Experimental Analysis of Neural Approaches for Synthetic Angle-of-Attack Estimation

Hindawi
International Journal of Aerospace Engineering
Volume 2021, Article ID 9982722, 13 pages
https://doi.org/10.1155/2021/9982722

Research Article

Experimental Analysis of Neural Approaches for Synthetic Angle-of-Attack Estimation

Angelo Lerro,[1] **Piero Gili**,[1] **Mario Luca Fravolini**,[2] **and Marcello Napolitano**[3]

[1]*Department of Mechanical and Aerospace Engineering, Polytechnic University of Turin, C.so Duca degli Abruzzi 24, Turin 10129, Italy*
[2]*Department of Electronic and Information Engineering, University of Perugia, Via G. Duranti 93, Perugia 06125, Italy*
[3]*Department of Mechanical and Aerospace Engineering, West Virginia University Morgantown, P.O. Box 6106, Morgantown, WV 26506, USA*

Correspondence should be addressed to Angelo Lerro; angelo.lerro@polito.it

Received 31 March 2021; Accepted 20 May 2021; Published 10 July 2021

Academic Editor: Erkan Kayacan

Synthetic sensors enable flight data estimation without devoted physical sensors. Within modern digital avionics, synthetic sensors can be implemented and used for several purposes such as analytical redundancy or monitoring functions. The angle of attack, measured at air data system level, can be estimated using synthetic sensors exploiting several solutions, e.g., model-based, data-driven, and model-free state observers. In the class of data-driven observers, multilayer perceptron neural networks are widely used to approximate the input-output mapping angle-of-attack function. Dealing with experimental flight test data, the multilayer perceptron can provide reliable estimation even though some issues can arise from noisy, sparse, and unbalanced training domain. An alternative is offered by regularization networks, such as radial basis function, to cope with training domain based on real flight data. The present work's objective is to evaluate performances of a single-layer feed-forward generalized radial basis function network for AoA estimation trained with a sequential algorithm. The proposed analysis is performed comparing results obtained using a multilayer perceptron network adopting the same training and validation data.

1. Introduction

Next-generation commercial aviation may use synthetic sensors (SS) for safety-critical operations [1] in addition or replacing devoted physical sensors. A synthetic sensor is mainly a state observer, or estimator, able to fuse together flight data available on the avionic bus aiming to estimate other flight parameters. As far as the air data system (ADS) is concerned, synthetic sensors can also be used as a means of mitigation to overcome some issues towards certification for unmanned aerial vehicles (UAVs) [2] and urban air mobility (UAM) aircraft [3].

State-of-the-art angle-of-attack (AoA) physical sensors are typically vanes (or multihole probes) protruding externally from the aircraft fuselage able to provide a direct measure of the flow angle.

Digital avionic solutions, e.g., fly-by-wire, enable AoA synthetic sensor implementation along with physical (or mechanical) sensors in order to analytically increase the system redundancy [4–6]. Another possible application is to use synthetic sensors to monitor physical sensors and to accommodate possible failures [7, 8]. Moreover, the concurrent use of dissimilar sources of the same air data (physical and synthetic ones) can be beneficial to solve some issues related to common failure modes or incorrect failure diagnosis of modern air data system [9, 10].

Synthetic sensors can be grouped into three classes: (1) model-based (e.g., Kalman filter) [11]; (2) data-driven (e.g., neural networks) [12]; and model-free [13, 14]. A comparison between these approaches can be found in [15]. Working with neural networks, training dataset based on experimental flight tests can be characterised by noisy data, uncovered (i.e., low density), or overpopulated (i.e., high density) areas of the flight envelope.

These aspects can lead to common issues of data concentration (nonuniform density) and unbalanced (or sparse)

hypercube where the neural network (NN) is defined. Although the multilayer perceptron is widely used for AoA estimation, this kind of network suffers from sparse domain with nonuniform density. In fact, in [16], it emerged that a "modified" ad hoc training dataset is necessary to achieve an acceptable level of AoA estimation accuracy. The "modified" training dataset is obtained using suitable data preprocessing techniques briefly discussed in Section 3.

