IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 6, 2019, accepted January 17, 2019, date of publication February 4, 2019, date of current version February 27, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2897487

Investigating the Generalization Capability and Performance of Neural Networks and Neuro-Fuzzy Systems for Nonlinear Dynamics Modeling of Impaired Aircraft

RAMIN NOROUZI, (Member, IEEE), AMIRREZA KOSARI, (Member, IEEE), AND MOHAMMAD HOSSEIN SABOUR
Faculty of New Sciences and Technologies, University of Tehran, Tehran 1439957131, Iran
Corresponding author: Mohammad Hossein Sabour (sabourmh@ut.ac.ir)

ABSTRACT One of the challenging problems in the case of aircraft failure is to determine the new altered dynamics of the impaired aircraft. Among various methods, neural networks and neuro-fuzzy systems can be used for high-fidelity modeling of the aircraft nonlinear dynamics with the aim of onboard applications in real time. However, the method with better generalization capability is more preferred specifically in the case of unpredicted aircraft failures. Generalization of a network is mainly dependent on the network's parameters, the employed training algorithm, and the amount of training data. In this paper, several neural networks and local model networks are trained using different training algorithms and different amounts of training data to model the nonlinear dynamics of an impaired aircraft with the damaged rudder. These networks are compared based on their generalizations to the new cases of rudder failure. The effect of using different amounts of training data on the generalization capability and performance of the networks has also been investigated. The results of this paper show that both network types have good performance but neural networks generalize better to the new failure cases than local model networks. Also based on the obtained results, a significant reduction in the number of training samples could be accomplished without a considerable decrease in the network's performance and generalization. Finally, a neural network-based sensitivity analysis method is proposed which utilizes the network's regression equation as an emulator for fast model evaluations and can be used as an advisory tool for choosing safer path planning strategies.

INDEX TERMS Artificial neural network, local model network, generalization, Bayesian regularization, LOLIMOT, HILOMOT, global sensitivity analysis.

I. INTRODUCTION

In the case of technical failures or external events such as control surface defects or icing, aircraft dynamics and parameters are changed. Due to the nonlinear dynamics of aircraft, usually the exact new altered dynamics cannot be determined by the pilot. Therefore the pilot who tries to plan a safe landing trajectory as soon as possible may implement a maneuver which is not feasible anymore according to the altered dynamics of the impaired aircraft and leads to aircraft loss of control (LOC).

Based on statistical reports published by Boeing and UK Civil Aviation Authority, Loss of Control (LOC) is the primary contributor among different factors causing fatal accident of commercial airliners [1], [2]. The number of fatal accidents has been decreasing despite the increase in the number of flights [2], however still LOC holds the greatest share in fatal accidents, despite all improvements made to pilot trainings and aircraft systems.

To avoid LOC, states, control inputs, and maneuvers' characteristics of the impaired aircraft should be within the admissible ranges of the degraded performance of the aircraft [3]. Generally, aircraft performance is characterized by its flight envelope which is dictated by the nonlinear dynamics of the aircraft. Hence, the degraded performance of an impaired aircraft is depicted in its new flight envelope; which is more confined than the nominal flight envelope of the unimpaired aircraft. However, it should be noted that different failure

The associate editor coordinating the review of this manuscript and approving it for publication was Bilal Alatas.

2169-3536 © 2019 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

degrees result in different flight envelopes [4]. In other words, the performance capabilities of the impaired aircraft vary with failure cases. Therefore it is crucial to evaluate the permissible ranges of the impaired flight envelope's characteristics and parameters as fast as possible after the occurrence of the failure, based on the specific occurred failure case.

Due to the need of immediate flight envelope evaluation in the event of failure, researchers have tried to develop numerically efficient flight envelope estimation techniques feasible for real-time execution to integrate them into adaptive flight envelope protection systems. Generating an offline database of impaired flight envelopes is one of these techniques. For instance, [5] presents the Envelope-Aware Flight Management System (EA-FMS) which is an augmentation to a conventional flight management system designed to prevent loss of control. In this system, it is assumed that the failure is characterized a priori and hence envelope estimation is performed offline for various failure degrees. Also, it is presumed that the offline generated database is applicable to any specific case online. In [6], first the actuator fault severity level is detected and then the new flight envelope is estimated via online interpolation of the flight envelopes from an offline database. A similar method for online flight envelope interpolation is proposed in [7]. However, using an offline database of impaired flight envelopes is only applicable to failures characterized a priori and cannot be used in case of unpredicted failures. Also, online interpolation of the flight envelopes from an offline database requires carrying massive databases onboard, and the result of the interpolation might be not accurate enough especially if the offline flight envelopes are estimated in high resolution based on high-fidelity nonlinear aircraft models.

Another technique is to estimate the new flight envelope only in the neighboring vicinity of the current trim state of the impaired aircraft. In this technique, flight envelope can be defined as a set of attainable trim states within a set of constraints, where trim state is defined as in section II. Loss of control may occur once any of the constraints is violated [3]. Due to the confined optimization space, real-time local flight envelope estimation is computationally feasible. In this method, local flight envelopes are estimated progressively as new flight conditions are visited. For instance, in [6], the reachable set theory is used to estimate local flight envelopes for airframe faults. However, online local flight envelope estimation (e.g. using approaches presented in [8]) is only fast enough for real-time evaluation of low-resolution flight envelopes, and estimating high-resolution flight envelopes demands non-negligible amount of time which must be spent at each step that a local flight envelope is calculated. Additionally, since the entire flight envelope is not apparent initially, the pilot may unintentionally choose to achieve a state or control which is beyond the admissible ranges of the impaired flight envelope.

Utilizing reduced complexity models instead of high-fidelity models is another technique which enables fast estimation of the flight envelope. This technique has been used in [9] and [10] to evaluate unimpaired and impaired flight envelopes. However, flight envelopes estimated using reduced complexity models are considerably simplified, and also lack the maneuvers' characteristic required for high fidelity nonlinear 6 degree-of-freedom post-failure emergency path planning.

According to the presented explanations, none of the aforementioned methods are capable of real-time evaluation of the permissible ranges of the high-fidelity impaired flight envelope parameters for an unpredicted failure degree. They are either based on simplified models or exploit high-fidelity nonlinear models to generate offline databases or evaluate online local flight envelopes.

On the other hand, due to significant learning capabilities of the Artificial Neural Networks (ANNs) and Local Model Networks (LMNs), they can be employed for system identification and modeling of the aircraft nonlinear dynamics. These networks can be trained offline and predict the intended dynamic parameters online in real-time. For instance, in [11], ANNs have been used to model the nonlinear unsteady behavior of aircraft at high angles of attack, whereas in [12] ANNs are used for nonlinear aircraft system identification. In [13], ANNs are utilized to develop an unsteady aerodynamic modeling method applied into coupled oscillations during post-stall maneuvers, and in [14], ANNs are used to model the nonlinear unsteady aerodynamics at constant or varying Mach numbers.

In [15], local linear models have been trained by LOLIMOT (LOcal LInear MOdel Tree) algorithm for predicting unsteady aerodynamic loads, whereas in [16], LMNs have been used to identify the nonlinear aerodynamic derivatives of aircraft. Also, LMNs have been used to model the airflow sensor calibration data [17] and combustion plant characteristics [18], successfully. In [19], the aeroacoustic behavior of aircraft air distribution system was predicted using LMNs, and in [20], LMNs were used in flight loads estimation of a transport aircraft.

Even though both ANNs and LMNs are able to model the nonlinear dynamics of aircraft, they are practically efficient only if they are able to generalize well to new data. This paper provides more in-depth and extended coverage of our previous research published in [21]. First, ANNs and LMNs are developed and trained with the aim of predicting the minimum speed (V_{min}) and maximum speed (V_{max}) of an impaired NASA Generic Transport Model (GTM) in the case of rudder failure. The trained networks are then compared based on their generalizations to unpredicted cases of rudder failure, and also with respect to different amounts of training data, where these training datasets are portions of the original training data; diminished with respect to the networks' input parameters. Furthermore, it is shown that the trained ANN's regression equation can be used for fast model evaluations required for accurate global sensitivity analysis.

The rest of this paper is organized as follows; in section II, the research methodology is presented including the networks' architecture, and training algorithms. In section III, the dynamic model used in this research is introduced, the training data generation and preparation process is explained, and the numerical results of the comparison between the ANN's and LMN's generalization are presented. Section IV introduces a neural network-based sensitivity analysis method along with numerical results and discussions. Finally, section V concludes the paper.

II. METHODOLOGY

As explained earlier, aircraft performance is depicted in its flight envelope. Generally, commercial airliners use steady state maneuvers as trajectory segments between a specified sequence of waypoints. Trajectory waypoints are either connected by steady state rectilinear maneuvers (i.e. direct routes) or steady state turning maneuvers (i.e. heading adjustment routes) [22]. Hence, when dealing with aircraft maneuverability, the parameters of maneuvering flight envelopes better describe the aircraft performance.

Maneuvering flight envelopes (MFEs) are defined as boundaries containing steady state maneuvers. Such steady state maneuvers are referred to as trim points in this research. A steady state maneuver is considered as the condition in which all force and moment components in the body-fixed coordinate system are zero or constant [23]. This requires all linear and angular velocity rates and aerodynamic angles rates to be zero, whilst controls are fixed.

$$(\dot{u}, \dot{v}, \dot{w}) = (\dot{p}, \dot{q}, \dot{r}) = (\dot{\alpha}, \dot{\beta}) = 0 \tag{1}$$

where, u, v, w, are airspeed components in the body-fixed axes and p, q, r, are roll rate, pitch rate and yaw rate, respectively. (1), is a general definition of trim state, however, the steady state maneuvers considered in this research are the two groups of level/climbing/descending rectilinear flight and level/climbing/descending turning flight. Therefore additional constraints need to be imposed:

$$\text{Level rectilinear: } \dot{\phi}, \dot{\theta}, \dot{\psi}, \gamma = 0 \tag{2}$$

$$\text{Climbing/descending rectilinear: } \dot{\phi}, \dot{\theta}, \dot{\psi} = 0, \gamma = cte \tag{3}$$

$$\text{Level turn: } \dot{\phi}, \dot{\theta}, \gamma = 0, \dot{\psi} = cte \tag{4}$$

$$\text{Climbing/descending turn: } \dot{\phi}, \dot{\theta} = 0, \dot{\psi}, \gamma = cte \tag{5}$$

where, ϕ, θ, ψ, and γ are roll (bank) angle, pitch angle, yaw (heading) angle, and flight path angle.

The trim state definition in terms of aircraft equations of motion can be written as

$$\dot{x}_{trim} = f\left(x_{trim}, u_{trim}\right) = 0 \tag{6}$$

$$\alpha\beta x_{trim} = [V, \alpha, \beta, p, q, r, \phi, \theta]^T = x^* \tag{7}$$

$$u_{trim} = [\delta_{th}, \delta_e, \delta_a, \delta_r]^T = u^* \tag{8}$$

$$\left(\gamma = \gamma^*\right), \left(\dot{\psi} = \dot{\psi}^*\right), (\dot{\phi}, \dot{\theta} = 0) \tag{9}$$

where f is a vector of nonlinear functions, x is the state vector, u is the control vector, and δ_{th}, δ_e, δ_a, and δ_r represent the engine throttle setting and the deflections in the elevator, aileron, and rudder respectively. Trim condition must satisfy the aircraft equations of motion, as in (6). In (9), γ^*, $\dot{\psi}^*$ are the desired constant values which define the steady state maneuvers presented in (2)−(5). Hence, for each desired steady state maneuver, the trim vectors (x^*, u^*) are found by solving all the aircraft nonlinear equations of motion $(\dot{x}_{trim} = 0)$ simultaneously for the desired flight path angle and turn rate $(\gamma^*, \dot{\psi}^*)$ at a specific total airspeed V^* and a specific altitude h^*. Therefore, MFEs are comprised of trim states characterized by four parameters $(h^*, V^*, \gamma^*, \dot{\psi}^*)$, and the state and control vectors of each trim state are shown as:

$$x^*\left(h^*, V^*, \gamma^*, \dot{\psi}^*\right), u^*(h^*, V^*, \gamma^*, \dot{\psi}^*) \tag{10}$$

(10) shows that MFEs are actually four-dimensional, however, we can show these flight envelopes as 3D volumes at each constant flight altitude h^*, as in Fig. 1 and Fig. 2.

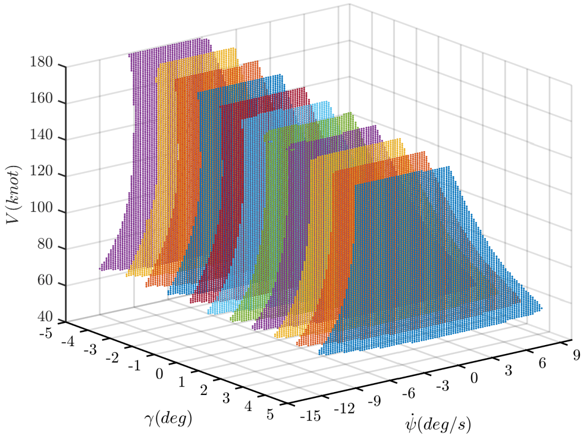


FIGURE 1. Rudder jammed at 10° at sea level.

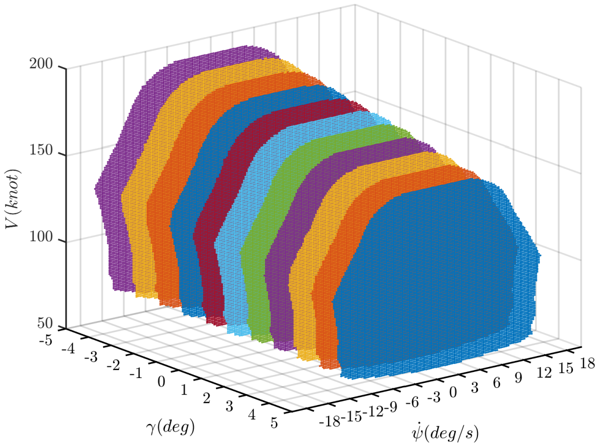


FIGURE 2. Unimpaired case at sea level.

Also by fixing flight path angle γ^*, 3D flight envelopes become 2D closed curves, as in Fig. 3.

V_{min} and V_{max} are two key characteristics of aircraft maneuverability. They are specified as the lowest and the highest speed an aircraft can acquire within its 2D $(V - \dot{\psi})$ maneuvering flight envelope. These two parameters vary with

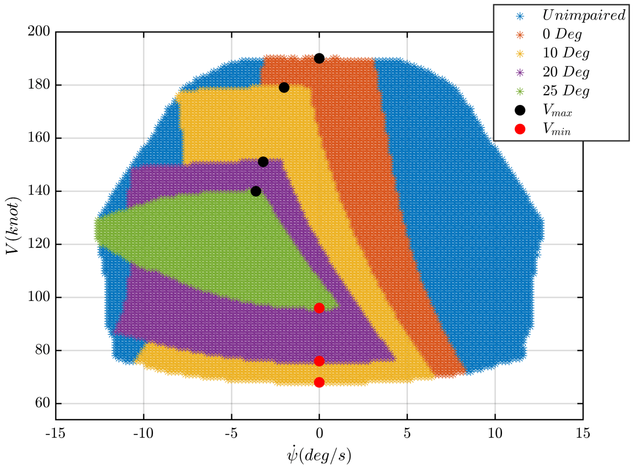


FIGURE 3. Rudder jamming cases at 10000 ft and $\gamma = -5°$.

different failure degrees. For instance, as it can be seen in Fig. 3, the minimum and maximum speeds of an impaired aircraft with jammed rudder at 0 degrees is the same as an unimpaired aircraft. However, as the failure gets severe, the flight envelope shrinks and the values of V_{min} and V_{max} change. Bigger jammed rudder angle means the aircraft is more sideslipped ($\beta \neq 0$) which yields in more persistent extra drag force. Such drag increases the required thrust, which in turn leads to lower maximum speed. Also at bigger jammed rudder angles, the lateral-directional moments that should be counteracted with aileron deflection are larger in magnitude and hence require more aileron effectiveness which is available only at higher speeds. Therefore the minimum speed is higher at bigger failure degrees. It is also shown in Fig. 1 that V_{max} decreases with the increase in flight path angle γ^*. Also V_{min} and V_{max} vary at different flying altitudes which is due to the change in air density and engines' available thrust.

Hence the values of V_{min} and V_{max} are dependent on the values of altitude h^*, flight path angle γ^*, and control surface's failure degree (i.e. rudder failure degree (δ_r) in this research). However, this dependency is derived from the nonlinear dynamics of the aircraft and cannot be evaluated straightforward. For instance, even though the minimum speed of an unimpaired aircraft can be calculated analytically from the stall speed equation, in the case of an impaired aircraft; it should be estimated via point by point evaluation of the trim points until the lowest feasible maneuver is found. The situation is the same to find the maximum speed of an impaired aircraft. This process which involves the evaluation of lots of trim points may take up to few hours and cannot be accomplished in real-time onboard the aircraft.

Instead in this research, ANNs and LMNs are trained with data comprising the values of V_{min} and V_{max} at various rudder failure degrees and different flight conditions. The trained networks estimate the values of V_{min} and V_{max} at unpredicted failure degrees or flight conditions, instantaneously.

A. INPUT DATA

As mentioned earlier, each trim point is characterized by the four parameters of (h^*, V^*, γ^*, $\dot{\psi}^*$). So varying altitude and flight path angle results in different 2D maneuvering flight envelopes and hence different V_{min} and V_{max}. Therefore flight altitude h^* and flight path angle γ^* are considered as the inputs of the network. The other inputs are determined based on the failure category. There are four categories of control surface failures [24]:

- Control restriction, in which, upper and/or lower limits of deflection are changed to new, equal or non-equal values
- Surface jam
- Reduced rate limits, in which, upper and/or lower rate limits are changed
- Surface runaway, which at first shows up as reduced rate limits, but eventually changes to surface jam case

The fourth category eventually becomes surface jam; hence there are three main failure categories which we have considered two more common of them in this research: control restriction and surface jam. These two categories are caused by physical damage, icing, or loss of hydraulic power.

When a control surface is jammed, the corresponding control parameter of the jammed surface is set to the jammed deflection angle in the aircraft equations of motion (e.g. $\delta_r = 20°$) or in the other words, the lower limit and upper limit of the surface deflection angle are changed to the same value (e.g. $20° \leq \delta_r \leq 20°$). Also in the case of restricted deflection angle, the constraint limits on the control parameter are changed to the new lower and upper limit values of the surface deflection angle (e.g. $-10° < \delta_r < 15°$). So in case of actuator failure, the lower and upper limit values of the surface deflection angle are considered as the inputs of the network.

