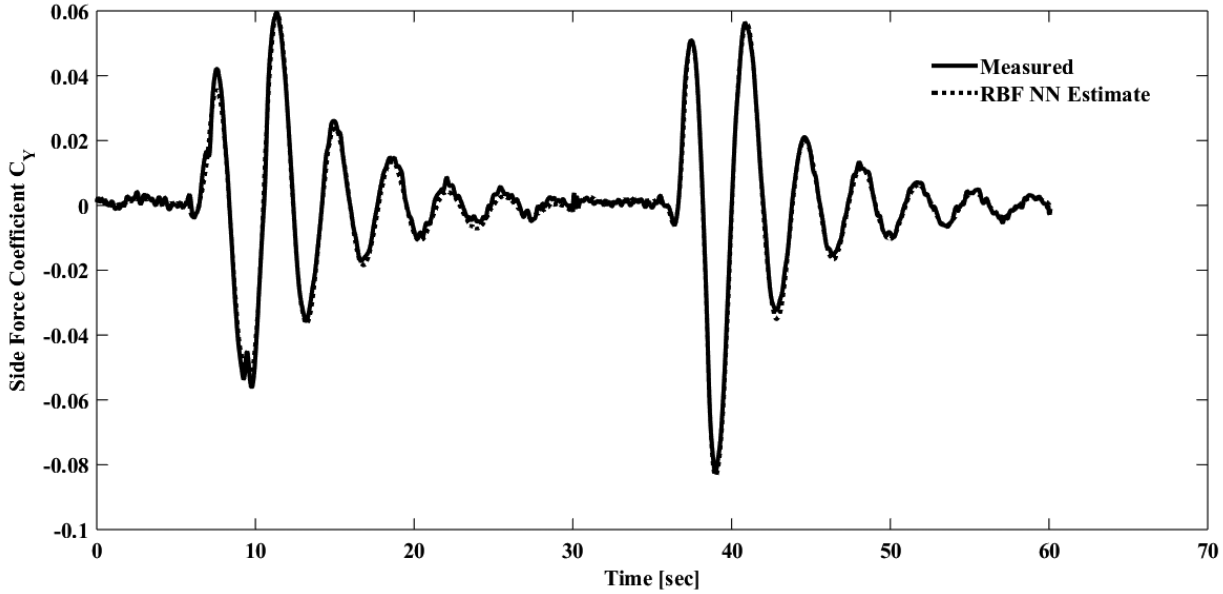

Figure 7. Comparison of RBF NN estimate and measured side force coefficient (C_Y) during neural network testing phase (Validation method 1).

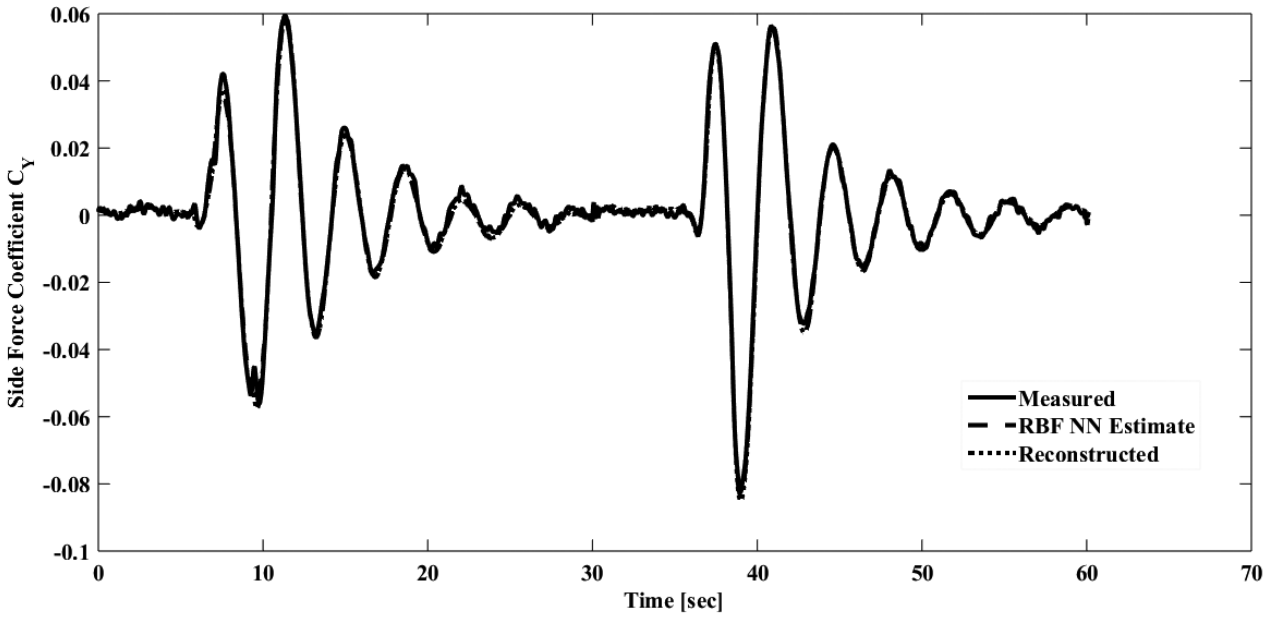

Figure 8. Comparison of RBF NN estimate, Measured and Reconstructed side force coefficient, C_Y (Validation method 2).

consistent with other methods. The advantage of using RBF neural network provides faster convergence compared to MLP neural networks. The proposed method can be applied for complete aerodynamic characterisation of fixed wing as well as rotorcraft flight vehicles.