In the present work, in order to avoid modifying the training dataset accordingly to the specific aircraft application, a "local" approximator is chosen for the intrinsic capability to better tolerate sparse domain (where the NN is defined) with respect to the "global" approximators (e.g., multilayer perceptron (MLP)). Moreover, the overpopulated areas work as "attractors" for batch training algorithms, whereas nonuniform density domains should be better tolerated by sequential training strategies. Among "local" approximators, the generalized radial basis function (GRBF) networks have shown to be very effective for (online) sequential learning [17] and, hence, sequentially trained GRBF are considered in this paper as alternative to batch-trained MLP. Considering the same training and validation databases, the objective of the present work is to compare GRBF and MLP performances in order to assess the approach to be used in operative scenarios for AoA estimation.

Training and test manoeuvres are extracted from experimental flight trials as described in Section 3, whereas the rationale behind the neural approaches is detailed in Section 2. The approach used for angel-of-attack estimation is described in Section 4, and a brief overview of previous works is described in Section 5. The GRBF-NN is introduced in Section 6 with a particular focus on parameter setup to select the most suitable GRBF-NN architecture to be applied to the present input-output mapping defined on experimental flight data. A comparison analysis between the SS-MLP and SS-GRBF is proposed in Section 7 before concluding the work.

2. Description of Neural Approaches

In [18], it is demonstrated that multilayer perceptron (MLP) neural networks with a single hidden layer and sigmoidal activation functions can approximate any continuous nonlinear input-output mapping function. In [19], it is demonstrated that a regularization network (such as the radial basis function (RBF) and generalized RBF (GRBF)) with a single hidden layer, radial activation functions, and constant smoothing factors can approximate any continuous nonlinear input-output mapping function. Given an input-output mapping function, the accuracy of MLP and GRBF approximations cannot be defined a priori as it depends on several considerations; the more relevant considerations for the neural approaches are discussed in this section.

The MLP has the ability to construct "global" approximations to nonlinear input-output mapping, whereas GRBF construct "local" approximations to nonlinear input-output mapping. The activation function of MLP belongs to ridge function class (e.g., Equation (4)), whereas the GRBF activation functions are classified as radial ones (e.g., Equation (5)) leading to regularization networks. The latter aspects can be observed in the hidden unit behaviour. In fact, the argument of any MLP's activation functions is the inner product of the input vector and the synaptic weight vector. On the other hand, the argument of the activation function of a single GRBF's neuron computes the Euclidean norm (or distance) between the input vector and the center vector (i.e., each center is dedicated to a specific input) of that hidden unit.

In [20], it is shown that the MLP can perform the nonlinear function approximation with fewer parameters of the RBF neural network for the same degree of accuracy. This is due to "global" characteristics of the MLP. In [21], the relationship between the MLP and the GRBF is demonstrated. Under some hypothesis, the MLP can approximate the GRBF with the same number of hidden units, the MLP's synaptic weights and biases replace the centers of the GRBF, and, hence, the GRBF's "local" representation of the input domain is devoted to the MLP's synaptic weights. The vice versa is only possible under more strict hypothesis; i.e., the GRBF cannot always approximate the MLP with the same number of neurons.

Even though MLP's "global" approximations can be demonstrated to be more powerful than the GRBF's "local" approximation with the same number of hidden neurons [21], the choice between the MLP and GRBF cannot be done a priori. In fact, experiments to prove the aforementioned connection between the MLP and GRBF are carried out on uniformly distributed input domains. This latter aspect is not very common dealing with aircraft flight tests that often leads to noisy and sparse definition domain. In [22, 23], the problem of sparse, or unbalanced, input domain, i.e., the density of the training data is lower if compared to other areas of the input domain, is discussed. Under this hypothesis, it is shown that the MLP has less approximation capabilities with respect to the GRBF. Another important aspect is the training dataset used to approximate the nonlinear input-output mapping function. In [24], the effect of input noise is studied for MLP, and in [25], the immunity to input noise is assessed for GRBF networks.

As far as NN applications for flow angle estimation are concerned, the first example of NNs used for flow angle estimation without using dedicated physical sensors (e.g., vane and distributed flush ports) can be found in [26] where a nonlinear autoregressive exogenous (NARX) technology is used to estimate noise-free AoA. Later, noisy flight data are used to train and validate NNs in [27] with a single-layer time-delay NN that is demonstrated to be effective if at least two past values of the same input series are considered, whereas in [28], single-layer feed-forward MLP-NN shows larger errors if past values are neglected. In [29], the AoA estimation is based on a single-layer feed-forward MLP-NN exploiting a patented approach [30] that is also used for the present work.

In Figure 1, training and validation stages are described within the external loop necessary for the trade-off parameter for MLP and GRBF networks.