According to the aforementioned explanations, the input parameters for both ANNs and LMNs are shown in Table 1:

TABLE 1. Inputs of the network.

Failure Type	Network Inputs
Rudder failure	h (flight altitude)
	γ (flight path angle)
	$\delta_r min$ (lower limit of rudder deflection)
	$\delta_r max$ (upper limit of rudder deflection)

B. OUTPUT DATA

Since the networks are designed to estimate the values of V_{min} and V_{max}, these two parameters are the outputs of the networks. In the case of ANN, both parameters are included in the output layer of one network as neural networks can have multiple outputs. However in the case of LMN, each output parameter is evaluated with one network.

Hence, in order to be able to compare the performance of multi-output ANN and LMN, the same loss function used in multi-output ANN is applied to the outputs of the two LMNs estimating V_{min} and V_{max}:

$$L = \frac{1}{D}\sum_{i=1}^{D}\frac{1}{O}\sum_{j=1}^{O}\left(\hat{y}_{ij} - y_{ij}\right)^2 \tag{11}$$

where the loss function L is the mean squared error (MSE) of the multidimensional outputs, D is the number of data samples, O is the output dimension, $\hat{y}_i$ is the target value of a specific sample and y_i is the network output.

C. NEURAL NETWORK ARCHITECTURE

Multilayer feedforward neural networks are often used to learn the nonlinear relationship between input and output vectors. In this research, the aim is to design a network which is capable of approximating the high-fidelity nonlinear equations of motion of aircraft with good accuracy. A two-layer network including one hidden layer with nonlinear sigmoid transfer function and one output layer with linear transfer function is most often used for function fitting (nonlinear regression), and it has been proven that given sufficient neurons in the hidden layer, the network can approximate any nonlinear function. This two-layer architecture is shown in Fig. 4 and is used in this study.

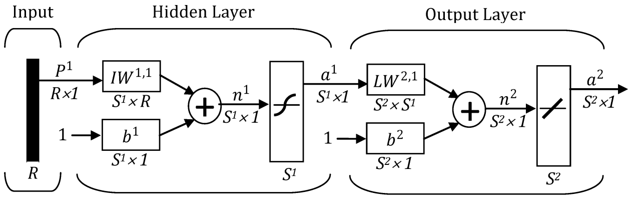


FIGURE 4. **Two-layer feedforward neural network architecture.**

In the presented architecture, R and S indicate the number of parameters in the input layer and the number of neurons in the corresponding layer, respectively. Also, a^1, and a^2 which represent the outputs of the tan-sigmoid transfer function and the linear transfer function are defined as below [25]:

$$a^1 = tansig\left(IW^{1,1}p^1 + b^1\right) == \frac{2}{1 + e^{-2\left(IW^{1,1}p^1+b^1\right)}} - 1 \tag{12}$$

$$a^2 = purelin\left(LW^{2,1}a^1 + b^2\right) = LW^{2,1}a^1 + b^2 \tag{13}$$

In this research, $S^2 = 2$ which is the number of the output parameters (i.e. values of V_{min} and V_{max}), also the dimension of R is 4 according to Table 1. Generally, the number of neurons in the hidden layer must be determined by trial and error; however, in the case studies of this research it was found that for the single actuator failures 10 hidden neurons are sufficient (see Fig. 41 in the Appendix).

D. NEURAL NETWORK TRAINING ALGORITHM

Typically the performance function used to train feedforward neural networks is chosen to be the sum of squares of the network errors on the training set:

$$F = mse = \frac{1}{n}\sum\nolimits_{i=1}^{n}(e_i)^2 = \frac{1}{n}\sum\nolimits_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{14}$$

The training process of a neural network involves optimizing the performance function by updating the network weights and biases at every iteration. This can be accomplished using a variety of training algorithms. For networks with up to a few hundred weights, the Levenberg-Marquardt is often the fastest backpropagation algorithm and will have the fastest convergence; it is also very accurate. However, for larger networks or highly nonlinear relationships between the inputs and outputs, Levenberg-Marquardt performance is relatively poor. In such cases, the trained network's generalization is weak and it overfits the data. One way to overcome this issue is to provide as much training data as possible. However, training data generation for flight envelope estimation is computationally extensive and very time-consuming.

Another way to improve network generalization and avoid overfitting in the case of small dataset is to use Bayesian Regularization training algorithm. In this method, regularization modifies the performance function by adding a term consisting of the mean of the sum of squares of the network weights and biases [26]:

$$F\left(w\right) = \zeta\left(\frac{1}{N}\sum_{j=1}^{N}w_j^2\right) + \eta\left(mse\right) = \zeta\left(msw\right) + \eta\left(mse\right) \tag{15}$$

where ζ and η are the performance ratio parameters. This modified performance function yields in smaller network weights and biases and a smoother network response. At each iteration, the Bayesian regularization modifies the linear combination presented in (15) by updating the performance ratio parameters, until the optimal ratio parameters are found and the network has good generalization qualities. The Bayesian regularization takes place within the Levenberg-Marquardt algorithm, which means Levenberg-Marquardt is used to minimize the modified performance function at each iteration, and the Gauss-Newton approximation to Hessian matrix (which is available in the Levenberg-Marquardt algorithm) is used to compute the effective number of parameters. In the Bayesian regularization algorithm, dataset is split into training data and test data as there is no stopping criterion based on validation data, thus the training continues until an optimal combination of weights and errors are found.

To find the optimum values of the performance ratio parameters, the Bayesian framework of MacKay [27] is used in the following steps [26]:

- ζ, η, and network weights are initialized.
- One step of the Levenberg-Marquardt algorithm is used to minimize the modified performance function.
- The effective number of parameters $[N_{eff} = N - 2\zeta tr(H)^{-1}]$ is calculated using the Gauss-Newton approximation to the Hessian:

$\mathrm{H} = \nabla^2 F(w) \approx 2\eta \mathbb{J}^T \mathbb{J} + 2\zeta I_N$, where N is the total number of network parameters, I is the identity matrix, and $\mathbb{J}$ is the Jacobian matrix computed through a standard backpropagation technique.

- New estimates of the performance ratio parameters are computed: $\zeta = \frac{N_{eff}}{2(msw)}$, and $\eta = \frac{n-N_{eff}}{2(mse)}$.
- The last three steps are iterated until convergence.

In this research, with the aim of comparison, both Bayesian regularization and Levenberg-Marquardt are used as the ANN training algorithms. Also, the multiple-neural-networks-training technique has been used in which each neural network is trained multiple times (20 times in this study). Each time; the training process starts from different initial weights and biases and with different divisions of data into training and test sets. These different conditions lead to different networks with different generalization qualities. It should be noted these different ANNs have the same network architecture (i.e. one hidden layer with 10 hidden neurons).

It should be noted that in the multiple-network technique, the test set of one network could be the training set of another network, so these test sets are not an independent measure of the networks' generalization. Therefore the original dataset is divided into two parts, one part that is used during the training process of each network and will be divided into training data, validation data (in the case of Levenberg-Marquardt algorithm), and test data, and the other part which is used as a completely independent test data. Finally, the network with the least error on the independent test data has the best generalization and will be selected – among the 20 networks; as the final network.

E. LOCAL MODEL NETWORK ARCHITECTURE

Local neuro-fuzzy models or a local model network is the alternative to ANN in the field of system identification. The basic idea of an LMN is that a nonlinear system can be approximated from measured data using several simple local models, instead of one intricate global model [16].

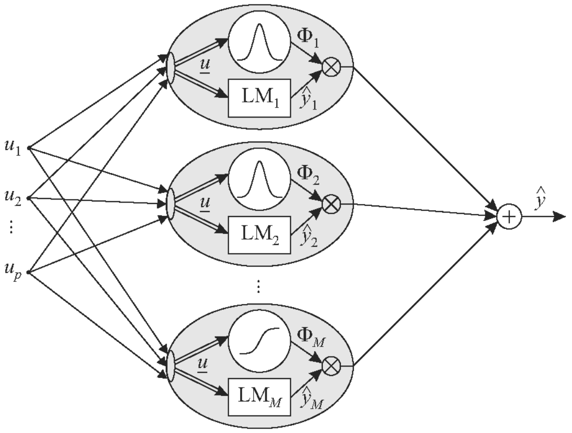


FIGURE 5. Local Model Network (LMN) structure [29].

A local model network structure is shown in Fig. 5. In this structure, each neuron M is comprised of two parts. A local model (LM_i) and a validity function (ϕ_i). The output $\hat{y}_i$ of each local model is weighted by its validity function [28]. Hence the output $\hat{y}$ of a local model network with p inputs $[u_1, u_2, \ldots, u_p]$ is the aggregation of M local model outputs$\hat{y}_i$.

$$\hat{y} = \sum\nolimits_{i=1}^{M} \hat{y}_i\,(\bar{u})\phi_i(\bar{u}) \tag{16}$$

Also in local model networks, validity functions and local models can have different input spaces. Hence (16) can be rewritten as:

$$\hat{y} = \sum\nolimits_{i=1}^{M} \hat{y}_i\,(\bar{x})\phi_i(\bar{z}) \tag{17}$$

where $\bar{x}$ and $\bar{z}$ span the consequent and premise input spaces respectively [28]. To have a smooth transition between local models, validity functions are between 0 and 1. Also to have a reasonable interpretation of the LMN, the validity functions should form a *partition of unity* $\sum_{i=1}^{M} \phi_i(\bar{u}) = 1$ [30].

The most common choice for a local model is a polynomial which can be of any order. However, as the order of the polynomial increases, the number of required local models for a specific accuracy decreases, because the number of local model parameters (i.e. local model complexity) increases. Therefore low order polynomials are good trade-offs [30].

In this research, three types of local models have been used to design the LMNs:

- *Local linear model* (LLM) in which the output is:

$$\hat{y}_i\,(\bar{u}) = w_{i,0} + w_{i,1}u_1 + w_{i,2}u_2 + \ldots + w_{i,p}u_p \tag{18}$$

where w_i is the i^{th} model parameters.

- *Full quadratic model* in which all cross-product terms are considered. This model becomes inefficient for high-dimensional input spaces due to the huge number of cross-terms.
- *Sparse quadratic model* which is a quadratic model without cross-terms such as u_1u_2.

F. LOCAL MODEL NETWORK TRAINING ALGORITHM

In this research, designed LMNs are trained using LOLIMOT and HILOMOT algorithms. LOLIMOT is an incremental tree-structured algorithm that starts with a single model (i.e. one neuron network) and refines its performance in each iteration by splitting the input space in an axis-orthogonal manner. In each iteration one new model (neuron) is added to the network because the worst local model is split into two equal halves. Hence unlike the ANN, the number of required neurons (local models) in an LMN is not pre-determined by the user but will be determined at the end of the iterations once the training process of the LMN is finished.

In this algorithm, the models' parameters are estimated by local least squares approach, and the validity functions are normalized orthogonal Gaussians [31]:

$$\phi_i\,(\bar{u}) = {}^{\mu_i(\bar{u})}\Big/_{\sum_{j=1}^{M} \mu_j(\bar{u})} \tag{19}$$

where

$$\mu_i(\bar{u}) = exp(-\frac{1}{2}\left(\frac{(u_1 - c_{i1})^2}{\sigma_{i1}^2} + \ldots + \frac{(u_p - c_{ip})^2}{\sigma_{ip}^2}\right)) \tag{20}$$

It can be seen that the validity function ϕ_i is dependent on the center c_{ij} and standard deviation σ_{ij} of the rectangles generated during partitioning. These parameters are similar to the hidden layer parameters of ANN [31].

When the partitioning is done in an axis-oblique manner, the algorithm is called HILOMOT (HIerarchical LOcal MOdel Tree) [28], [30]. By using HILOMOT, the limitations of axis-orthogonal splits are lifted. Also flat model structure is transformed into hierarchical model structure. In this algorithm, axis-oblique partitioning is done using sigmoid splitting functions [30].

Both LOLIMOT and HILOMOT are used to train all three types of local models mentioned in the previous subsection. Also, the same multiple-networks-training technique used in the ANN training is used in the training of LMNs too. So the LMNs training process can be summarized as below:

- Three local model networks are generated using the three local model types.
- Each network is trained using both LOLIMOT and HILOMOT. Hence a total of 6 networks are trained.
- Since the termination criterion for both algorithms is the performance on the validation dataset, among the 6 trained networks; the one with the lowest error on the validation dataset is selected.
- The above three steps are iterated 20 times (i.e. 20 rounds of training) and 20 LMNs are selected. The LMN with the least error on the independent test data has the best generalization and will be selected – among the 20 LMNs; as the final network.

III. VALIDATION OF THE DYNAMICS MODELING METHOD

In order to have a valid comparison between the generalization capabilities of the trained ANNs and LMNs, the estimated values of V_{min} and V_{max} must be compared with those evaluated via solving the aircraft nonlinear equations of motion.

A. DYNAMIC MODEL

A key element in every scientific research is a valid model. The Generic Transport Model (GTM) – with tail number T2, is a 5.5% twin – turbine powered, dynamically scaled aircraft which is designed with the aim of flying into drastic upset conditions and being safely recovered. A schematic of GTM-T2 along with its control surfaces is shown in Fig. 6. The GTM-T2 properties are shown in Table 2.

As part of NASA AvSP (Aviation safety program)'s Integrated Resilient Aircraft Control (IRAC) Project, extensive wind tunnel tests were performed on the GTM to create extended-envelope aerodynamic data set. Test data were obtained at angles of attack as low as −5° and up to +85° and sideslip angles ranging from −45° to +45° [32].

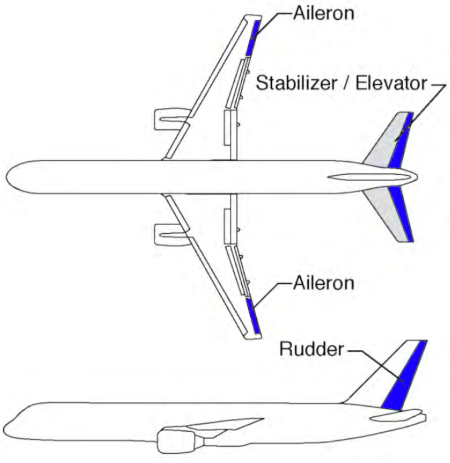


FIGURE 6. Schematic of 5.5% subscale GTM-T2 (based on [32]).

TABLE 2. GTM-T2 properties.

Property	Quantity
Takeoff weight, W_0	257 N (26.2 kg)
Wing area, S	0.5483 m^2
Wing span, b	2.09 m
Length, l	2.59 m
Mean aerodynamic chord, $\bar{c}$	0.2790 m

The dynamic model used in this research is the GTM-T2, high fidelity, nonlinear, 6 DOF, MATLAB® – Simulink® model, also known as *"GTM-DesignSim"* [33]. The model utilizes extensive wind tunnel test data in tabular form as the required aerodynamic database. The model's Simulink® environment is shown in Fig. 7.

B. TRAINING DATA GENERATION

As mentioned in section II, V_{min} and V_{max} (i.e. outputs of the networks) are the lowest and highest speed of an aircraft within its 2D ($V - \dot{\psi}$) maneuvering flight envelope. Hence in order to generate the required training dataset, maneuvering flight envelopes of the impaired GTM must be evaluated at different failure degrees and various flight conditions.

This subsection briefly describes the process in which maneuvering flight envelopes are estimated for the purpose of training data collection. For more details refer to [4] and [34].

Flight envelopes estimated in this research are actually maneuvering flight envelopes, which mean they are boundaries containing steady state maneuvers (i.e. trim points). It was mentioned that the trim vectors (x^*, u^*) are found by solving $\dot{x}_{trim} = 0$ for the desired trim parameters $(h^*, V^*, \gamma^*, \dot{\psi}^*)$. Solving ($\dot{x}_{trim} = 0$) is not analytically possible due to nonlinearities of the equations of motion and their complexity. However, we can derive trim points by numerically solving the corresponding constrained nonlinear optimization problem, in which, the cost function

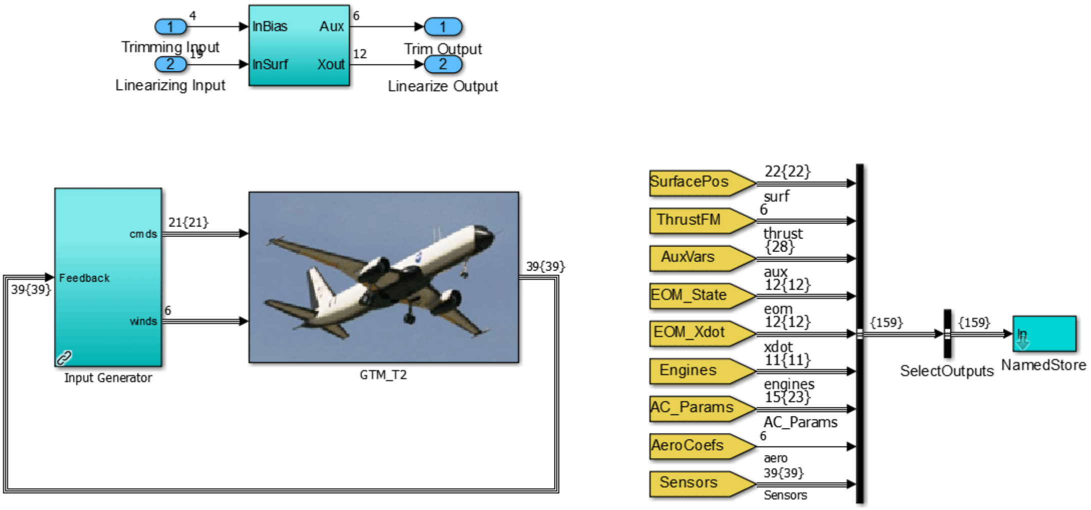


FIGURE 7. GTM-T2 Simulink® environment.

is J [4], [23]:

$$J\,(x,u) = {}^1\!/_2\,\dot{x}_{trim}^T Q \dot{x}_{trim} \tag{21}$$

where Q is a positive definite weighting matrix which specifies the state derivatives contributions to the cost function J. Cost function J is subject to a number of equality and inequality constraints. Quadruplet $(h^*, V^*, \gamma^*, \dot{\psi}^*)$, directly defines the following equality constraints:

$$h - h^* = 0 \tag{22}$$

$$V - V^* = 0 \tag{23}$$

(22) and (23) constrain the altitude and airspeed at which we are trying to find trim points.

$$tan\theta - \frac{ab + sin\gamma^* \sqrt{a^2 - sin^2\gamma^* + b^2}}{a^2 - sin^2\gamma^*} = 0, \quad \theta \neq \pm\pi/2 \tag{24}$$

where,

$$a = cos\alpha cos\beta,\, b = sin\phi sin\beta + cos\phi sin\alpha cos\beta \tag{25}$$

Equation (24) defines the rate-of-climb constraint which designates the desired flight path angle. More detail on the derivation of (24) can be found in [23].