REFERENCES

1. Klein, V. & Morelli, E. Aircraft system identification: Theory and practice. Education Series, AIAA Inc., Reston, VA, 2006.
2. Jategaonkar, R.V. Flight vehicle system identification: A time domain methodology. Ed 2nd, Progress in Astronautics and Aeronautics, AIAA Inc., Reston, Virginia, 2015, 245.
3. Tischler, M.B. & Remple, R. K. Aircraft and rotorcraft system identification: Engineering methods with. flight test examples. Ed 2nd. Education Series, AIAA Inc., Reston, VA, 2012.
4. Hamel, P.G. & Jategaonkar, R.V. Evolution of flight vehicle system identification. *Journal Aircraft*, 1989, **33**(5), 609-622.
5. Hess, R.A. On the use of back propagation with feedforward neural networks for aerodynamic estimation problem. AIAA Paper 93-3638-CP, Aug. 1993. doi: 10.2514/6.1993-3638
6. Youseff, H.M. & Juang, J.C. Estimation of aerodynamic coefficients using neural networks. AIAA Paper 93-3639-CP, Aug. 1993. doi: 10.2514/6.1993-3639

7. Linse, D.J. & Stengel, R.F. Identification of aerodynamic coefficients using computational neural networks. *J. Guidance, Control, Dynamics*, 1993, **16**(6), 1018-1025. doi: 10.2514/3.21122.
8. Basappa, K. & Jategaonkar, R.V. Aspects of feedforward neural network modeling and its application to lateral-directional flight data. DLR Institute's Report 111-95/30, Braunschweig, Germany, Sept. 1995.
9. Raisinghani, S.C.; Ghosh, A.K. & Kalra, P.K. Two new techniques for parameter estimation using neural networks. *Aeronautical Journal*, 1998, **102**(1011), 25-29.
10. Raisinghani, S.C.; Ghosh, A.K. & Khubchandani, S. Estimation of aircraft lateral-directional parameters using neural networks. *Journal Aircraft*, 1998, **35**(6), 876-881. doi: 10.2514/2.2407
11. Raisinghani, S.C. & Ghosh, A.K. Parameter estimation of an aeroelastic aircraft using neural networks. *Sadhana*, 2000, 25, Part 2, 181-191.
12. Ghosh, A.K. & Raisinghani, S.C. Frequency-domain estimation of parameters from flight data using neural networks. *J. Guidance, Control, Dyn.*, 2001, **24**(3), 525-530.
13. Peyada, N.K. & Ghosh, A.K. Aircraft parameter estimation using neural network based algorithm. *In* AIAA Atmospheric Flight Mechanics Conference, 10-13 Aug., 2009, Chicago, Illinois.
14. Das, S.; Kuttieri, R.A.; Sinha, M. & Jategaonkar, R. Neural partial differential method for extracting aerodynamic derivatives from flight data. *J. Guidance, Control, Dyn.*, 2010, **33**(2), 376-384. doi: 10.2514/1.46053
15. Sinha, M. & Kuttieri, R.A. High angle of attack parameter estimation of cascaded fins using neural network. *Journal Aircraft*, 2013, **50**(1), 272-291. doi: 10.2514/1. C031912
16. Sinha, M.; Kuttieri, R.A. & Chatterjee, S. Nonlinear and linear unstable aircraft parameter estimations using neural partial differentiation. *J. Guidance, Control, Dyn.*, 2013, **36**(4), 1162-1176. doi: 10.2514/1.57029
17. Dongare, V. & Mohamed, V. Lateral-directional aerodynamic parameter estimation using neural partial differentiation. *In* International Conference on Cognitive Computing and Information Processing (CCIP), 2015, Noida, India, 3-4 March, 2015. doi: 10.1109/CCIP.2015.7100730
18. Chauhan, R.K. & Singh, S. Application of neural networks based method for estimation of aerodynamic derivatives. *In* 7th International Conference on Cloud Computing, Data Science & Engineering- Confluence, 12-13 Jan 2017, Noida, India. doi: 10.1109/CONFLUENCE.2017.7943124
19. Kumar, R.; Omkar, S.N. & Ganguli, R. Computation of rotorcraft aerodynamic derivatives using neural networks. *In* Heli-Japan 2006 AHS International Meeting on Advance Rotorcraft Technologies and Life Saving Activities, Nagoya, Aichi Prefecture, Japan, 15-17 Nov., 2006, pp. T213-2-1-T213-2-17.
20. Kumar, R.; Ganguli, R.; Omkar, S.N. & Kumar, M.V. Rotorcraft parameter identification from real time flight data. *Journal Aircraft*, 2008, **45**(1). doi:10.2514/1.320.24
21. Hornik, K.; Stinchcombe, M. & White, H. Multilayer feedforward networks are universal approximators. *Neural Networks*, 1989, **2**, 359-366.
22. Broomhead, D.S. & Lowe, D. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 1988, **2**(3), 321-355.
23. Park, J. & Sandberg, I.W. Universal approximation using radial basis function networks. *Neural Computation*, 1991, **3**, 246-257.
24. Moody, J. & Darken, C.J. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1989, **1**, 281–294.
25. Chen, S.; Cowan, C. & Grant, P. Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Trans. Neural Networks*, 1991, **2**, 302-309.
26. Duchon, J. Splines minimizing rotation-invariant semi-norms in Sobolev spaces. Constructive Theory of Functions of Several Variables (Lecture Notes in Math 571), New York, Springer, 1977, pp. 85-100.
27. Hardy, R.L. Multiquadric equations of topography and other irregular surfaces. *J. Geophys. Res.,* 1971, **76**, 1905-1915,.
28. Schagen, I.P. Sequential exploration of unknown multi-dimensional functions as an aid to optimization. *IMA J. Numerical Anal.*, 1984, **4**, 337-347.
29. Verma, N.K. & Roy, A. Self optimal clustering technique using optimized threshold function. *IEEE Sys. J.*, 2013, **99**, 1-14.
30. Gustafson, D.E. & Kessel, W.C. Fuzzy clustering with a fuzzy covariance matrix. *In* 1978 IEEE Conference on Decision and Control Including the 17th Symposium on Adaptive Processes, San Diego, CA, USA, 10-12 Jan., 1979
31. Dun, J.C. A Fuzzy Relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal Cybernetics*, 1973, **3**(3).
32. MacQueen, J. Some methods for classification and analysis of multivariable observations. *In* Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. University of California Press, Berkeley, California, 1967, **1**, pp. 281-297.
33. Verma, N.K.; Roy, A. & Cui Y. Improved mountain clustering algorithm for gene expression data analysis. *J. Data Mining Knowledge Discovery*, 2011, **2**(1), 30-35.
34. Gath, I. & Geva, A.B. Unsupervised optimal fuzzy clustering. *IEEE Trans. Pattern Analysis Machine Intelligence*, 1989, **11**(7), 773-781. doi: 10.1109/34.192473
35. Karayiannis, N. Reformulated radial basis neural networks trained by gradient descent. *IEEE Trans. Neural Networks*, 1999, **10**, 657-671.
36. Sum, J.; Leung, C. & Kan, W. On the Kalman filtering method in neural network training and pruning. *IEEE Trans. Neural Networks*, 1999, **10**, 161-166.