As preliminary activity, training and validation data are selected from the entire flight test database as described in Section 3. Once the network type is selected, several NNs are defined considering several numbers of neurons. All

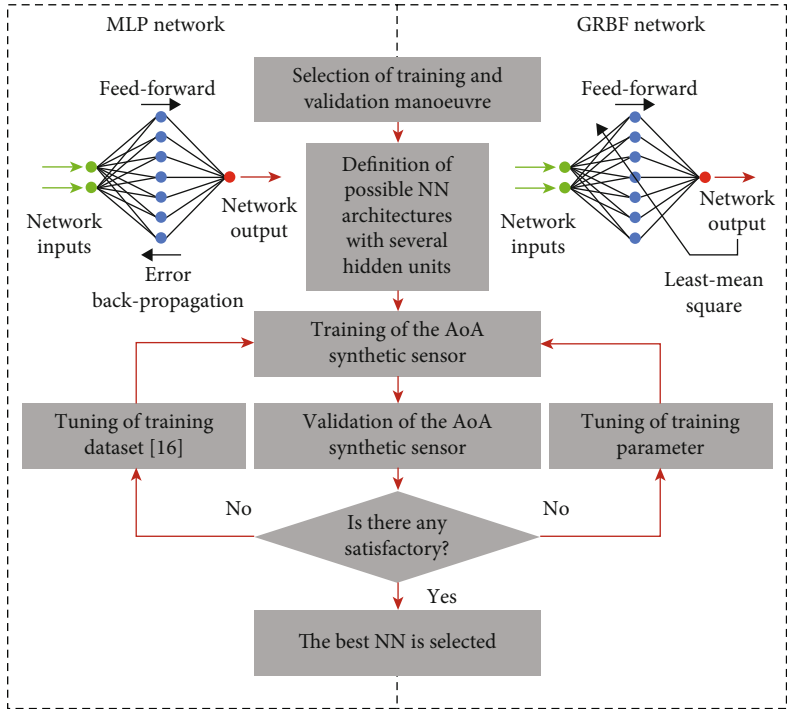


Figure 1: AoA synthetic sensor trade-off: flow charts for MLP and GRBF networks.

networks that satisfy performance requirements (defined in Section 4.1) are compared, and the best is selected according to the criterion introduced in Section 6.1.

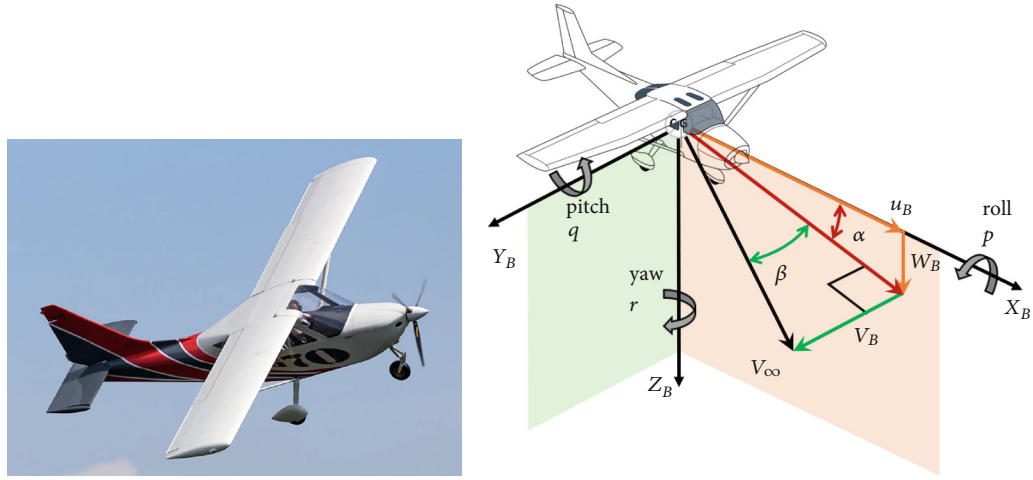
If no network is able to satisfy the performance requirements, MLP and GRBF require different analyses. As far as the MLP is concerned, a possible solution is given in [16] where the training dataset is "modified" ad hoc and the trade-off is repeated. The same approach cannot be adopted for GRBF networks trained sequentially where the training algorithm [31] expects to process continuous data. Therefore, rather than "modified" training data, the GRBF trade-off is based on training parameter tuning. Moreover, dealing with experimental flight data, training data manipulations are not always straightforward and they should fit the specific aircraft application. In fact, as described in Section 5, the MLP required several actions until satisfactory performances are achieved.