Substituting $\dot{\psi} = \dot{\psi}^*$ and $\dot{\phi} = \dot{\theta} = 0$ in the aircraft rotational kinematic equations yields:

$$p + \dot{\psi}^* sin\theta = 0 \tag{26}$$

$$q - \dot{\psi}^* cos\theta sin\phi = 0 \tag{27}$$

$$r - \dot{\psi}^* cos\theta cos\phi = 0 \tag{28}$$

Equations (22), (23), (24), (26), (27), and (28), form the required equality constraints on the cost function J. On the other hand, inequality constraints are dictated by physical limits on the control inputs, that being:

$$|\delta_{th} - 0.5| \le 0.5\, |\delta_e| \le 30$$

$$|\delta_a| \le 20\, |\delta_r| \le 30 \tag{29}$$

To obtain the feasible trim points, the constrained optimization problem defined by J and the sets of equality and inequality constraints is solved via sequential quadratic programming (SQP) technique through the *trimgtm* function of the *GTM-DesignSim*, with a convergence criterion of 10^{-7}.

Being feasible is not enough for a trim state to include it inside the boundaries of the maneuvering flight envelope. Feasibility is the necessary condition whilst stability is the sufficient condition for inclusion in flight envelope. A nonlinear system is considered stable at a specific trim point if the system inherently converges to the trim state when being in the vicinity of the trim point.

Hence the stability of the aircraft at each feasible trim point is evaluated by linearizing the equations of motion about x^* via perturbation method:

$$\dot{X} = AX + BU \tag{30}$$

where $X = x - x^*$, $U = u - u^*$, and A, B are constant Jacobian matrices:

$$A = {}^{\partial f}\!/_{\partial x}\Big]_{x^*,u^*} \tag{31}$$

$$B = {}^{\partial f}\!/_{\partial u}\Big]_{x^*,u^*} \tag{32}$$

Analytical derivation of matrices A, B is complicated due to tabular form of aerodynamic and propulsion data [4]. Hence, numerical approximations obtained through a set of first-order differences are used [4], [22], [23]:

$$A \approx {}^{\left(f\,(x^* + \varepsilon e_i,\, u^*) - f\,(x^*, u^*)\right)}\!/_{\varepsilon} \tag{33}$$

$$B \approx {}^{\left(f\,(x^*, u^* + \varepsilon e_i) - f\,(x^*, u^*)\right)}\!/_{\varepsilon} \tag{34}$$

in which, ε is a positive and small number (10^{-6} in this research), and e_i is the i^{th} column of an n-dimensional identity matrix; with n being the size of the trim state vector x^*.

A trim state is considered stable if matrix A has no positive real eigenvalues or complex eigenvalues with positive real

- Minimize cost function J in (21) subject to
    - Equality constraints in (22), (23), (24), (26), (27), (28)
    - Inequality constraints in (29)
    - Additional constraints
        - $-5° \leq \alpha \leq 10.5°$
        - $-30° \leq \phi \leq 30°$
- If $J \leq 10^{-7} \Longrightarrow$ triplet $(V^*, \gamma^*, \dot{\psi}^*)$ is feasible, then
    - Linearize system by (30), (33), (34)
        - Check eigenvalues of A
            - If stable $\Longrightarrow$ save triplet $(V^*, \gamma^*, \dot{\psi}^*)$ as acceptable maneuver
            - If unstable, check controllability using (35)
                - If controllable $\Longrightarrow$ save triplet $(V^*, \gamma^*, \dot{\psi}^*)$ as acceptable maneuver
                - If uncontrollable $\Longrightarrow$ ignore triplet $(V^*, \gamma^*, \dot{\psi}^*)$ and move to next triplet
- If $J > 10^{-7} \Longrightarrow$ triplet $(V^*, \gamma^*, \dot{\psi}^*)$ is infeasible, ignore it and move to next triplet

FIGURE 8. **Trim point investigation process.**

part. Stable trim points are more preferable, because the aircraft naturally tends to damp the effect of small disturbances around them, while at unstable trim points; aircraft diverges away from the trim state.

However, if the feasible trim state is unstable, it still can be accepted as part of the flight envelope if it is controllable, i.e. if the linear perturbation system about this trim state has a full rank controllability matrix C:

$$C = [B \ AB \ A^2B \cdots A^{n-1}B] \tag{35}$$

That is because when the system is controllable, the closed-loop eigenvalues of the linearized system can be assigned arbitrarily using a linear controller. Hence a controllable trim state can be maintained despite disturbances given a capable control law [22].

To evaluate the required 3D maneuvering flight envelopes, the explained numerical procedure is implemented iteratively for each triplet $(V^*, \gamma^*, \dot{\psi}^*)$, as in Fig. 8.

C. TRAINING DATA PREPARATION

In order to have a broad training dataset, rudder failures have been chosen from lowest to highest degrees such that they cover different sections of rudder operational range. Table 3 presents failure cases considered in the training dataset. Values in bracket are the lower limit (LL) and the upper limit (UL) of the rudder deflection, as in $[LL, UL]$. In fact, jamming failure is a special case of restriction failure, in which LL and UL are identical. It should be noted that in the derivation process of the failure trim points, the limits in the inequality constraints (29) are changed to the LL and UL of the failures.

3D MFEs of the unimpaired and impaired GTM have been evaluated at four different altitudes of Sea Level, 10000 ft, 20000 ft, and 30000 ft. Taking into account the unimpaired case and all considered rudder failure cases at the mentioned flying altitudes, 3D MFEs have been evaluated for 112 different cases (Table 4). All evaluated 3D MFEs are presented in [35].

TABLE 3. **Training data rudder failures.**

Failure Type	Failure degree
Surface Jam	$-30°, -20°, -10°, 0°, 10°, 20°, 30°$
Control Restriction	$[-30°, -20°], [-30°, -10°], [-30°, 0°], [-30°, 10°],$ $[-30°, 20°], [20°, 30°], [10°, 30°], [0°, 30°],$ $[-10°, 30°], [-20°, 30°], [-20°, 20°], [-20°, -10°],$ $[-20°, 0°], [-20°, 10°], [-10°, 0°], [-10°, 10°]$ $[10°, 20°], [0°, 20°], [-10°, 20°], [0°, 10°]$

TABLE 4. **Training data 3D maneuvering flight envelope.**

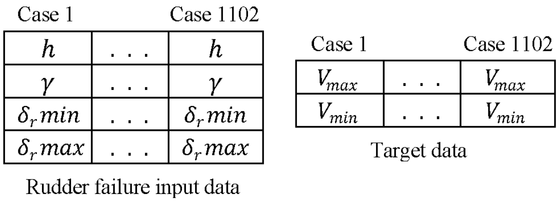
Failure Type	Control Surface/Altitude	Quantity		
Surface Jam	Rudder	7	28	112
	Altitude	4		
Control Restriction	Rudder	20	80	
	Altitude	4		
Unimpaired	Altitude	4	4	

TABLE 5. **Flight envelope increments.**

Parameter	Resolution Increment Size
V	1 knot
γ	1 deg
$\dot{\psi}$	0.2 deg/s

The smaller the resolution increments in V, γ, and $\dot{\psi}$ ranges, the more the trim points, and hence the higher the accuracy of the flight envelopes and their boundaries. Therefore we chose these increments as per Table 5 so that high fidelity flight envelopes could be estimated:

In this research, MFEs have been evaluated for different flight path angles within the range of $-5° \leq \gamma \leq 5°$.



Case 1		Case 1102
h	...	h
γ	...	γ
$\delta_r min$	...	$\delta_r min$
$\delta_r max$	...	$\delta_r max$

Rudder failure input data

Case 1		Case 1102
V_{max}	...	V_{max}
V_{min}	...	V_{min}

Target data

FIGURE 9. Training data structure.

Algorithm 1 Training Process of ANNs

```
Given a training dataset as in Fig. 9: $I = Inputs_{(4\times\mathcal{T})}$, $T = Targets_{(2\times\mathcal{T})}$
Load the training dataset and divide it into two parts:
90% for designing networks: $x_1$, $t_1$($x$ denotes inputs and $t$ denotes targets)
10% to be used as independent test data: $x_2$, $t_2$
Initialize two empty matrices: $net(1, j)$, $Test\_Performance(1,j)$
Set Training function = Bayesian Regularization
    Number of hidden layers = 1
    Number of hidden neurons = 10
    Performance function = Mean squared error
For all $j \in \{1, \dots, 20\}$, do
Divide $x_1$, $t_1$ randomly into training set (90%) and test set (10%)
Train $net(1, j)$ using the training set and test set of $x_1$, $t_1$
Evaluate performance of $net(1, j)$ with respect to $x_2$, $t_2$
Let $Test\_Performance(1, j)$ = performance of $net(l,j)$
End for
Return $net(l, j)$, $Test\_Performance(1, j)$
```

Thus, given the 1 *deg* resolution increment in the γ range, totally 1102 2D MFEs were estimated for the rudder failure. The maximum speed and minimum speed of these 2D MFEs are the targets of the designed networks.

As mentioned earlier, the training dataset is comprised of two sections. The first section consists of input data in the form of a ($4 \times \mathcal{T}$) matrix where $\mathcal{T}$ represents the number of training samples and the second section consists of target data in the form of a ($2 \times \mathcal{T}$) matrix. The following figure clarifies the structure of the training data.

The above pseudocode presents the training process of ANN. As explained in subsection II.D, the training dataset is divided into two parts, one to design the network which itself is randomly divided into training set and test set, and the other to be used as independent test data. The ANN is trained 20 times with different training set and test set and evaluated each time with respect to the independent test data. Eventually, the network with the least mean squared error (MSE) on the independent test data is selected as the final network.

Similarly, as explained in subsection II.F, for the training process of LMN, three local model types are initialized and trained with LOLIMOT and HILOMOT algorithms using the first part of the training dataset which is divided into training set and validation set. This results in 6 trained networks (i.e. 3 trained with LOLIMOT and 3 trained with HILOMOT), for which the one with lowest MSE on the validation set is selected. The process is iterated 20 times and the LMN with the best performance on the second part of training dataset (i.e. the independent test data) is selected as the final LMN.

Algorithm 2 Training Process of LMNs

```
Given a training dataset as in Fig. 9: $I = Inputs_{(4\times\mathcal{T})}$, $T = Targets_{(1\times\mathcal{T})}$
Load the training dataset and divide it into two parts:
90% for designing networks: $x_1$, $t_1$($x$ denotes inputs and $t$ denotes targets)
10% to be used as independent test data: $X_2$, $t_2$
Initialize three empty matrices:
$net(i, j)$, $Selected\ LMN(i)$, $Test\_Performance(i)$
For all $i \in \{1, \dots, 20\}$ do
Initialize three local model types: Linear, Full quadratic, Sparse quadratic
Set Training algorithm = LOLIMOTI HILOMOT
    Performance function = Mean squared error
Forall $j \in \{1, \dots, 6\}$ do
Divide $x_1$, $t_1$ randomly into training set and validation set
Train $net(i, j)$ using the training set and validation set of $x_1$, $t_1$
Evaluate performance of $net(i, j)$ with respect to validation set of $x_1$, $t_1$
Let $Val\_Performance(i, j)$= validation performance of $net(i, j)$
End for
$Selectnet(i, j)$ with lowest $Val\_Performance$
Let $Selected\ LMN(i)$=selected $net(i_1, j)$
Evaluate performance of $Selected\ LMN(i)$ with respect to $x_2$, $t_2$
Let $Test\_Performance$(i)=performance of $Selected\ LMN(i)$
End for
Return $Selected\ \ LMN$(i), $Test\_Performance(i)$
```

The above pseudocode presents the training process of LMN. It should be noted that since the output of the LMNs is one dimensional, each row of the target matrix is trained separately. Hence in the following pseudocode, the matrix T is in the form of ($1 \times \mathcal{T}$).

D. NUMERICAL RESULTS

This section provides the results of the networks' training along with the comparison between the networks' generalization. Fig. 10 depicts the performance of the 20 ANNs trained with the Bayesian regularization and Levenberg-Marquardt algorithms; on the independent test data. It can be seen that the ANNs trained with both algorithms have MSEs in the order of 1×10^{-4} which is an indication of good

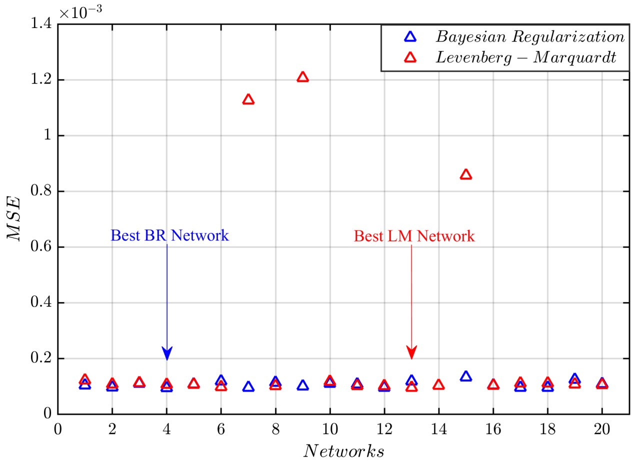


FIGURE 10. Bayesian regularization and Levenberg-Marquardt performance on the independent test data.

generalization and high accuracy of the selected networks. However in most ANNs, the Bayesian regularization outperforms Levenberg-Marquardt with a slight difference. In fact, Bayesian regularization is more robust than Levenberg-Marquardt with respect to the network parameters.

TABLE 6. Networks' MSEs on independent test data.

Network	Bayesian regularization MSE	Levenberg-Marquardt MSE
1	1.0308E-04	1.2198E-04
2	9.6487E-05	1.0679E-04
3	1.0916E-04	1.1201E-04
4	9.4251E-05	1.0618E-04
5	1.0656E-04	1.0565E-04
6	1.1727E-04	9.6980E-05
7	9.5053E-05	1.1267E-03
8	1.1308E-04	1.0072E-04
9	1.0002E-03	1.2066E-03
10	1.0892E-04	1.1610E-04
11	1.0753E-04	1.0104E-04
12	9.5544E-05	1.0146E-04
13	1.1746E-04	9.4897E-05
14	1.0243E-04	1.0201E-04
15	1.3282E-04	8.5716E-04
16	1.0348E-04	1.0249E-04
17	9.5586E-05	1.1126E-04
18	9.5294E-05	1.1108E-04
19	1.2433E-04	1.0744E-04
20	1.0869E-04	1.0502E-04

Table 6 presents the 20 trained ANNs along with their MSEs on the independent test data for each of the two training algorithms. The best network of each of the two training algorithms (i.e. the network with the least error on independent test data) is distinguished in Table 4 with green color.

It is shown in Table 6 and Fig. 10 that for the 7th, 9th, and 15th networks, the Levenberg-Marquardt algorithm has resulted in relatively larger MSEs. Similar behavior occurred even after repeating the whole 20 networks training process. This behavior is due to the stopping criterion of the Levenberg-Marquardt-algorithm based on the validation data. Each of the 20 training processes is initialized with different network parameters; however, the early stopping has prevented the aforementioned three networks from reaching a combination of network parameters with better performance. On the hand, since there is no such stopping criterion in the Bayesian regularization algorithm, the ANNs trained with this method have all reached an optimum value for the network parameters.

Generally, ANN's test performance is not as good as the training performance, so it is expected for the test MSE to be higher than the training MSE. However, the negligible differences between the training performances of the best ANNs and their performances on the independent test data reveal their good fit and generalization.

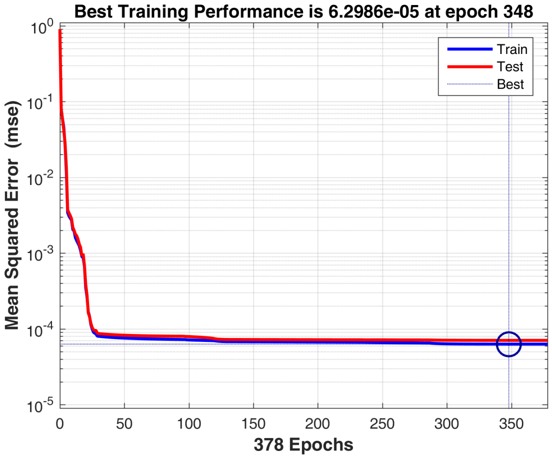


FIGURE 11. Training performance of the Bayesian regularization's final network.

Figures 11, 12 and 13 present the training performance, regression plots, and error histogram of the best ANN trained with the Bayesian regularization algorithm (i.e. the 4th network) whereas figures 14, 15, and 16 depict the performance of the best Levenberg-Marquardt trained ANN (i.e. the 13th network).

The two top and the lower left regression plots of figures 12 and 15 belong to the first part of the dataset (i.e. the part used to design network) whereas the lower right plot corresponds to linear regression on the independent test data (i.e. the second part of the dataset). It can be seen that for both networks, the R-value which is an indication of the relationship between the network outputs and targets; is very close to 1 and shows very good fit on both parts of the training dataset.

It should be noted that before training the networks, the target values of the training dataset are normalized between 0 and 1 (as seen in the regression plots).

The comparison between the generalizations of ANNs and LMNs is shown in Fig. 17. By comparing the black colored data, it can be seen that LMNs have also good accuracy, however in none of the 20 trained networks they can generalize to the independent test data as well as the ANNs.