37. Duro, R. & Reyes, J. Discrete-time backpropagation for training synaptic delay-based artificial neural networks. *IEEE Trans. Neural Networks*, 1999, **10**, 779-789.
38. Narendra, K. & Thathachar, M. Learning automata - An introduction. Prentice-Hall, Englewood Cliffs, NJ, 1989.
39. Kirkpatrick, S.; Gelatt, C. & Vecchi, M. Optimization by simulated annealing. *Science,* 1983, **220**, 671-680.
40. Chen, S.; Wu, Y. & Luk, B. Combined genetic algorithm optimization and regularized orthogonal least squares learning for radial basis function networks. *IEEE Trans. Neural Networks*, 1999, **10,** 1239-1243.
41. Haykin, S. Adaptive Filter Theory. Fourth Edition, Pearson Education, 2007.
42. Simon, D. Optimal State Estimation: Kalman, H∞, and Nonlinear Approaches. A John Wiley & Sons, Inc., Publication, NJ, 2006.
43. Jategaonkar, R.V. Identification of the Aerodynamic Model of the DLR Research Aircraft ATTAS from Flight Test Data. DLR-FB 90-40, Braunschweig, Germany, 1990.

ACKNOWLEDGMENT

The author is grateful to Mr R.V. Jategaonkar for granting permission to use the flight data of ATTAS aircraft.

CONTRIBUTORS

Mr Jitu Sanwale has obtained his BE (Electronics & Instrumentation) and MTech (Communication Systems) from SGSITS Indore and IIT Roorkee, in 2006 and 2008, respectively. Presently, he is working as Manager (Design) in Hindustan Aeronautics Limited, Nasik. His areas of interest include: System identification, flight controller, autopilot and navigation systems of fixed wing aircrafts.

In the present study, he has done all the theoretical and simulation studies for aerodynamic parameter estimation of ATTAS aircraft.

Mr Dhan Jeet Singh has obtained his BE (Electronics & Communication Engineering) from B.I.E.T Jhansi, in 2005 and perusing MS (Research) in Control and Automation from Department of Electrical Engineering, IIT Kanpur. Presently, he is working as Sr. Manager (Design) in Aircraft Upgrade Research & Design Centre (AURDC) of Hindustan Aeronautics Limited, Nasik. He has research experience in flight control and navigation systems of fixed wing fighter aircrafts.

In the current study, he has contributed in the analysis of results and training of neural network at first layer by k-means clustering.