3. Training and Validation Database Description

The flight test database is populated by data collected during a flight test campaign conducted in the north of Italy during certification flight trials of the ULM aircraft G70 Figure 2(a). The G70 is a propeller-driven aircraft with traditional wing-tail configuration, 2 seats, and nonretractable landing gear. A fully fledged flight test instrumentation (FTI) suite is installed onboard [32], and it is capable of supporting certification procedures. A second FTI used for synthetic sensor implementation is installed onboard that is equipped with an independent ADS and attitude and inertial reference system (AHRS).

From the whole flight test database, suitable manoeuvres, or records, are chosen for the learning and the validation stages of the synthetic sensor as reported in Tables 1 and 2, respectively. The objective of the training dataset is to cover the widest area of the aircraft flight envelope by means of exciting as much as possible one dynamic mode at a time, whereas the test dataset is aimed at collecting manoeuvres that are not represented in the training dataset, e.g., coupling aircraft modes. As can be seen from Figure 3(c), the training was collected within the normal operative range (i.e., from the stall speed without flaps V_{S1} to maximum normal operative airspeed V_{NO}), whereas some points chosen for the validation stage exceed the V_{NO} (i.e., the yellow areas) due to high dynamics involved in the manoeuvres. Therefore, these points are acceptable exceptions for the present application. In fact, the same airspeed is already considered in the AoA estimation with Equation (2); therefore, the dynamic pressure exceedance is not recognised as critical. Moreover, the proposed view of Figure 3(c) is only limited to a single input variable (q_c or CAS) out of seven.

Validation manoeuvres are selected from the available flight test database that are not included in the training dataset as can be seen from Table 2. Moreover, in [16], in order to overcome domain's low-density areas, artificial points are considered. As the lack of points is noted during a steady-state flight condition, the flight test database is artificially augmented using one hundred (100) points and collected under the "flight test no. 8" that here is only used for validation purposes. The artificial steady-state points are calculated using the G70 aerodynamic model identified with flight tests as described in [16]. These artificial points are calculated simulating steady-state flight conditions in the operative AoA range between 0° (corresponding to maximum airspeed in turbulent



(a) (b)

Figure 2: Flight test aircraft (a) G70 Aircraft from Ing. Nando Groppo S.r.l. and (b) body reference frame with aerodynamic angles (α, β), linear relative velocities (u_B, v_B, w_B), and angular rates (p, q, r).

Table 1: Description of training manoeuvres.

Flight test no.	Date and time	Total time	Description
1	10th June 2017 8:50	2220	Sawtooth glide, pitch sweep
2	10th June 2017 9:45	2000	Sawtooth glide, dutch roll
3	10th June 2017 14:35	420	Phugoid (stick fixed and free)
4	10th June 2017 16:41	480	Steady heading sideslip

Table 2: Description of validation manoeuvres.

Flight test no.	Date and time	Total time	Description
5	10th June 2017 14:30	580	Sawtooth glide
6	10th June 2017 15:37	1900	Sawtooth glide, Phugoid (stick fixed and free)
7	11th June 2017 16:35	900	Sawtooth glide
8	N/A	N/A	Simulated steady-state flight condition [16]

air) and 12° (corresponding to the controlled low speed conditions) as it would be measured by the Pitot boom. The input-output mapping related to artificial AoA values is characterised by null inputs (and consequently included in the training boundaries) except for the pitch angle ($\theta = \alpha$) and the airspeed (or $q_c = 1/2\rho_\infty V_\infty = (2W/S)/C_{L,\alpha}\alpha$) that is calculated considering a mean weight. It can be observed that they are within the training boundary in the plane CAS-AoA of Figure 3(c).

Once all data are selected, all variables contained in the input/output training vectors are normalised between ±1 considering minimum and maximum values of the training. The hypercube where the NN is defined is represented with the box plot method [33]. For each single box, the central mark indicates the median and the bottom and top edges of the box indicate the 25th and 75th percentiles, respectively. The whiskers (dashed vertical lines) cover to the most extreme data points not considered outliers whereas the outliers are plotted using the + symbol.