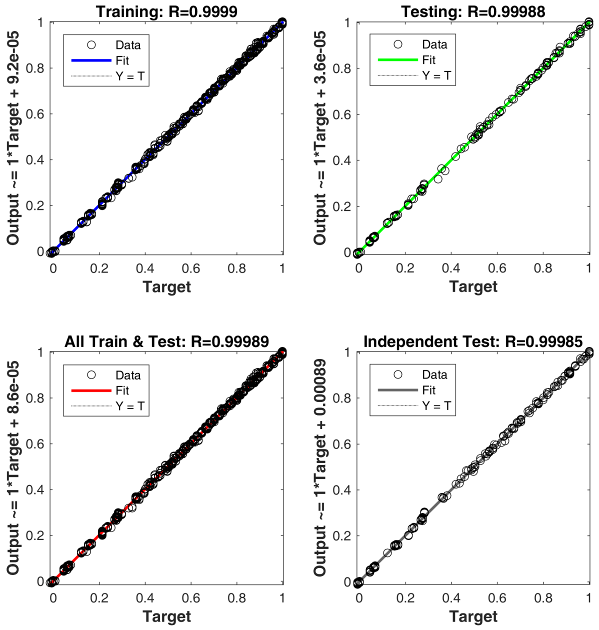


FIGURE 12. Linear regression plots of the Bayesian regularization's final network.

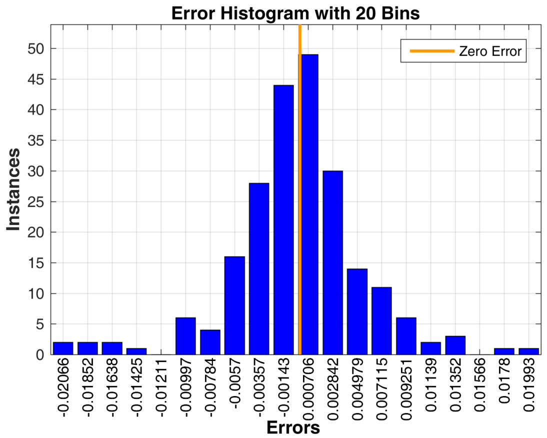


FIGURE 13. Bayesian regularization's final network's histogram of error on evaluating the independent test data.

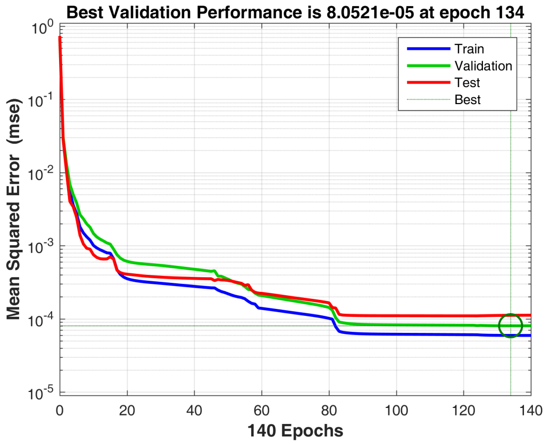


FIGURE 14. Training performance of the Levenberg-Marquardt's final network.

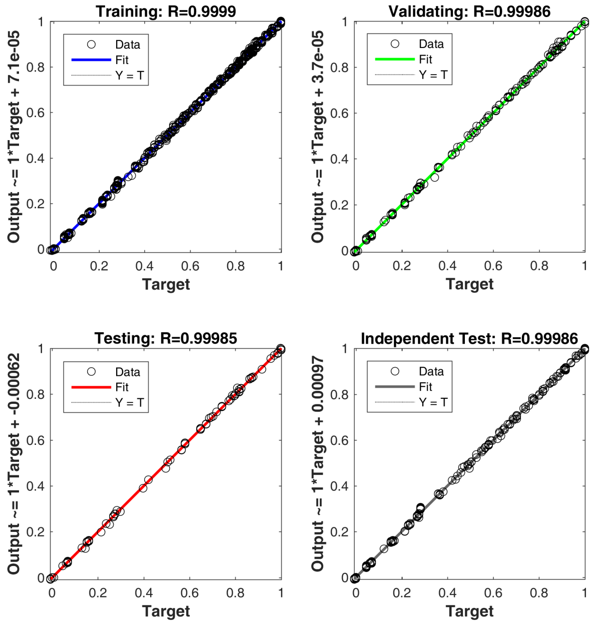


FIGURE 15. Linear regression plots of the Levenberg-Marquardt's final network.

Since ANNs' outputs are 2-dimensional, the black colored MSEs of the ANNs are in fact the networks' averaged errors of their outputs (i.e. V_{min} and V_{max}). Also as mentioned earlier, since LMNs' outputs are 1-dimensional, each of the 20 LMNs' final MSEs (i.e. black colored data) has been calculated by Eq. (11) using the errors of two LMNs calculating V_{min} and V_{max}.

For the LMNs estimating V_{max}, the best network among the 6 trained networks of each of the 20 iterations is the one trained with HILOMOT, whereas in the case of V_{min}, the best networks of the 20 iterations are either trained with LOLIMOT or HILOMOT.

Also, it can be seen that the error in estimating V_{min} is approximately in the same range for both the ANNs and LMNs. However, higher errors of estimating V_{max} in the LMNs have resulted in higher averaged MSEs. So LMNs cannot estimate V_{max} with the same accuracy of ANNs.

As shown in Fig. 18, the 18th network is the best LMN based on the averaged MSEs. Also Fig. 19 depicts the best ANN along with the errors of estimating V_{min} and V_{max} for all 20 ANNs.

Since the main source of difference between the generalizations of the LMNs and ANNs is in the estimation of V_{max}, the performance of the 18th LMN estimating V_{max} is shown in Fig. 20. Also figures 21 and 22 present all 6 LMNs trained at the 18th round of training. As mentioned earlier, at each round during the training process of each of the 6 LMNs, a local model (neuron) is added to the network at each iteration which consequently increases the number of network parameters and hence the model complexity. For each of the

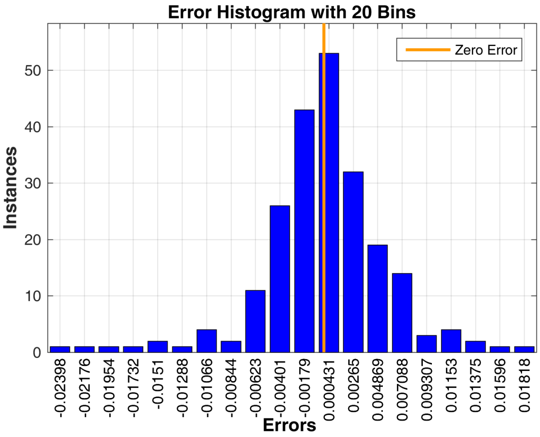


FIGURE 16. Levenberg-Marquardt's final network's histogram of error on evaluating the independent test data.

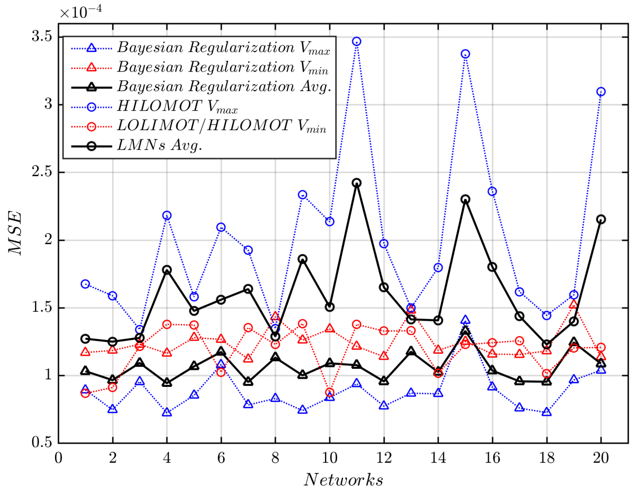


FIGURE 17. ANNs and LMNs performance on the independent test data.

6 networks, the training process is carried on and local models are added to the network during iterations until the performance of the network on validation data is not improved for two consecutive iterations. The best performance for each of the 6 LMNs is recorded and the network corresponding to the lowest achieved MSE is selected as the final LMN of the 18th round of training. In figures 21 and 22, the crossed network which is comprised of quadratic models trained by HILOMOT is the best and final LMN. This final LMN is the so called 18th network whose performance is shown in Fig. 20 and has 24 models (neurons) with 452 parameters. The other 5 LMNs are networks consisting of linear and sparse quadratic models trained by LOLIMOT and HILOMOT, and full quadratic models trained by LOLIMOT.

To understand why LMN has a higher error than ANN in the estimation of V_{max}, the best networks' squared errors of V_{max} in the 111 failure cases of the independent test data are shown in Fig. 23.

In other words, the V_{max} squared errors shown in Fig. 23 belong to the 4th ANN trained with the Bayesian regularization algorithm and the 18th LMN estimating V_{max}.

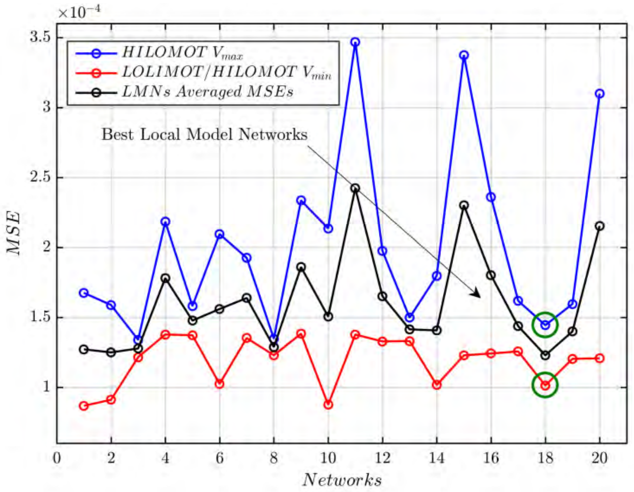


FIGURE 18. LMNs performance on the independent test data.

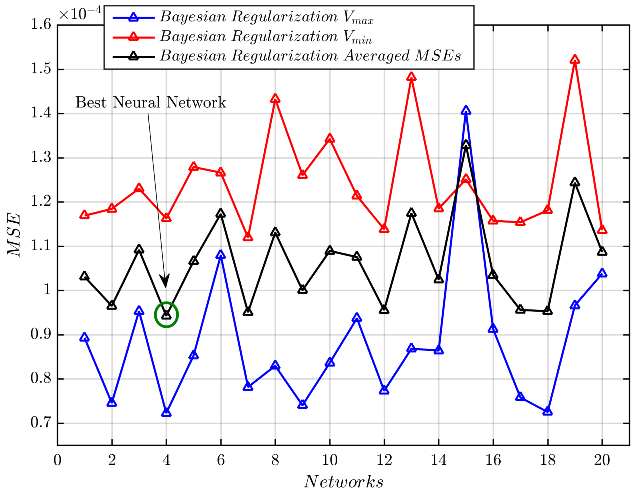


FIGURE 19. ANNs performance on the independent test data.

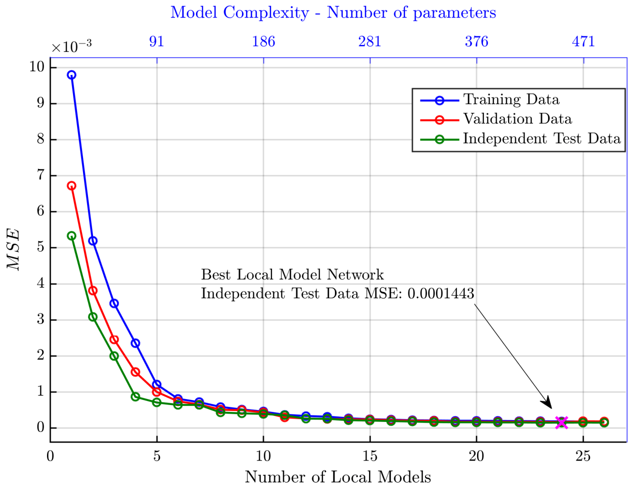


FIGURE 20. Best LMN's performance for estimating V_{max} (18th network).

As mentioned earlier, none of the 111 failure cases of the independent test data were used during the training processes of the ANNs and LMNs. As can be seen in Fig. 23, for most

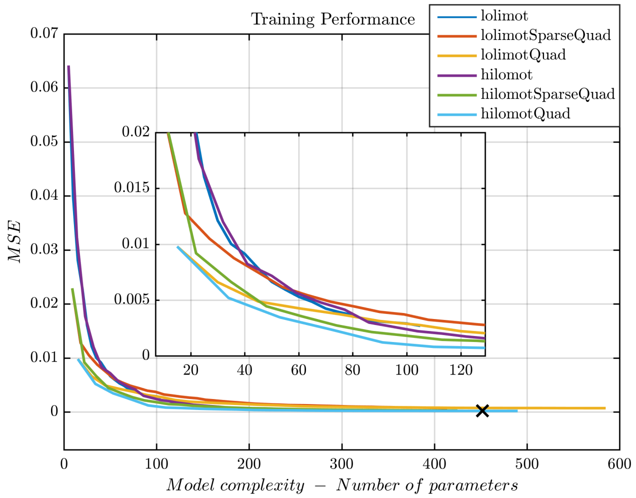


FIGURE 21. Training performance of the 6 LMNs trained at the 18th round of training.

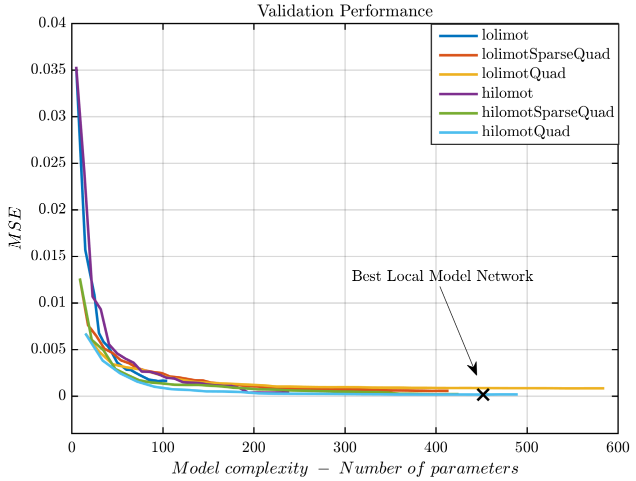


FIGURE 22. Validation performance of the 6 LMNs trained at the 18th round of training.

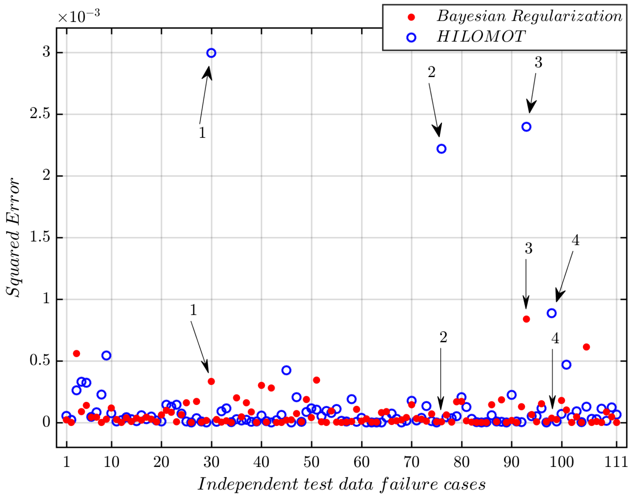


FIGURE 23. Squared errors of estimating V_{max}.

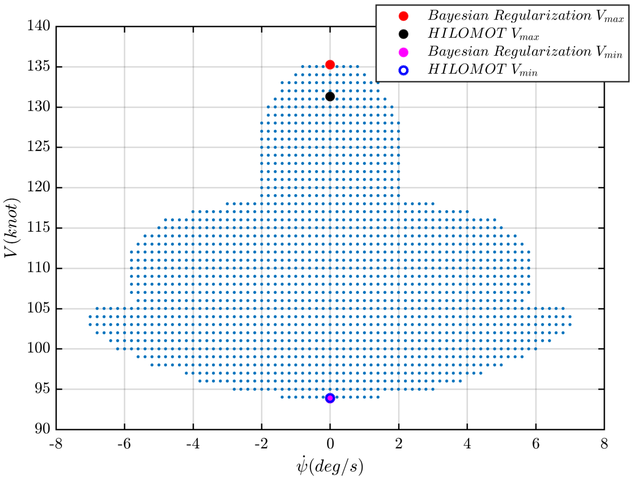


FIGURE 24. Rudder restricted to [−30°, +10°] at 30000 ft and $\gamma = 1°$.

of the failure cases of the independent test data, the best ANN and the best LMN have similarly very low errors. However, there are few cases in which the LMN has estimated V_{max} with larger errors than the ANN. Such outliers are the reason for higher MSE of the 18th LMN than the 4th ANN. Hence, even with good averaged performance, the generalization capability of LMN is not as good as the generalization of ANN.

Figures 24, 25, 26, and 27 show the values of V_{max} estimated by the best networks (i.e. the 4th Bayesian regularization ANN and the 18th LMN) for the outliers 1, 2, 3 and 4 of Fig. 23, respectively. Outlier 1 corresponds to a restricted rudder case whereas outliers 2, 3, and 4 correspond to three jammed rudder cases.

In these figures, blue dots represent the trim states evaluated via the iterative numerical procedure explained in the subsection III.B. In other word, the area covered by the blue dots is the corresponding 2D maneuvering flight envelope. Obviously, the exact values of V_{min} and V_{max} are the speeds of the lowest and the highest blue dots, respectively.

It can be seen in these figures that the LMN's estimation of V_{min} is close to those of the ANNs. However, the LMN's estimated V_{max} is considerably lower than the exact value, yielding in false elimination of many feasible trim states at the top of the flight envelope.

E. EFFECT OF DIFFERENT TRAINING DATASETS

Results of the previous subsection have been obtained from networks trained by a high-fidelity training dataset. As mentioned earlier, various flight path angles (γ), flying altitudes, and different failure degrees were chosen such that 1102 training samples were generated.

However, collecting training data is not an easy task. Specifically, in the case of modeling aircraft nonlinear dynamics, the process is very time-consuming and as

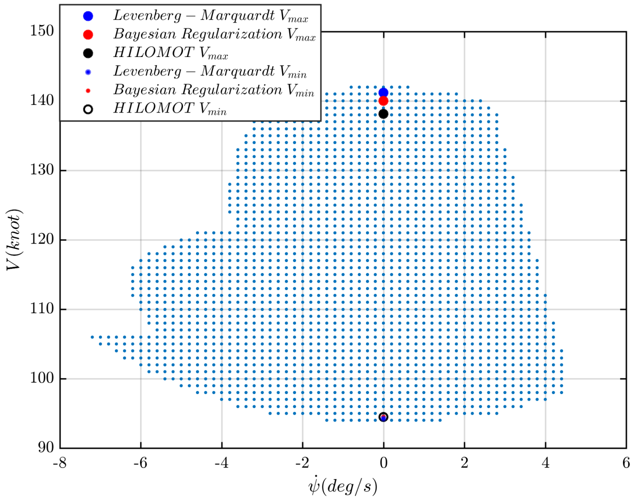


FIGURE 25. Rudder jammed at 10° at 30000 ft and $\gamma = 0°$.