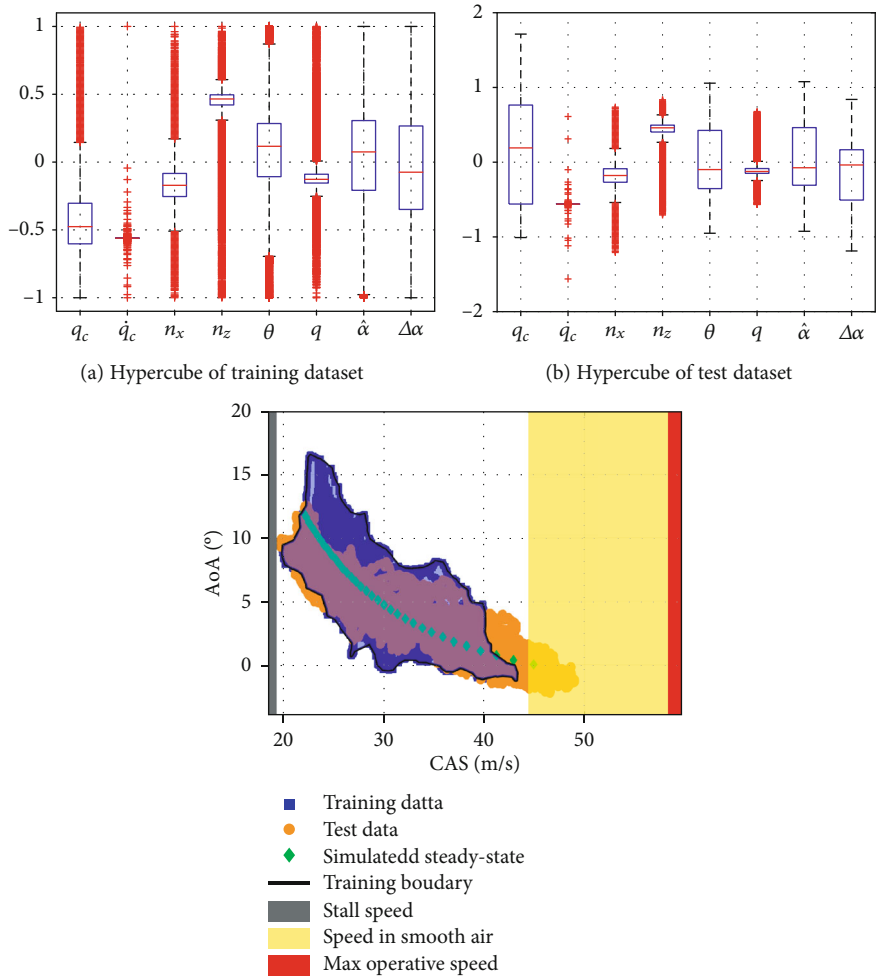
From analysis of Figure 3(a), it is clear that the training domain is sparse (due to the presence of large number of outliers denoted with the + symbol) and it is not uniformly covered. This characteristic is intrinsic of the flight test database as it is practically impossible to fly all possible input data combinations. Moreover, in order to achieve a uniform hypercube, all training manoeuvres should be repeated for all possible aircraft weight and balance configurations that again is practically not feasible. The flight tests considered in this work are flown with several weight and balance configurations, and this aspect is mitigated by the proposed approach that considers a preliminary kinematic AoA evaluation as in Equation (2).

Data distribution shown in Figure 3(b) demonstrates that most of the flight test data is included in the training domain, whereas, as said before, some test points lay outside the training perimeter, such as the dynamic pressure or the calibrated airspeed (CAS).

4. Proposed Approach for AoA Estimation

With the patent [30], the nonlinear mapping between input and output is proposed as follows:

$$\alpha_{SS} = \widehat{\alpha} + \Delta\alpha, \tag{1}$$



(a) Hypercube of training dataset

(b) Hypercube of test dataset

(c) Flight database on the AoA-CAS plane. The red, yellow, and black areas represent the maximum operative, calm air, and stall boundaries as shown on the speed indicator

Figure 3: Comparison between training and test datasets.

where functional dependencies are split into two contributions $\widehat{\alpha}$ and $\Delta\alpha$. A first estimation $\widehat{\alpha}$ is obtained with levelled flight equations [34], whereas $\Delta\alpha$ is the difference between the first estimation and the true value, $\widehat{\alpha} - \alpha$. From kinematic considerations, the initial estimation $\widehat{\alpha}$ is evaluated as follows:

$$\widehat{\alpha} = \theta - \gamma = \theta - \arcsin \frac{V_D}{V_\infty}, \tag{2}$$

where θ is the pitch angle, γ is the flight path angle, V_D is the down velocity in the inertial reference system (or GNSS), and V_∞ is the true airspeed. For low and constant altitudes, as in the present application, the V_∞ can be substituted with the CAS avoiding the onboard temperature measure.

The correction $\Delta\alpha$ proposed in [35] is based on NN using the feed-forward approximator described as

$$\Delta\alpha = f_\alpha(q_c, \dot{q}_c, n_x, n_z, \theta, q, \widehat{\alpha}), \tag{3}$$

where q_c is the impact pressure (defined as the difference between the total and static pressure); $\dot{q}_c$ is the time derivative of q_c; n_x, n_y, and n_z are the inertial (or proper) accelerations measured by the AHRS, respectively, along X_{Body}, Y_{Body}, and Z_{Body} axes; and q is the pitch rate. A possible implementation of the AoA synthetic estimation is represented in Figure 4, where data from the GNSS, ADS, and AHRS are required.

4.1. Error Requirements. The estimation error is calculated as the difference between the estimated AoA (α_{SS}) and the true AoA (α). To accommodate future applications of alternative solutions for flow angle estimations, a working group is defining the new standard AS7984 "Minimum Performance Standards, for Angle of Attack (AoA) and Angle of Sideslip (AoS)" to cover the various sensor technologies used to measure flow angles that provide relevant output to other aircraft safety-critical systems [36]. In this work, the AoA estimation targets are qualitatively established as follows:

(i) Steady-state mean error < 0.5°

(ii) Steady-state max error < 1.0°

(iii) Dynamic 2σerror < 1.5°

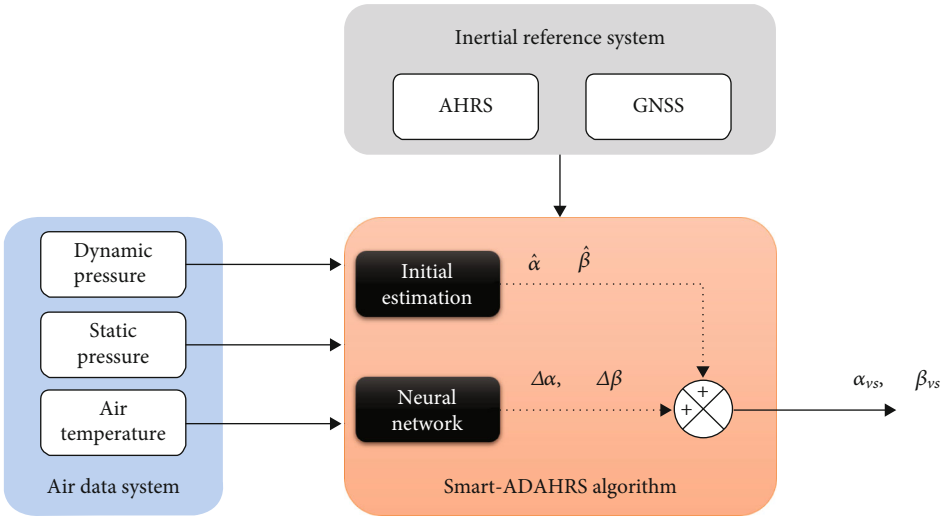


Figure 4: General schematic of the AoA estimation using the approach proposed in Equation (1).

(iv) Dynamic max error < 5.0°

5. AoA Synthetic Sensor Based on MLP-NN

The MLP-NN trade-off is presented in [29] leading to a NN architecture with a single hidden layer with 13 neurons and one linear output layer. The activation functions is a sigmoid function:

$$f_j(\mathbf{x}) = \frac{2}{1 + e^{-\mathbf{x}}} - 1, \tag{4}$$

where f_j is related to the *jth* neuron processing all inputs $\mathbf{x}$.

During the training stage, neural network's parameters are estimated solving the nonconvex problem of the error function optimization. Different heuristic rules exist, and the Levenberg-Marquardt algorithm is considered [16]. The MLP-NN is trained using the batch back-propagation algorithm that makes the MLP-NN insensitive to the order in which the data are presented. A single training is performed in about 20 on a personal computer.

Even though the batch-trained MLP-NNs have shown to be very effective for offline learning dealing with flow angle estimation [37], it also emerged that the AoA estimation suffers from two main issues: (1) overfitting and (2) sparse hypercube.