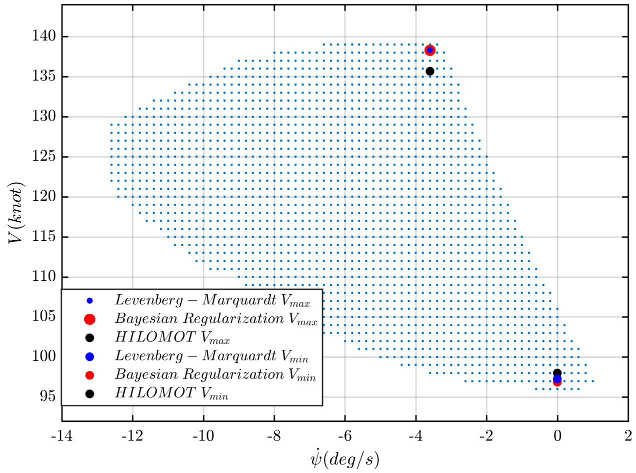


FIGURE 26. Rudder jammed at 25° at 10000 ft and $\gamma = -5°$.

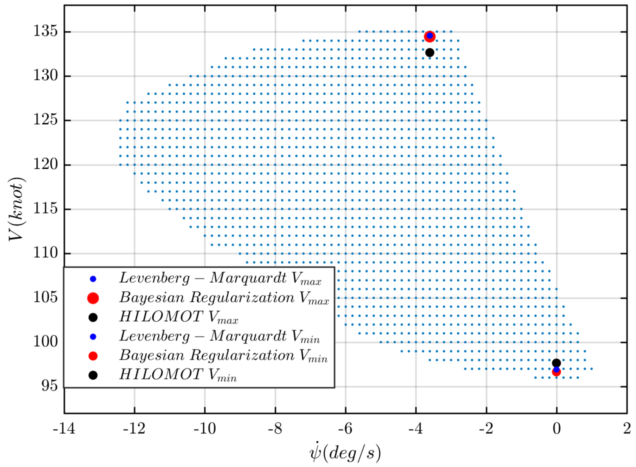


FIGURE 27. Rudder jammed at 25° at 10000 ft and $\gamma = -4°$.

TABLE 7. Input parameters' values of the original dataset.

Input Parameter	Network Inputs
h	[Sea level, 10000, 20000, 30000] ft
γ	[−5, −4, −3, −2, −1, 0, 1, 2, 3, 4, 5] deg
$\delta_r min$	[−30, −20, −10, 0, 10, 20, 30] deg
$\delta_r max$	[−30, −20, −10, 0, 10, 20, 30] deg

TABLE 8. Input parameters' values of the first diminished dataset.

Input Parameter	Network Inputs
h	[Sea level, 10000, 20000, 30000] ft
γ	[−5, 0, 5] deg
$\delta_r min$	[−30, −20, −10, 0, 10, 20, 30] deg
$\delta_r max$	[−30, −20, −10, 0, 10, 20, 30] deg

mentioned earlier it might take hours to generate few samples of the training dataset. Therefore it is useful to check how much the number of required training data could be diminished without a significant decline in the networks' performance. It is expected that reducing the number of training samples would degrade the networks' generalization and performance on the independent test data. However, it is desirable to find a compromise between the required number of training data and the networks' generalization.

One way to reduce the size of the required training dataset is to check which network inputs have more dominant effect on the network's performance and generalization. Each network input (which is either an aircraft state or control or a function of them) has its own level of influence in the nonlinearities of the aircraft dynamics. Inputs with higher levels of influence have more effect on the network's performance and hence require more training samples. These are inputs with more nonlinear behavior that are harder to predict. On the other hand, inputs with lower influence can be considered with lesser training samples, which consequently lead to a smaller size of the required training dataset.

According to Table 1 and subsection III.C, the original training dataset used so far is in the form of a 4 × 1102 input data matrix and a 2 × 1102 target data matrix, constructed from 1102 2D maneuvering flight envelopes evaluated at the following values of the network's input parameters:

For instance, [$h = 10000$ ft, $\gamma = -4°$, $\delta_r min = -10°$, $\delta_r max = 20°$] is one of the 1102 training samples.

In this section, 5 diminished training datasets are constructed by reducing the number of input parameters' instances shown in Table 7. For the first dataset, only the beginning, the ending, and the middle values of the flight path angle are considered. This way the γ instances are reduced from 11 to 3 which consequently yields in a reduction from 1102 to 260 training samples. Table 8 presents the input parameters of first diminished dataset:

For the second dataset (Table 9), the lower and upper limits of the rudder deflection value are considered only at −20°, 0°, and 20°. This results in 261 total training samples.

Similarly, for the third dataset (Table 10), the lower and upper limits of the rudder deflection value are considered

TABLE 9. Input parameters' values of the second diminished dataset.

Input Parameter	Network Inputs
h	[Sea level, 10000, 20000, 30000] ft
γ	[−5, −4, −3, −2, −1, 0, 1, 2, 3, 4, 5] deg
$\delta_r min$	[−20, 0, 20] deg
$\delta_r max$	[−20, 0, 20] deg

TABLE 10. Input parameters' values of the third diminished dataset.

Input Parameter	Network Inputs
h	[Sea level, 10000, 20000, 30000] ft
γ	[−5, −4, −3, −2, −1, 0, 1, 2, 3, 4, 5] deg
$\delta_r min$	[−30, −10, 10, 30] deg
$\delta_r max$	[−30, −10, 10, 30] deg

TABLE 11. Input parameters' values of the fourth diminished dataset.

Input Parameter	Network Inputs
h	[Sea level, 10000, 20000, 30000] ft
γ	[−5, 0, 5] deg
$\delta_r min$	[−20, 0, 20] deg
$\delta_r max$	[−20, 0, 20] deg

only at $-30°$, $-10°$, $10°$, and $30°$. This results in 344 training samples.

The fourth and fifth training datasets are combinations of the three aforementioned datasets. The fourth dataset (Table 11) is constructed by considering the reduced parameters of the first and second diminished datasets. In other words, for the fourth training dataset, the flight path angle values of Table 8 and rudder deflection values of Table 9 are considered as below:

This yields in 71 training samples for the fourth diminished dataset.

TABLE 12. Input parameters' values of the fifth diminished dataset.

Input Parameter	Network Inputs
h	[Sea level, 10000, 20000, 30000] ft
γ	[−5, 0, 5] deg
$\delta_r min$	[−30, −10, 10, 30] deg
$\delta_r max$	[−30, −10, 10, 30] deg

Likewise, the fifth dataset which is the combination of the first and third diminished datasets has 88 training samples, as shown in Table 12.

Summary of the 5 diminished datasets is presented in Table 13:

As can be seen in Tables 8 to XII, in all of the diminished datasets, the considered flying altitudes are the same as the original training dataset. That is because the flying altitude has already been considered at only 4 instances distanced from each other by 10000 ft.

TABLE 13. Input parameters' values of the fifth diminished dataset.

Dataset	Reduced parameters	Training Samples
#1	γ	260
#2	δ_r	261
#3	δ_r	344
#4	γ, δ_r	71
#5	γ, δ_r	88

The training processes presented in the pseudocodes of the previous subsection are applied to the diminished training datasets to train ANNs and LMNs. Similar to the original training dataset, for each of the 5 diminished datasets the network with the lowest MSE on the independent test data is chosen as the best network of that dataset. Fig. 28 presents the best ANNs and LMNs performance on the independent test data for the 5 diminished datasets.

In this figure, the 6th dataset is the original training dataset and the values in parentheses indicate the number of training samples for each dataset. As expected, ANN performs better than LMN on all datasets. Also, it can be seen that the MSE is not linearly dependent on the number of training samples. That is because each dataset has been constructed by reducing the samples of the input parameters in its own specific way, and as mentioned earlier; the input parameters have different levels of influence in the nonlinearities of the aircraft dynamics. Hence different datasets have different natures in terms of the nonlinear relationship between the input parameters and the targets.

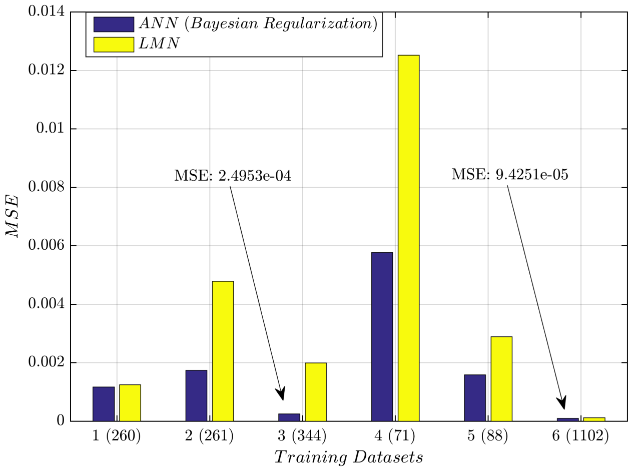


FIGURE 28. ANN and LMN performance on the independent test data for different training datasets.

To be more specific, it can be seen in Fig. 28 and Table 14 that the MSE of the second dataset is higher than the MSE of the fifth dataset even though the number of training samples of the second dataset is larger than the number of samples of the fifth dataset. In other words, more training samples of the second dataset have not led to a lower error. That is because the error of the datasets cannot be justified

TABLE 14. Performance of ANN and LMN on different datasets.

Dataset	MSE of ANNs	MSE of LMNs	Training Samples
#1	1.1746E-03	1.2526E-03	260
#2	1.7366E-03	4.7900E-03	261
#3	2.4953E-04	1.9966E-03	344
#4	5.7781E-03	1.2519E-02	71
#5	1.5934E-03	2.8904E-03	88
#6	9.4251E-05	1.2278E-04	1102

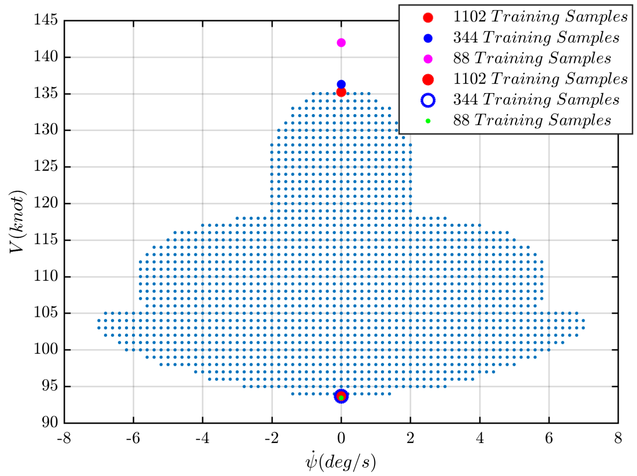


FIGURE 29. Rudder restricted to [−30°, +10°] at 30000 ft and $\gamma = 1°$.

solely based on the number of samples, but it is mainly due to the considered values of the input parameters. According to Tables 9 and 12, the considered rudder defection values of the second dataset are [−20°, 0°, 20°] whereas for the fifth dataset are [−30°, −10°, 10°, 30°]. This shows that ANN can better model the nonlinear relationship of the inputs and targets when δ_r instances are reduced and considered as [−30°, −10°, 10°, 30°] rather than [−20°, 0°, 20°].

The same explanation exists for the difference between the MSEs of the first and second datasets.

As shown in Fig. 28, the best performance (i.e. the lowest error) among the 5 diminished datasets belongs to the third dataset whose sample size is almost 69% smaller than the original dataset. The following figures present the values of V_{min} and V_{max} estimated by the third, fifth, and the original datasets for the same failure cases of the previous subsection.

As can be seen in figures 29 to 32, the numerical values of V_{min} and V_{max}estimated by the third dataset are very close to those estimated by the original dataset. In fact, their difference is not significant considering that the amount of training data of the third dataset is only 31% of the original dataset. The results of the fifth dataset (with 8% of the size of the original dataset) are almost the same as the third dataset except for the failure case of Fig. 29 where the error of estimating V_{max} is considerable. Hence, the fifth dataset cannot generalize as well as the third dataset and is not a good choice.

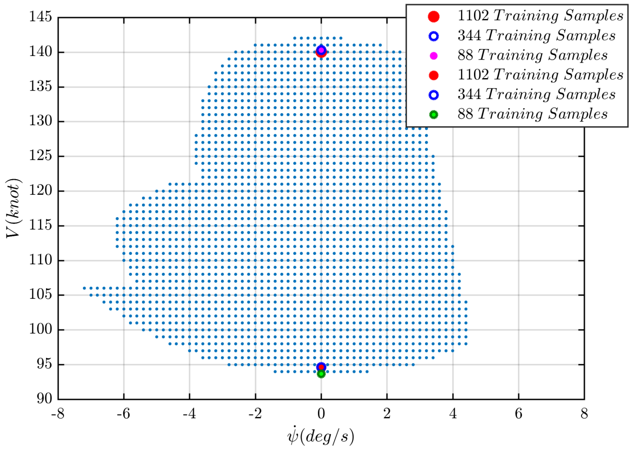


FIGURE 30. Rudder jammed at 10° at 30000 ft and $\gamma = 0°$.

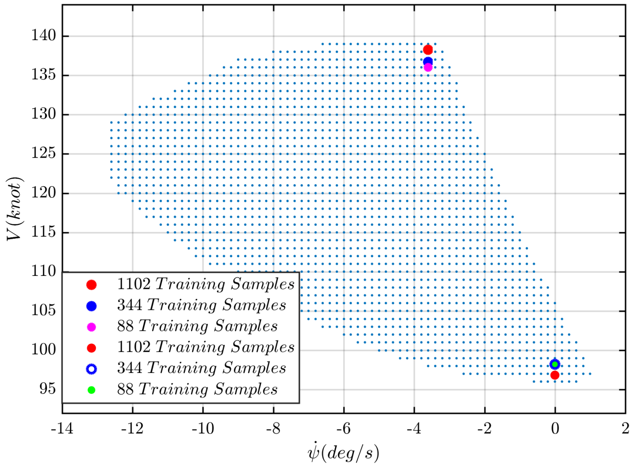


FIGURE 31. Rudder jammed at 25° at 10000 ft and $\gamma = -5°$.

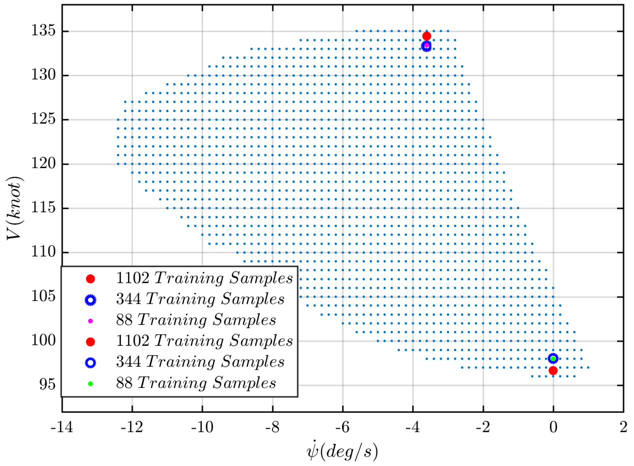


FIGURE 32. Rudder jammed at 25° at 10000 ft and $\gamma = -4°$.

Therefore in order to have a good compromise, the amount of training data can be decreased to more than one third which significantly reduces the required time and the computational cost of the training data generation.

Obviously, none of the ANNs trained with the 5 diminished datasets can generalize to the independent test data as well as the best ANN trained with the original dataset. In other words, none of the ANNs of this section can predict the values of V_{min} and V_{max} as accurate as the best ANN of the previous subsection. However, the aim of this section is to show that the performance of a network is not determined only by the number of training samples, and in the case of the need to fast training data collection where not enough training data could be generated, the instances of the input parameters could be chosen such that the nonlinear relationship between the inputs and targets is established as well as possible, and the errors of the model are negligible.

Base on the results of this subsection, it can be inferred that in order to have a relatively acceptable performance, the minimum and maximum of the input parameter's range must be included in the training dataset along with a middle value or symmetrical values within the range.

IV. NEURAL NETWORK-BASED SENSITIVITY ANALYSIS

In this section, a neural network-based global sensitivity analysis approach is presented which enables assessing the degree of effect of different contributing parameters to the variations of the impaired aircraft's maneuvering flight envelope key characteristics.

This assessment is specifically very important for the post-failure path planning where prior understanding of the affecting parameters on the aircraft performance is vital for a safe trajectory and has an important role in specifying the underlying path planning strategy. More discussion and examples on this are presented at the end of this section.

Generally, sensitivity analysis which investigates how the variation of a system model is attributed to the changes in input factors requires numerous model evaluations regardless of the sensitivity analysis method used. Roughly speaking, thousands of model evaluations are required by global sensitivity analysis methods [36]. The higher the nonlinearity and complexity of the model, the more model evaluations are needed for an accurate analysis.

In the case of a high-fidelity nonlinear 6 DOF aircraft model, the relationship between aircraft performance parameters and specific input factors is governed by multiple inter-related equations involving other variables. Hence the model evaluations required by the sensitivity analysis methods are such computationally intensive that even a semi-accurate global sensitivity analysis becomes almost impossible.

For instance, in the case of aircraft rudder failure presented in this paper, it was shown that the values of V_{min} and V_{max} are dependent on the values of altitude (h), flight path angle (γ), and rudder failure degree (δ_r). However, there are no direct equation relating $h\gamma$, and δ_r to V_{min} and V_{max}. Mathematically, the values of V_{min} and V_{max} could be calculated via the state derivative equation ($\dot{V}$) which is the first equation among the 12 equations of motion of aircraft:

$$\dot{V} = g(cos\,\phi\, cos\,\theta\, sin\,\alpha\, cos\,\beta + sin\,\phi\, cos\theta\, sin\beta - sin\,\theta\, cos\,\alpha\, cos\,\beta) - \frac{\bar{q}S}{m} C_{D_{wind}} + \frac{T}{m} cos(\alpha + \alpha_T)\, cos\,\beta \tag{36}$$

In which, g is the gravitational acceleration, $\bar{q}$ is dynamic pressure, T is the aircraft thrust force, α_T is the thrust angle, and $\mathrm{C_{D_{wind}}}$ is the drag coefficient in wind axes [23].