A chance to mitigate the unbalanced definition hypercube, or data concentration, can rely on pruning similar flight data in order to avoid high-density areas in the hypercube as shown in [16]. On the other side, it is not always possible to cover the entire hypercube with a flight test campaign. To solve the latter issue, an augmentation approach was proposed in [16] by means of introducing specific simulated flight test points to populate low-density areas of the training hypercube. Both preprocessing approaches lead to a "modified" training dataset that is useful to increase the MLP-NN performances trained with a batch algorithm.

Table 3 collects results obtained using MLP-NNs from [16, 29]. Results of the SS based on MLP and trained with the entire flight data records are labelled with SS-MLP, whereas the SS-MLP-M indicates results obtained with the same NN architecture but trained with the "modified" training dataset. From error analysis on the flight tests of Table 3, it is clear that the SS-MLP shows larger errors with respect to SS-MLP-M that, instead, has acceptable performances if compared to those required in Section 4.1.

6. AoA Synthetic Sensor Based on GRBF-NN

The AoA estimator proposed with the present work belongs to the class of (growing) generalized radial basis function neural networks (GRBF-NNs) trained with a sequential algorithm. As discussed in Section 2, GRBF are better approximators than MLP with sparse definition domain and the sequential training algorithm should avoid overfitting due to domain's overpopulated areas. Therefore, dealing with common issues of real flight test data for flow angle estimation, the GRBF pretrained sequentially are expected to perform as the MLP with the "modified" training dataset.

One of the crucial differences with respect to MLP is the GRBF's Gaussian activation functions that are able to form a local representation of the mapping function to be approximated. This aspect makes GRBF-NN more insensitive to data distribution on the input domain but the order used in the training is crucial as it is designed to work online. In fact, each basis function's output decays exponentially moving away from its center (or mean value) as can be seen from Equation (5). In a conventional Gaussian GRBF-NN, each *j*th neuron has the following activation function:

$$\phi_j(\mathbf{x}) = \exp\left(-\frac{\left\|\mathbf{x} - \boldsymbol{\mu}_j\right\|^2}{2\sigma_j^2}\right), \tag{5}$$

where $\mathbf{x}$ is the input vector to the network. The neurons are statically allocated on a uniform n-dimensional grid (where n is the dimension of the input vector $\boldsymbol{x}$) covering

Table 3: Estimation error characteristics for MLP and GRBF networks. The 2σ value is calculated as the error of the 95.45% of the data points.

Flight test no.	SS-MLP			SS-MLP-M			SS-GRBF		
	Mean	2σ	|Max|	Mean	2σ	|Max|	Mean	2σ	|Max|
1	−0.13°	1.06°	4.05°	−0.13°	1.48°	3.31°	0.03°	1.36°	4.73°
2	0.24°	1.17°	5.78°	0.21°	1.39°	4.44°	−0.41°	1.48°	5.12°
3	−0.18°	1.17°	2.53°	−0.15°	1.33°	1.83°	−0.13°	1.23°	4.20°
4	−0.24°	1.62°	3.59°	−0.29°	1.82°	3.35°	0.07°	2.06°	4.42°
5	−0.39°	1.99°	5.43°	−0.23°	1.51°	3.63°	−0.10°	1.43°	4.83°
6	−0.04°	1.28°	4.00°	0.10°	1.10°	3.33°	−0.39°	1.37°	4.26°
7	−0.50°	2.43°	5.19°	−0.30°	1.48°	4.02°	0.08°	1.80°	4.14°
8 (artificial)	−1.03°	N/A	1.37°	−0.41°	N/A	0.69°	0.01°	0.81°	0.95°

the region of interest for the input space. Therefore, the vector $\boldsymbol{\mu}$ contains n centers associated to the jth neuron. According to the universal approximation theorem for RBF [19], there is a single smoothing factor (or Gaussian width) σ associated with the jth neuron.

While constructive procedures for determining the center positions and the variances of the neurons are introduced in [38], the main problem of GRBF is that the total number of neurons grows exponentially with the input dimension. In order to avoid this problem for conventional GRBF architectures, a sequential learning technique for GRBF–NNs was proposed in [39], defined resource-allocating network (RAN), with emphasis on fast learning, good generalization, and compact representation.