(36) is dependent on the altitude (h) through the air density (ρ) in the dynamic pressure:

$$\bar{q} = \frac{1}{2}\rho V^2 \tag{37}$$

It is also dependent on flight path angle (γ) both via the thrust-required equation:

$$T_R = W\gamma + \frac{1}{2}\rho V^2 S C_{D_0} + {}^{2KW^2}\Big/_{\rho V^2 S cos^2\phi} \tag{38}$$

and the following equivalency [23]:

$$g(cos\,\phi\, cos\,\theta\, sin\,\alpha\, cos\,\beta + sin\,\phi\, cos\,\theta\, sin\,\beta - sin\,\theta\, cos\,\alpha\, cos\,\beta) = -g\, sin\,\gamma \tag{39}$$

Rudder failure degree (δ_r) is one of the components of the aircraft non-dimensional forces and moments coefficients and exists in (36) through the body-axes side force coefficient (C_Y) and body-axes drag force coefficient (C_D) which appear in the equation of the drag coefficient in wind axes:

$$C_{D_{wind}} = C_D\, cos\,\beta - C_Y\, sin\,\beta \tag{40}$$

However, ρ, ϕ, θ, α, β, and δ_r are components of other aircraft equations of motion too. Hence evaluating V_{min} and V_{max} based on (36) requires solving all the aircraft nonlinear equations of motion ($\dot{x} = 0$) simultaneously for the corresponding trim point. Moreover, as explained earlier, the process should be iterated over multiple trim points until the one corresponding to V_{min} or V_{max} is found. That is because in the case of an impaired aircraft, due to additional drag resulted by the damaged rudder, V_{max} is no longer at its nominal value and V_{min} is no longer evaluable through the conventional stall speed equation.

Considering the aforementioned procedure, evaluating at least thousands of model outputs (V_{min} or V_{max}) in order to assess the sensitivity of V_{min} or V_{max} with respect to input factors (h, γ, and δ_r) is computationally expensive and practically impossible. Furthermore, putting aside the issue of computational cost, aircraft equations of motion do not include the lower limit and upper limit of the rudder deflection angle as input variables. Instead, these limits are exerted during the nonlinear optimization process of deriving trim points.

On the other hand, the regression equation of the neural network provides a direct link between the intended input factors and the model output. Specifically, the regression equation of the best ANN (i.e. with the least MSE) from the previous sections can be used as an emulation model to obtain much faster evaluations of the model response. An emulator which is a computationally efficient model and usually has the form of an algebraic relation is calibrated over a dataset and can be used instead of a computationally demanding model in case faster model evaluations are required as in sensitivity analysis. The following subsection elaborates on the proposed emulation model [36].

A. ANN REGRESSION EQUATION AS EMULATOR

As explained in subsection II.C, the ANNs considered in this research are in the form of a two-layer network including one hidden layer with nonlinear sigmoid transfer function and one output layer with linear transfer function which is the network architecture often used for function fitting (nonlinear regression). This two-layer architecture was shown in Fig. 4.

In order to obtain the ANN regression equation, the trained network computational process should be simulated using the network's weights, biases, and settings associated with the pre/post – processing functions. For the rudder damage scenario considered in this research, the ANN regression equation ($\mathcal{F}$) has the following general form:

$$V_{min} \text{ } or \text{ } V_{max} = \mathcal{F}(h, \gamma, \delta_r min, \delta_r max) \tag{41}$$

Before training the ANN, it is useful to scale the inputs and targets data such that they fall in the range $[-1, 1]$. This is done using a mapping function which processes inputs and targets matrices and normalizes the minimum and maximum values of each row to $[-1, 1]$. After the network is trained, the mapping settings should be applied to any inputs that are given to the trained network. Hence, considering the vector I to be the new inputs being applied to the trained network:

$$I = \begin{bmatrix} h \\ \gamma \\ \delta_r min \\ \delta_r max \end{bmatrix}_{4\times1} \tag{42}$$

We have the processed inputs vector I_p such that:

$$\begin{aligned} I_{p(4\times1)} = {} & [I - IV_{min}]_{4\times1} \circ \big[[1-(-1)]_{4\times1} \\ & \oslash [IV_{max} - IV_{min}]_{4\times1}\big] + [(-1)]_{4\times1} \end{aligned} \tag{43}$$

where IV_{min} and IV_{max} are vectors of the minimum and maximum of the training inputs data, and $\circ$ and $\oslash$ denote Hadamard product and division, respectively. It should be noted that I_p is only a function of I once the numerical values of IV_{min} and IV_{max} are substituted.

Next the I_p vector is multiplied by the hidden layer's weights matrix, summed with the hidden layer's biases vector, and then the hyperbolic tangent sigmoid transfer function is applied to the resulting vector to generate the output vector of the hidden layer:

$$\begin{aligned} a^1_{10\times1} &= tansig\left(IW_{(10\times4)} I_{p(4\times1)} + b^1_{(10\times1)}\right) \\ &= [[2]_{10\times1} \oslash \left[1+e^{-2(IWI_p+b^1)}\right]_{10\times1}] - [1]_{10\times1} \end{aligned} \tag{44}$$

where a^1 represent the output of the hidden layer transfer function, IW and b^1 represent weights and biases of the hidden layer, $tansig$ denotes the tan-sigmoid transfer function, and any vector in the form of $[k]_{m\times1}$ represents m rows with identical values of k [25].

Finally, the model output which is the output of the linear transfer function of the output layer is calculated by applying the output layer's weights and biases vectors to a^1:

$$a^2_{2\times1} = LW_{(2\times10)} a^1_{(10\times1)} + b^2_{(2\times1)} = \begin{bmatrix} V_{max} \\ V_{min} \end{bmatrix} \tag{45}$$

Each row of the two-row vector a^2 is an ANN regression function. Based on the training data structure presented in Fig. 9, the top and bottom row correspond to the regression functions producing $\mathrm{V_{max}}$ and $\mathrm{V_{min}}$ from the four input factors h, γ, $\delta_r min$, and $\delta_r max$, respectively.

For the damage case considered in this research and using the explained network architecture, each of the two obtained regression equations is in the form of the sum of a constant decimal number and 10 fractions with exponential functions in the denominators:

$$\begin{bmatrix} V_{max} \\ V_{min} \end{bmatrix} = \begin{bmatrix} \dfrac{c1}{e^{g1(h,\gamma,\delta_{rmin},\delta_{rmax})}+1} + \cdots + \dfrac{c10}{e^{g10(h,\gamma,\delta_{rmin},\delta_{rmax})}+1} + c11 \\ \dfrac{d1}{e^{k1(h,\gamma,\delta_{rmin},\delta_{rmax})}+1} + \cdots + \dfrac{d10}{e^{k10(h,\gamma,\delta_{rmin},\delta_{rmax})}+1} + d11 \end{bmatrix} \tag{46}$$

where $c1, \ldots, c11$ and $d1, \ldots, d11$ are constant decimal numbers, and $g1, \ldots, g11$ and $k1, \ldots, k11$ are linear combinations of h, γ, $\delta_r min$, $\delta_r max$.

Each of the two regression equations indicated in (46) can be used as an emulator during the sensitivity analysis of the aircraft performance parameter $\mathrm{V_{max}}$ or $\mathrm{V_{min}}$ with respect to the input factors (h, γ, $\delta_r min$, $\delta_r$max). Each emulator is a function of only the intended variables (i.e. input factors), hence enabling efficient model evaluations. To be specific, the emulators obtained based on the 4th ANN's regression functions were capable of around 30000 model evaluation per minute on a standard desktop PC with 3.00 GHz AMD Phenom quad-core processor, under Windows 7 operating system, and using MATLAB® version 9.3 (R2017b). As mentioned earlier, without using an ANN-based emulator, each model evaluation would require multiple trim point evaluations and could take several minutes to complete.

Another advantage of using ANN-based emulator is that it can be used as a unique tool to visualize the variation of the intended model output with respect to specific input factors. As will be seen in the rest of this section, this feature of the ANN-based emulator is very useful in better understanding and interpretation of the results of the sensitivity analysis. For instance, Fig. 33 depicts the variations of $\mathrm{V_{max}}$ with the changes in the lower and upper limits of the rudder deflection angle (i.e. changes in the failure degree) at 10000 ft and zero flight path angle. It can be seen that extreme changes in $\mathrm{V_{max}}$ correspond to sever rudder restriction cases with

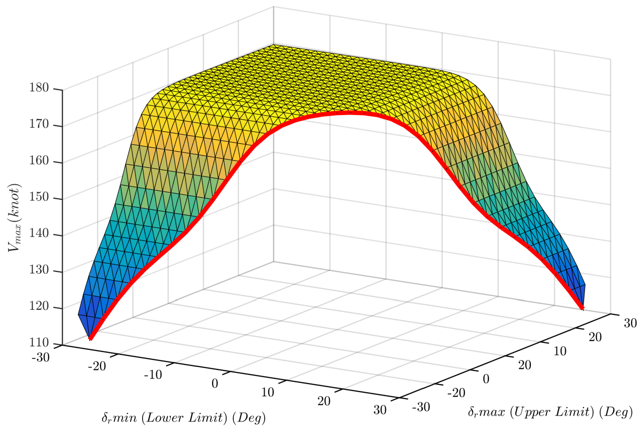


FIGURE 33. $\mathbf{V_{max}}$ **versus** δ_r **min and** δ_r **max at 10000 ft and** $\gamma = 0°$**.**

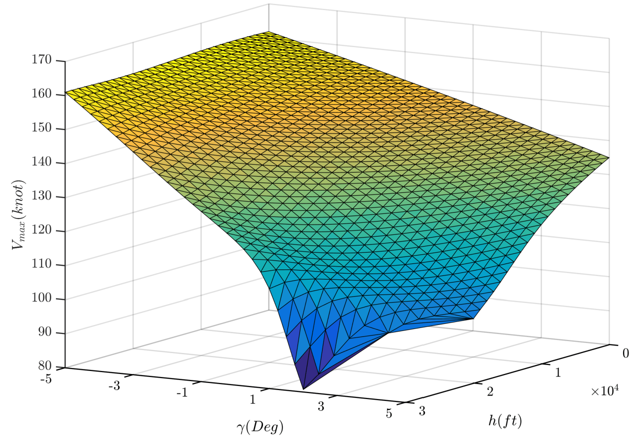


FIGURE 34. $\mathbf{V_{max}}$ **versus** γ **and** h **for rudder jammed at 15°.**

high deflection angles. In this figure, the red thick curve demonstrates rudder jamming cases.

Also, Fig. 34 represents the variations of V_{max} with changes in the altitude and flight path angle for an impaired GTM with jammed rudder at 15°.

The next subsection discusses the global sensitivity analysis methods used along with the numerical results of their applications to the derived ANN-based emulators.

B. GLOBAL SENSITIVITY ANALYSIS

Two main classes of sensitivity analysis methods are local and global based on the input variability space under analysis [37]. Local method investigates the output variability due to variations of an input factor around a specific nominal value. On the other hand, global sensitivity analysis considers the variations of input factors within the entire variability space [36].

Sensitivity analysis methods also differ in their sampling strategy. In OAT (i.e. One factor At a Time) sampling, all input factors except one factor at a time are kept fixed, whereas in non-OAT sampling all input factors are varied simultaneously. Utilizing the latter strategy can characterize the interactions between different combinations of input factors, while methods using OAT sampling at best can only give a hint on the importance of the interactions [36].

The local method which is based on the OAT sampling measures the output sensitivity to the i^{th} input factor through the partial derivative of the regression equation at some nominal input value I^0, which is approximated by the corresponding finite differences. Hence, for the aforementioned damage scenario considered in this research we have, (47), as shown at the bottom of this page, where, $\eta = \frac{I^0}{\mathcal{F}(I^0)}$ is the normalization factor used to rescale the sensitivities of different factors with different units. The number of required model evaluations in the local method is $(\mathcal{M} + 1)$ with $\mathcal{M}$ being the number of input factors which is 4 in the case study of this research.

For a highly nonlinear system such as an impaired aircraft, local method is not a proper choice for sensitivity analysis as it only provides the local sensitivity and is not capable of assessing the sensitivity through the whole input space.

An extended version of the local method which evaluates the global sensitivity by aggregating multiple local sensitivities measured over different points within the input space is the Morris method [38], also named the Elementary Effect Test (EET) [39]. In this method, r trajectories are built in the input space each comprising $(\mathcal{M} + 1)$ points. The starting point of each trajectory can either be selected using random sampling (as originally suggested by Morris [38]) or by Latin-Hypercube sampling as proposed in [40]. Latin-Hypercube sampling is a particular type of stratified sampling which reduces the gaps between the clusters of sampled points and hence produces a more uniform sample grid than the ones generated by the pseudo-random number generator in the random sampling strategy [41].

Once the starting point in each trajectory is selected, the subsequent points of the trajectory are evaluated either by moving one input at a time (OAT approach) a fixed step Δ (i.e. pre-specified step between two consecutive points of the trajectory) or through the radial-based design where variations of all subsequent points of the trajectory are taken from the starting point of the trajectory [42]. The latter provides more efficiency and conforms better to the non-OAT sampling sensitivity analysis [36]:

$$I_i = I_0 + \Delta_i, I_{i+1} = I_0 + \Delta_{i+1} \tag{48}$$

where I_0 is the starting point of the trajectory and each point has its corresponding variation step Δ.

$$\eta(\frac{\partial \mathcal{F}}{\partial I_i}\bigg|_{I^0}) = \eta \left\{ \frac{\mathcal{F}(I_1, \ldots, I_i + \Delta_i, \ldots, I_{\mathcal{M}}) - \mathcal{F}(I_1, \ldots, I_i, \ldots, I_{\mathcal{M}})}{\Delta_i} \right\} \tag{47}$$

In the EET method, the Elementary effects are finite differences calculated via (47). The mean of these Elementary effects over the complete set of trajectories is a sensitivity measure ($\mathrm{S_i}$) and indicates the effect of input factor $\mathrm{I_i}$ on the model output:

$$S_i = \frac{1}{r}\sum_{j=1}^{r}\left(\eta\left(\left.\frac{\partial\mathcal{F}}{\partial I_i}\right|_{I^0}\right)\right)^j \tag{49}$$

The standard deviation of Elementary effects is also a sensitivity measure indicating the effect of nonlinearities and interactions of input factor I_i on the model output variations [37]:

$$\sigma_i = \sqrt{\frac{1}{r-1}\sum_{1}^{r}\left[\eta\left(\left.\frac{\partial\mathcal{F}}{\partial I_i}\right|_{I^0}\right) - S_i\right]^2} \tag{50}$$

The EET method provides a general indication of each input factor's main-effect and interactions, with far less number of required model evaluations than other more accurate and detailed global sensitivity methods. Specifically, it is useful for detecting non-influential factors that can be disregarded in other time-consuming global methods [36].

In the following subsection, we have applied the EET method with Latin-Hypercube sampling and radial-based design to the impaired GTM with rudder damage.

One of the most powerful global sensitivity analysis methods is the variance-based method which provides accurate numerical indices for the main-effects, total-effects, and interactions of input factors. However, this method requires tens of times more model evaluations than the EET method [36]. Very fast model evaluations through the derived ANN-based emulator practically meet this requirement. In the following subsection, the numerical results of applying this method to the case study of this research are presented.

In variance-based sensitivity analysis, the contribution to the model output variance from a specific input factor is considered a measure of sensitivity. Assuming Y to be the model output we have [39]:

$$E_{I_i}\left(V_{I_{\sim i}}\left(Y|I_i\right)\right) + V_{I_i}\left(E_{I_{\sim i}}\left(Y|I_i\right)\right) = V(Y) \tag{51}$$

where V and E denote variance and expected value, respectively. The first term of (51) which represents the average of conditional variance of Y, taken over all factors but the i^{th} input factor (I_i) when I_i is fixed, is a measure of influence of other factors but I_i. Hence, the smaller this term, the greater the second term of (51) and the influence of I_i.

Therefore the second term is the first-order effect (main-effect) of I_i on Y, and the corresponding first-order sensitivity index is defined as:

$$S_i = \frac{V_{I_i}\left(E_{I_{\sim i}}\left(Y|I_i\right)\right)}{V(Y)} \tag{52}$$

Similarly, the total-effect of the input factor I_i which is the overall contribution from the i^{th} input to the output variance including the first-order effect and all interactions with other inputs (i.e. all higher order effects due to these interactions) is defined as [39]:

$$S_i^T = \frac{E_{I_{\sim i}}\left(V_{I_i}\left(Y|I_{\sim i}\right)\right)}{V(Y)} = 1 - \frac{V_{I_{\sim i}}\left(E_{I_i}\left(Y|I_{\sim i}\right)\right)}{V(Y)} \tag{53}$$

When the input factors are independent (i.e. orthogonal), the variance of the model output can be decomposed using the so-called ANOVA-HDMR (High Definition Model Representation) decomposition [37], [39]:

$$V\left(Y\right) = \sum\nolimits_i V_i + \sum\sum\nolimits_{j>i} V_{ij} + \ldots + V_{12\ldots\mathcal{M}} \tag{54}$$

Dividing both sides of (54) by V (Y) enables the evaluation of higher order indices which correspond to the effect of interactions between input factors that cannot be expressed as the sum of their main effects [39].

In the case of four input factors as in this research, the relation between the Sobol indices is:

$$S_i^T = S_i + S_{ij} + S_{ik} + S_{ih} + S_{ijk} + S_{ijh} + S_{ikh} + S_{ijkh} \tag{55}$$

where $\mathrm{S_{ij}}$, $\mathrm{S_{ik}}$, $\mathrm{S_{ih}}$, S_{ijk}, $\mathrm{S}_{\mathrm{ij}h}$, S_{ikh}, and $\mathrm{S_{ijkh}}$ indicate higher order indices.

First and total-order effect indices are generally estimated via Monte-Carlo estimators within a Monte-Carlo based numerical procedure. As proposed by Kucherenko *et al.* [44], $\mathrm{S_i}$ and $\mathrm{S_i^T}$ can be efficiently estimated through the following steps:

- Two $(N, \mathcal{M})$ matrices of random numbers are generated and defined as A and B, where N is the number of base samples.
- A matrix C_i is constructed from all columns of B except the i^{th} column which is taken from A.
- Model output is evaluated for all samples in the matrices A, B, and C_i:

$$Y_{A(N\times 1)} = \mathcal{F}\left(A\right),\ Y_{B(N\times 1)} = \mathcal{F}\left(B\right),\ Y_{C_i(N\times 1)} = \mathcal{F}(C_i) \tag{56}$$

-

$$S_i = \frac{V_{I_i}\left(E_{I_{\sim i}}\left(Y|I_i\right)\right)}{V(Y)} = \frac{\left({}^1\!/\!_N\right)\sum_{j=1}^{N} Y_A^j Y_{Ci}^j - \mathcal{F}_0^2}{\left({}^1\!/\!_N\right)\sum_{j=1}^{N} {(Y_A^j)}^2 - \mathcal{F}_0^2} \tag{57}$$

where

$$\mathcal{F}_0^2 = \left(\frac{1}{N}\sum\nolimits_{j=1}^{N} Y_A^j\right)^2 \tag{58}$$

-

$$S_i^T = 1 - \frac{\left({}^1\!/\!_N\right)\sum_{j=1}^{N} Y_B^j Y_{Ci}^j - \mathcal{F}_0^2}{\left({}^1\!/\!_N\right)\sum_{j=1}^{N} {(Y_A^j)}^2 - \mathcal{F}_0^2} \tag{59}$$

To evaluate the higher order indices associated with the interactions between inputs, the matrix $C_{i\ldots j}$ is constructed in the same manner as C_i, except that all columns corresponding to the intended inputs must be taken from matrix A.

$$S_i = \frac{\left({}^{1}/_{N}\right)\sum_{j=1}^{N}\left[\mathcal{F}(i,\sim i)^{j}\mathcal{F}(i,\sim i^{\prime})^{j}\right]-\left(\frac{1}{N}\sum_{j=1}^{N}\mathcal{F}(i,\sim i)^{j}\right)^{2}}{\left({}^{1}/_{N}\right)\sum_{j=1}^{N}\left(\mathcal{F}(i,\sim i)^{j}\right)^{2}-\left(\frac{1}{N}\sum_{j=1}^{N}\mathcal{F}(i,\sim i)^{j}\right)^{2}} \tag{60}$$

$$S_i^T = 1-\frac{\left({}^{1}/_{N}\right)\sum_{j=1}^{N}\left[\mathcal{F}(i^{\prime},\sim i^{\prime})^{j}\mathcal{F}(i,\sim i^{\prime})^{j}\right]-\left(\frac{1}{N}\sum_{j=1}^{N}\mathcal{F}(i,\sim i)^{j}\right)^{2}}{\left({}^{1}/_{N}\right)\sum_{j=1}^{N}\left(\mathcal{F}(i,\sim i)^{j}\right)^{2}-\left(\frac{1}{N}\sum_{j=1}^{N}\mathcal{F}(i,\sim i)^{j}\right)^{2}} \tag{61}$$

For instance, to evaluate $\mathrm{S_{ij}}$, a matrix C_{ij} is constructed by substituting the i^{th} and j^{th} columns of B with the same columns from A [37].

The total computational cost of the aforementioned numerical procedure is $N(\mathcal{M}+2)$, due to $2N$ model evaluations required for matrices A and B, and $N\mathcal{M}$ model evaluations required for the complete set of matrix C.

When all or some of the input factors are not independent, the decomposition of (54) does not hold. In such cases, the joint probability density function of the dependent inputs and conditional distributions must be taken into account instead of the marginal distributions considered when the inputs were independent. It is suggested in [44] that the first and total-order indices of a dependent input factor could be estimated via the Monte-Carlo estimators, (60) and (61), as shown at the top of this page. Equations (60) and (61) are similar to the estimators suggested by Saltelli (i.e. (57) and (59)), except that $\mathcal{F}(\mathrm{i},\sim\mathrm{i})^{\mathrm{j}}$ is the model evaluations based on the joint density function of the input factor i and other input factors, $\mathcal{F}(\mathrm{i},\sim\mathrm{i}')^{\mathrm{j}}$ is the model evaluations based on the conditional distribution of other factors when the i^{th} input factor is fixed, and $\mathcal{F}(\mathrm{i}',\sim\mathrm{i}')^{\mathrm{j}}$ is the model evaluations based on the joint density function of the input factor i and other input factors in the second base samples (similar to matrix B).

Among the four input factors of the case study of this research (h, γ, $\delta_r min$, $\delta_r max$), the lower limit and upper limit of the rudder deflection angle are correlated, which means their values are dependent upon each other. For instance, an impaired rudder with restricted lower limit of −10° cannot have a restricted upper limit of −20° (i.e. failure case [−10°, −20°] is not valid), whereas for the lower limit being at −25° ; an upper limit value of −20° is valid (i.e. failure case [−25°, −20°] is valid). Hence, in order to estimate the Sobol indices, (60) and (61) were used along with the sampling strategy proposed in [44]. In this strategy, uniformly distributed random numbers are generated between 0 and 1, and then samples are transformed into their variability range based on their joint distribution function. All lower limit and upper limit samples should lie within a triangle observing the following inequality condition:

$$\delta_r min \leq \delta_r max \tag{62}$$

According to the nominal range of rudder deflection angle [−30°, 30°], the mentioned triangle has the coordinates: $A_1(-30,\ 30)$, $A_2(-30,\ -30)$, and $A_3(30,\ 30)$. The following equation transforms any two randomly selected samples $(p1, p2)$ between 0 and 1 to a point within the triangle boundary [44]:

$$q=\left(1-\sqrt{p1}\right)A_1+\sqrt{p1}\,(1-p2)\,A_2+\sqrt{p1}p2A_3 \tag{63}$$

Utilizing this transformation along with the estimators presented in (60) and (61) enables applying the variance-based method to the rudder damage scenario.

C. NUMERICAL RESULTS AND DISCUSSION

As explained in the previous subsection, since the ANN trained for the case study of this research has two outputs ($\mathrm{V_{max}}$ and $\mathrm{V_{min}}$), there are two regression equations each acting as an emulator for the corresponding output. So the numerical results have been obtained for both $\mathrm{V_{max}}$ and $\mathrm{V_{min}}$.

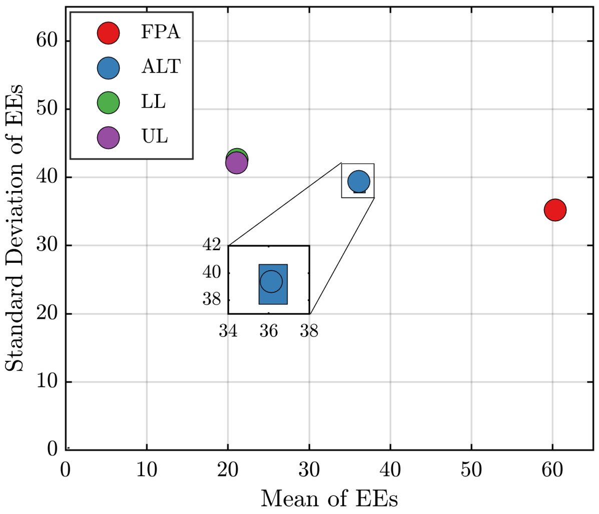


FIGURE 35. Effects of the inputs on $\mathrm{V_{max}}$ based on the EET method.

Figures 35 and 36 present the means and standard deviations of the Elementary effects of the input factors on $\mathrm{V_{max}}$ and $\mathrm{V_{min}}$, respectively. The more mean of EEs of a point, the more effective the input factor. The more deviation of EEs of a point, the more its interactions with other inputs. As can be seen, the obtained results are accurate enough for the method used and the confidence bounds (shown as rectangles) are narrow. As mentioned earlier, the purpose of the EET method is a general analysis of the sensitivities that can be achieved with fewer model evaluations than the

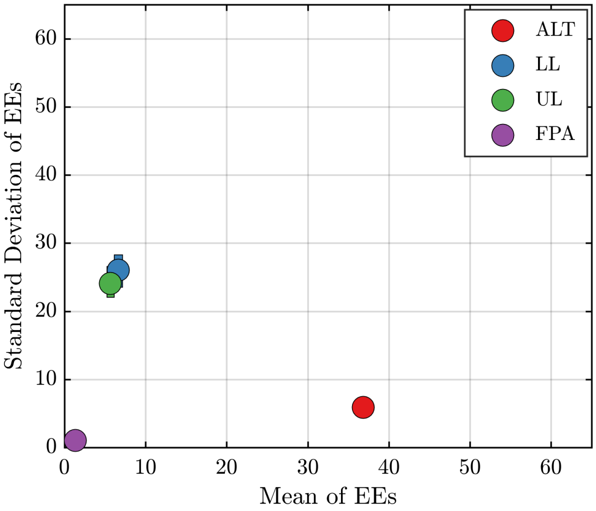


FIGURE 36. Effects of the inputs on V_{min} based on the EET method.

precise variance-based method. To obtain these results via the EET method with Latin-Hypercube sampling and radial-based design, 12000 trajectories were considered and a total of 60000 model evaluations were conducted ($\mathcal{M}+1$ samples per trajectory). Samples for this method were also generated based on the joint density function explained in the previous subsection.

Based on the evaluated Elementary effects, flight path angle is the most influent input factor on the variations of V_{max}. Lower limit and upper limit of rudder have the least individual effects but their interactions and nonlinearities are more than the other input factors. Flight path angle is a non-influential factor on the variations of V_{min}, both individually and based on the interactions. Altitude has the most main-effect on V_{min}, and lower and upper limits of rudder have the highest interactions on V_{min} as on V_{max}.

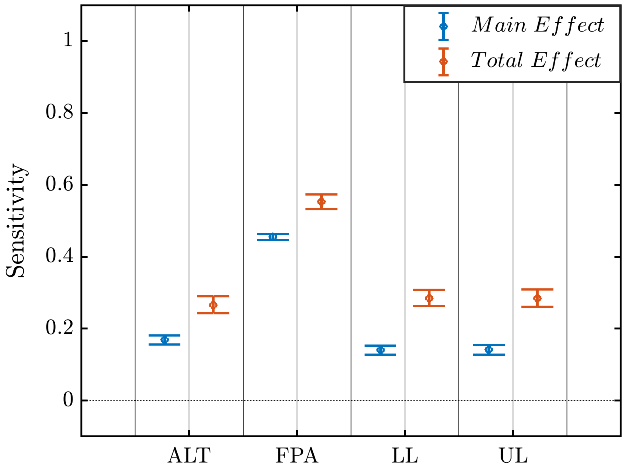


FIGURE 37. First and total-order indices for the model output V_{max}.

Figures 37 and 38 present the Sobol sensitivity indices of the input factors obtained through the variance-based method for the model outputs V_{max} and V_{min}, respectively. As can be seen, the error bars show low estimation errors for the evaluated indices. The results are similar to those obtained by the EET method. First and total order indices of the lower and upper limits of rudder are identical because aircraft and rudder deflection range are symmetrical about the body x-axis, and slight differences between their numerical values are due to the numerical method errors.

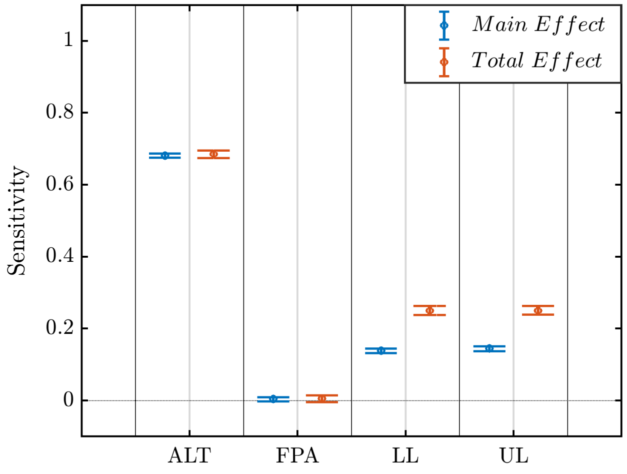


FIGURE 38. First and total-order indices for the model output V_{min}.

Numerical values of the first-order, higher order, and total-order Sobol indices are presented in Table 15:

TABLE 15. Sobol indices of input factors for model outputs V_{max} and V_{min}.

V_{max}				V_{min}			
S_h	0.1687	S_h^T	0.2660	S_h	0.6809	S_h^T	0.6849
S_γ	0.4558	S_γ^T	0.5528	S_γ	0.0042	S_γ^T	0.0051
$S_{\delta rmin}$	0.1411	$S_{\delta rmin}^T$	0.2848	$S_{\delta rmin}$	0.1385	$S_{\delta rmin}^T$	0.2498
$S_{\delta rmax}$	0.1417	$S_{\delta rmax}^T$	0.2840	$S_{\delta rmax}$	0.1444	$S_{\delta rmax}^T$	0.2501
$S_{h\gamma}$	0.0973			$S_{h\gamma}$	$\cong 0$		
$S_{\delta rmin\delta rmax}$	0.1437			$S_{\delta rmin\delta rmax}$	0.1113		

As can be seen in Fig. 37, flight path angle has the largest value of first-order index, and altitude has the second rank. The main-effect of each rudder deflection limit is small because most samples fall in the smooth area of the surface shown in Fig. 33, where variations of V_{max} are negligible. However, there are instances where deflection limits are very close to each other (i.e. failure is severe) and very drifted with respect to the rudder neutral point, in such cases the variation of V_{max} is extreme. For instance, a lower limit of $-25°$ does not have a significant main-effect because for most samples its combinations with the upper limit value correspond to a point on the smooth area of the aforementioned surface. However, samples containing the combination $[-25°, -23°]$ result in a big reduction of V_{max}. These interactions between the rudder deflection limits are responsible for the non-zero second order index $S_{\delta rmin\delta rmax}$ which is the difference between the deflection limits' first-order and total-order indices. As shown in Fig. 34, variations in flight path angle results in a persistent decrease in V_{max}, hence flight path angle has very large main-effect. But also it can be seen in the same figure that at high altitudes and certain flight path angles, the variation of V_{max} is more drastic. This is due to interactions between altitude and flight path angle and accounts for the non-zero index $S_{h\gamma}$. Higher order indices other than $S_{\delta rmin\delta rmax}$ and $S_{h\gamma}$ are zero.

According to the obtained results, flight path angle does not have any effect on the variations of V_{min}, that is because by changing the flight path angle, the lower boundary of the maneuvering flight envelope remains the same and does not change. Altitude is the most influential factor on the variations of V_{min}, and rudder deflection limits have almost the same main and total-effects as they have on V_{max}. Unlike the model output V_{max}, there are no interactions between altitude and flight path angle for V_{min} and slight differences between main and total-effect indices of these input factors is due to numerical procedure errors. Variance-based method results were obtained using a total of 3 million model evaluations conducted in about 100 minutes.

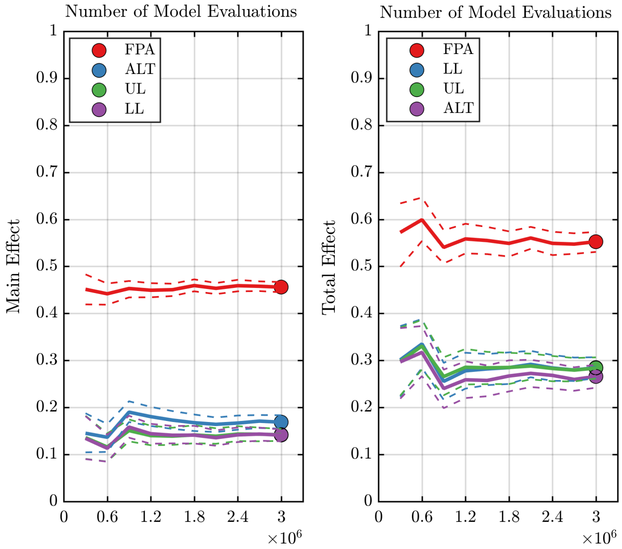


FIGURE 39. Convergence analysis of first and total-order indices for the model output V_{max}.

It is important to evaluate the robustness of the estimated indices. This is done by deriving confidence intervals via bootstrapping through a convergence analysis [45]. Figures 39 and 40 present the analysis results for the estimated indices for both model outputs V_{max} and V_{min}. It can be seen that for both model outputs, the confidence intervals of the evaluated indices have converged at 3 million model evaluations. This shows that the estimated indices are robust with respect to different samples of the model.

Due to the good structured ANN-based emulator used, such a large number of model evaluations was computationally affordable. However, in the case were other failure scenarios with higher number of input factors are considered, a quasi-random sampling method with low-discrepancy such as Sobol sequences can reduce the number of the required samples to estimate the Sobol indices [37], [44].

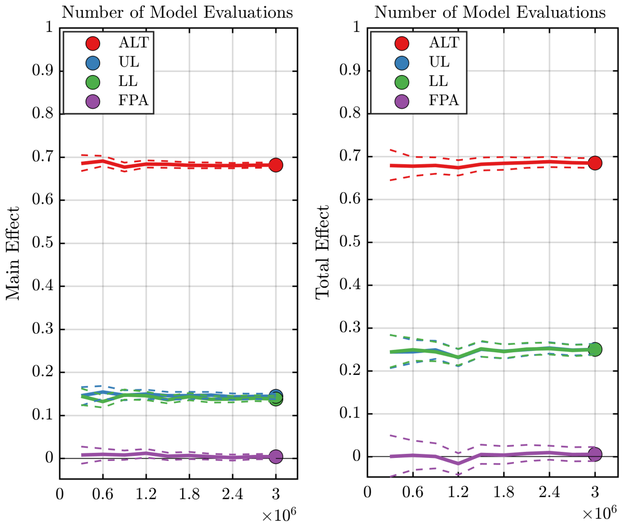


FIGURE 40. Convergence analysis of first and total-order indices for the model output V_{min}.

As mentioned at the beginning of this section, by utilizing the proposed ANN-based sensitivity analysis approach, the ANN which is going to be used onboard the impaired aircraft for real-time evaluation of V_{max} or V_{min}, can also be used to create a prior knowledge of the influential input factors and their interactions so that a safer path planning strategy is chosen for the impaired aircraft during flight.

Maneuvering flight envelope of an impaired aircraft is contracted due to the imposed failure. A secondary failure which may be induced due to the excessive use of the damaged control surface would shrink the already restricted flight envelope, hence increasing the possibility of loss of control. Also, changing the flight condition to an altitude or a flight path angle that contracts the maneuvering flight envelope more than before; would increase the risk of loss of control following a new failure or an adverse atmospheric condition. Therefore it is important to identify the parameters with significant effect on the flight envelope limits (e.g. V_{max} and V_{min}). Such parameters could be control surfaces deflection limits, altitude, and flight path angle in the case of actuator failures. Once the corresponding parameters are identified, a path planning strategy with less emphasize on changing those parameters should be used instead of the nominal strategies such as minimum-time or minimum-fuel. The proposed sensitivity analysis method which provides accurate results on the influential parameters can be used as an advisory for the decision making process of choosing the safer path planning strategy.

For instance, in the case of a terrain ahead of an impaired aircraft with damaged rudder, assuming that collision with terrain could be avoided either by performing a shorter climbing turn or a longer level turn, even though performing a climbing turn is in compliance with the minimum-time path planning, it requires an increase in the flight path angle which based on the sensitivity analysis results would lead to a considerable reduction of $\mathrm{V}_{\mathrm{max}}$ (i.e. shrinking maneuvering flight envelope). This is important because in this case where maneuvering flight envelope is more limited, a secondary control surface failure could lead to a complete loss of control of the aircraft. However, maneuvering flight envelope in the longer level turn is less limited because flight path angle is not changed, therefore it is less likely for a secondary surface failure to eliminate the whole flight envelope and cause loss of control. Hence, it is more reliable to choose the level turn to avoid the terrain even though it is not the minimum-time path.

Another case is when the aircraft is climbing and rudder failure occurs. In such case, it is safer to perform the necessary climbing with lower climbing rate (smaller flight path angle). Because based on the sensitivity analysis results, altitude has smaller effect on $\mathrm{V}_{\mathrm{max}}$ than the flight path angle. Hence increasing altitude would not contract the flight envelope as much as increasing flight path angle does. Again, a more contracted maneuvering flight envelope is more prone to loss of control due to a secondary failure.

In the case of combinatory failures such as an impaired aircraft with damaged rudder and elevator, a comparison between the Sobol indices of rudder and elevator deflection limits reveals which one is less effective in changing the maneuvering flight envelope, so that an objective for the path planning could be chosen that puts the control efforts mainly on this less effective input factor. This way the flight envelope would not degrade too much in the case of a secondary failure of that control surface.

V. CONCLUSIONS

In this research, a comparison has been made between the generalization capabilities of ANNs and LMNs in the aircraft nonlinear dynamics modeling. To do so, maneuvering flight envelopes were evaluated for a number of rudder failure degrees based on a high-fidelity nonlinear transport model. Flight conditions and failure degrees for which the flight envelopes were evaluated were organized and prepared as the input data whereas and the values of V_{min} and V_{max} of the evaluated flight envelopes were considered as the target data. Multiple ANNs were trained using Bayesian regularization and Levenberg-Marquardt algorithms. Also, multiple LMNs were created based on linear, full quadratic, and sparse quadratic models and trained by LOLIMOT and HILOMOT algorithms. Performance of the trained ANNs and LMNs were evaluated on an independent test data and the results show that both network types have good accuracy; however ANN generalizes better to the new data. Also, the effect of different reductions in the number of training samples was investigated on the networks' performance. 5 diminished training datasets were constructed by reducing the number of samples of the input parameters (h, γ, δ_r) in different ways. According to the results, the networks' performance is not solely dependent on the number of training data and a good compromise between the amount of training data and networks' generalization could be obtained if an appropriate reduction in the training samples is implemented. Finally, an ANN-based global sensitivity analysis approach was proposed which utilizes the trained network's regression equation as an emulator for fast model evaluations required by precise sensitivity analysis methods such as variance-based method. The proposed approach can be used to choose the safer path planning strategies of an impaired aircraft.

Two alternatives to ANN are Support Vector Machine (SVM) and GP (Gaussian Process) which generally tend to have better generalization than ANN. However, ANNs were used in this research due to faster setup process and learning, the capability of predicting more than one output, and the possibility of online training which could be used in future works concerning real-time path planning of the impaired aircraft. For the case studies of this research the generalizations of the trained ANNs were accurate enough; however, in the case of structural failures were the number of inputs increases, the network's size grows and the ANN's generalization could degrade drastically. In such cases utilizing SVM or GP could provide better generalization. Hence, it is an interesting topic for future researches to investigate replacing ANN with SVM for failure cases with higher number of inputs.

APPENDIX

In order to find the number of required hidden neurons, neurons were added to the hidden layer one by one up to 20 neurons. Network's performance for each number of hidden neurons was estimated, and the process was iterated 10 times. As shown in Fig. 41, results show that the network's performance improves significantly by increasing the number of neurons from 1 to 4. Increasing the number of neurons from 5 to 9 slightly improves the network's performance however the value of MSE fluctuates considerably. From 10 hidden neurons all 10 iterations have close MSEs and increasing

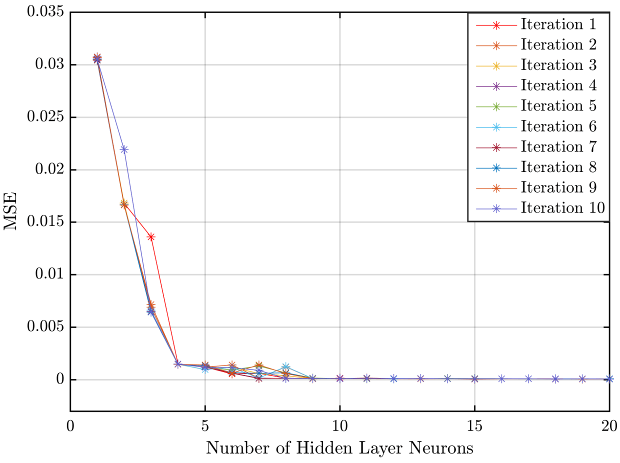


FIGURE 41. Variation of MSE by the number of hidden layer's neurons.

the number of hidden neurons does not improve the network's performance anymore. Hence, 10 hidden neurons are selected.

REFERENCES

[1] *Statistical Summary of Commercial Jet Airplane Accidents Worldwide Operations | 1959–2014*, Boeing Commercial Airplanes, Seattle, WA, USA, 2015.

[2] TSO, Norwich, U.K. (2013). *Global Fatal Accident Review 2002–2011*. Accessed: Jul. 20, 2018. [Online]. Available: http://publicapps.caa.co.uk/cap1036

[3] J. Wilborn and J. Foster, "Defining commercial transport loss-of-control: A quantitative approach," in *Proc. AIAA Atmos. Flight Mech. Conf. Exhib.*, Aug. 2004, pp. 16–19.

[4] M. J. Strube, "Post-failure trajectory planning from feasible trim state sequences," M.S. thesis, Dept. Aerospace Eng. Univ. Maryland, College Park, MD, USA, 2005.

[5] F. A. Di Donato, S. Balachandran, K. McDonough, E. Atkins, and I. Kolmanovsky, "Envelope-aware flight management for loss of control prevention given rudder jam," *J. Guid., Control, Dyn.*, vol. 40, no. 4, pp. 1027–1041, Apr. 2017. doi: 10.2514/1.G000252

[6] L. Tang, M. Roemer, J. Ge, A. Crassidis, J. Prasad, and C. Belcastro, "Methodologies for adaptive flight envelope estimation and protection," in *Proc. AIAA Guid., Navigat., Control Conf.*, Aug. 2009, pp. 10–13. doi: 10.2514/6.2009-6260

[7] Y. Zhang, C. C. de Visser, and Q. P. Chu, "Online aircraft damage case identification and classification for database information retrieval," in *Proc. AIAA Atmos. Flight Mech. Conf.*, 2018. [Online]. Available: https://www.icas.org/ICAS_ARCHIVE/ICAS2014/data/papers/2014_0555_paper.pdf. doi: 10.2514/6.2018-1020

[8] K. McDonough and I. Kolmanovsky, "Fast computable recoverable sets and their use for aircraft loss-of-control handling," *J. Guid., Control, Dyn.*, vol. 40, no. 4, pp. 934–947, Apr. 2017. doi: 10.2514/1.G001747

[9] T. Lombaerts, S. Schuet, D. Acosta, J. Kaneshige, K. Shish, and L. Martin, "Piloted simulator evaluation of safe flight envelope display indicators for loss of control avoidance," *J. Guid., Control, Dyn*, vol. 40, no. 4, pp. 948–963, Apr. 2017. doi: 10.2514/1.G001740

[10] P. K. Menon, P. Sengupta, S. Vaddi, B.-J. Yang, and J. Kwan, "Impaired aircraft performance envelope estimation," *J. Aircr.*, vol. 50, no. 2, pp. 410–424, Mar. 2013. doi: 10.2514/1.C031847

[11] D. I. Ignatyev and A. N. Khrabrov, "Neural network modeling of unsteady aerodynamic characteristics at high angles of attack," *Aerosp. Sci. Technol.*, vol. 41, pp. 106–115, Feb. 2015. doi: 10.1016/j.ast.2014.12.017

[12] S. A. Bagherzadeh, "Nonlinear aircraft system identification using artificial neural networks enhanced by empirical mode decomposition," *Aerosp. Sci. Technol.*, vol. 75, pp. 155–171, Apr. 2018. doi: 10.1016/j.ast.2018.01.004

[13] Y. Lyu, W. Zhang, J. Shi, X. Qu, and Y. Cao, "Unsteady aerodynamic modeling of biaxial coupled oscillation based on improved ELM," *Aerosp. Sci. Technol.*, vol. 60, pp. 58–67, Jan. 2017. doi: 10.1016/j.ast.2016.10.029

[14] J. Kou and W. Zhang, "Multi-kernel neural networks for nonlinear unsteady aerodynamic reduced-order modeling," *Aerosp. Sci. Technol.*, vol. 67, pp. 309–326, Aug. 2017. doi: 10.1016/j.ast.2017.04.017

[15] M. Winter and C. Breitsamter, "Efficient modeling of generalized aerodynamic forces across mach regimes using neuro-fuzzy approaches," in *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*. Cham, Switzerland: Springer, 2016, pp. 467–477. doi: 10.1007/978-3-319-27279-5_41

[16] S. Seher-Weiss, "Identification of nonlinear aerodynamic derivatives using classical and extended local model networks," *Aerosp. Sci. Technol.*, vol. 15, no. 1, pp. 33–44, Jan./Feb. 2011. doi: 10.1016/j.ast.2010.06.002

[17] S. Weiss, F. Thielecke, and H. Harders, *Ein Neuer Ansatz zur Modellierung von Luftdatensystemen* (A New Approach to Modeling Air Data Systems), (in German). Berlin, Germany: DGLR Jahrestagung, 1999.

[18] P. Giesemann and F. Thielecke, "Prediction of gas concentrations and temperature for an experimental combustion plant with local model networks," in *Proc. Neural Netw. Appl.*, Magdeburg, Germany, Mar. 1999, pp. 199–206.

[19] S. F. Bokhari and O. von Estorff, *Predicting the Aeroacoustic Behavior of Aircraft Air-Distribution System by Using Neurofuzzy Local Model Networks*. Bonn, Germany: German Aerospace Society, 2012.

[20] M. Halle and F. Thielecke, "Flight loads estimation using local model networks," in *Proc. 29th Congr. Int. Council Aeronautical Sci.*, St. Petersburg, Russia, 2014.

[21] R. Norouzi, A. Kosari, and M. H. Sabour, "A comparison between generalization capability of neural network and neuro-fuzzy system in nonlinear dynamics identification of impaired aircraft," in *Proc. IEEE 9th Annu. Inf. Technol., Electron. Mobile Commun. Conf. (IEMCON)*, Nov. 2018, pp. 687–693. doi: 10.1109/IEMCON.2018.8614987

[22] Y. Tang, E. Atkins, and R. Sanner, "Emergency flight planning for a generalized transport aircraft with left wing damage," in *Proc. AIAA Guid., Navigat. Control Conf. Exhib.*, Aug. 2007, pp. 20–23.

[23] B. Stevens, F. Lewis, and E. Johnson, *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*, 3rd ed. Hoboken, NJ, USA: Wiley, 2016.

[24] C. Edwards, T. Lombaerts, and H. Smaili, *Fault Tolerant Flight Control*. Berlin, Germany: Springer, 2010.

[25] T. P. Vogl, J. K. Mangis, A. K. Rigler, W. T. Zink, and D. L. Alkon, "Accelerating the convergence of the back-propagation method," *Biological*, vol. 59, nos. 4–5, pp. 257–263, Sep. 1988. doi: 10.1007/BF00332914

[26] F. Dan Foresee and M. T. Hagan, "Gauss-Newton approximation to Bayesian learning," in *Proc. Int. Conf. Neural Netw.*, Jun. 1997, pp. 1930–1935.

[27] D. J. C. MacKay, "Bayesian interpolation," *Neural Comput.*, vol. 4, no. 3, pp. 415–447, 1992.

[28] T. Fischer and O. Nelles, "Merging strategy for local model networks based on the lolimot algorithm," in *Artificial Neural Networks and Machine Learning—ICANN*. Cham, Switzerland: Springer, 2014, pp. 153–160.

[29] B. Hartmann and T. Ebert, "LMNtool—Toolbox zum automatischen trainieren lokaler modellnetze," in *Proc. 22nd Workshop Comput. Intell.*, Dortmund, Germany, Dec. 2012, pp. 341–355.

[30] O. Nelles, "Axes-oblique partitioning strategies for local model networks," in *Proc. IEEE Conf. Comput. Aided Control Syst. Design*, Oct. 2006, pp. 2378–2383.

[31] O. Nelles, *Nonlinear System Identification*. Berlin, Germany: Springer, 2001.

[32] J. Foster *et al.*, "Dynamics modeling and simulation of large transport airplanes in upset conditions," in *Proc. AIAA Guid., Navigat., Control Conf. Exhib.*, Aug. 2005, pp. 15–18.

[33] *GTM_DesignSim*. Accessed: May 10, 2018. [Online]. Available: https://github.com/nasa/GTM_DesignSim

[34] G. Yi and E. Atkins, "Trim State discovery for an adaptive flight planner," in *Proc. 48th AIAA Aerosp. Sci. Meeting Including New Horizons Aerosp. Expo.*, Jan. 2010, pp. 4–7.

[35] R. Norouzi, A. Kosari, and M. H. Sabour, "Data for: Maneuvering flight envelope evaluation and analysis of generic transport model with control surfaces failures," Univ. Tehran, Tehran, Iran, 2018. doi: 10.17632/k4ntmx43x5.1

[36] F. Pianosi *et al.*, "Sensitivity analysis of environmental models: A systematic review with practical workflow," *Environ. Model. Softw.*, vol. 79, pp. 214–232, May 2016. doi: 10.1016/j.envsoft.2016.02.008

[37] J. Morio, "Global and local sensitivity analysis methods for a physical system," *Eur. J. Phys.*, vol. 32, no. 6, pp. 1577–1583, Oct. 2011. doi: 10.1088/0143-0807/32/6/011

[38] M. D. Morris, "Factorial sampling plans for preliminary computational experiments," *Technometrics*, vol. 33, no. 2, pp. 161–174, 1991.

[39] A. Saltelli *et al.*, *Global Sensitivity Analysis: The Primer*. Hoboken, NJ, USA: Wiley, 2016.

[40] A. van Griensven, T. Meixner, S. Grunwald, T. Bishop, M. Diluzio, and R. Srinivasan, "A global sensitivity analysis tool for the parameters of multi-variable catchment models," *J. Hydrol.*, vol. 324, nos. 1–4, pp. 10–23, Jun. 2006. doi: 10.1016/j.jhydrol.2005.09.008

[41] S. Burhenne, D. Jacob, and G. P. Henze, "Sampling based on Sobol'sequences for Monte Carlo techniques applied to building simulations," in *Proc. 12th Conf. Int. Building Perform. Simulation Assoc.*, Sydney, NSW, Australia, Nov. 2011, pp. 1816–1823.

[42] F. Campolongo, A. Saltelli, and J. Cariboni, "From screening to quantitative sensitivity analysis. A unified approach," *Comput. Phys. Commun.*, vol. 182, no. 4, pp. 978–988, Apr. 2011. doi: 10.1016/j.cpc.2010.12.039

[43] S. Kucherenko, S. Tarantola, and P. Annoni, "Estimation of global sensitivity indices for models with dependent variables," *Comput. Phys. Commun.*, vol. 183, no. 4, pp. 937–946, Apr. 2012. doi: 10.1016/j.cpc.2011.12.020

[44] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, "Shape distributions," *ACM Trans. Graph.*, vol. 21, no. 4, pp. 807–832, Oct. 2002. doi: 10.1145/571647.571648

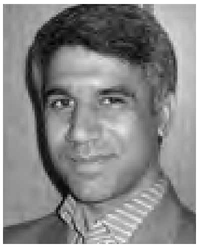
[45] F. Pianosi, F. Sarrazin, and T. Wagener, "A MATLAB toolbox for global sensitivity analysis," *Environ. Model. Softw.*, vol. 70, pp. 80–85, Aug. 2015. doi: 10.1016/j.envsoft.2015.04.009

RAMIN NOROUZI (M'17) received the B.S. and M.S. degrees in aerospace engineering from the Sharif University of Technology, Tehran, Iran, in 2010 and 2012, respectively. He is currently pursuing the Ph.D. degree in aerospace engineering with the University of Tehran, Iran.

His research interests include trajectory optimization, terrain following and terrain avoidance, optimal control, and impaired aircraft dynamics evaluation using neural networks and neuro-fuzzy systems.

AMIRREZA KOSARI (M'17) received the B.S. degree in aerospace engineering from Amirkabir University, Tehran, Iran, in 1998, and the M.S. and Ph.D. degrees in aerospace engineering from the Sharif University of Technology, Tehran, in 2001 and 2008, respectively.

From 2010 to 2016, he was an Assistant Professor with the Faculty of New Sciences and Technologies, University of Tehran, Iran, where he has been an Associate Professor, since 2017. His research interests include trajectory optimization, optimal control, cooperative flights, and spacecraft attitude control. He has authored several papers in these fields.



MOHAMMAD HOSSEIN SABOUR received the B.S. and M.S. degrees in mechanical engineering from the University of Tehran, Iran, in 1990 and 1993, respectively, and the Ph.D. degree in aerospace engineering from Concordia University, Montreal, QC, Canada, in 2005.

He has three years of work experience at Pratt and Whitney, Canada, and more than 30 years of work experience in aerospace, railway, and marine areas in Iran. He has also the experience of five years teaching at Concordia University, Canada, and 23 years teaching of mechanical and aerospace engineering courses at different Iranian universities. Since 2010, he has been an Assistant Professor with the Faculty of New Sciences and Technologies, University of Tehran. He has authored ten books, 21 journal articles, and 40 conference papers in fracture, control, fuzzy, entrepreneurship, and strategic management areas. He has also authored three entries in Springer Encyclopedia of Tribology (NY, USA, Springer, 2013).