The RAN growth strategy is based on three criteria listed here: (1) the current estimation error criteria $e(k) = y(k) - \widehat{y}(k) \geq E1$; 2) the novelty criteria $\|x(k) - \mu_j(k)\| \geq E2$; (3) the windowed mean error criteria $(1/N)\sum_{i=0}^{N} [y(k-N+i)\widehat{y}(k-N+i)] \geq E3$. When all three growth criteria are satisfied, a new neuron $(M+1)$ is added; otherwise, only the vector $\boldsymbol{\Theta}$ containing the tuning parameters is updated. When a new neuron is added, its center μ_{M+1}, variance σ_{M+1}, and weights $\mathbf{w}_{M+1}$ are updated accordingly to criteria defined in [40]. Here, it is worth underlying that when a new neuron is added, the new variance is initialised as

$$\boldsymbol{\Theta}(k+1) = \lambda \left\| x(k) - \mu_j(k) \right\|_{\infty}, \tag{6}$$

with $j \in [1, M]$ and λ is the "overlap" training parameter.

Least squares and gradient descent algorithms are commonly used [41–44] for online tuning of the network parameters. From previous experiences [31, 40], a gradient-based algorithm is used in the present work because of its lower computational effort. The online discrete time adaptation rule is given by

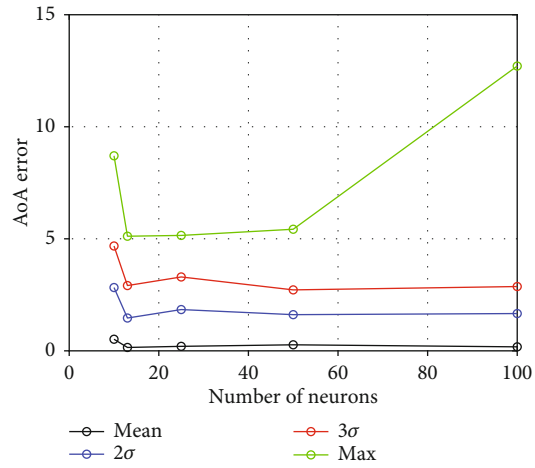
$$\boldsymbol{\Theta}(k+1) = \boldsymbol{\theta}(k) - \eta \left. \frac{\partial \widehat{y}}{\partial \boldsymbol{\theta}} \right|_k e(k), \tag{7}$$

where $e(k)$ is the prediction error and η is the learning rate. For a fully tuned RAN, the vector of parameters to be updated at each step is given by $\boldsymbol{\Theta} = [\mathbf{W}, \boldsymbol{\Pi}, \boldsymbol{\Sigma}]$ and $\mathbf{W}$ is the vector of the output weights, $\boldsymbol{\Pi}$ is the vector containing the positions (centers) of each neuron, and $\boldsymbol{\Sigma}$ is the vector of the variances for each neuron. Therefore, generally speaking, three learning rates can be adopted η_W, η_Σ, and η_Π, respectively, for $\mathbf{W}$, $\boldsymbol{\Pi}$, and $\boldsymbol{\Sigma}$. In order to avoid an excessive increase of the network size, a pruning strategy can also be applied when the defined maximum neuron number is reached. This modified architecture is called minimal RAN (MRAN) [31]. When a neuron is pruned, a new one is added following the same rules described before.

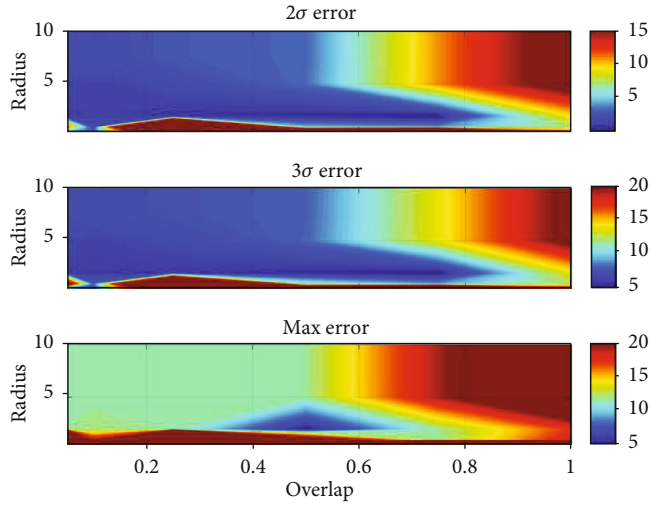
With the fully tuned extended MRAN (EMRAN), the growing and pruning mechanisms remain unchanged, while the parameters are updated following a "winner takes it all" strategy. In other words, only the parameters of the neurons within a defined "radius" are updated because they are considered the most active, while all the others are left unchanged. This strategy allows a significant reduction of the number of parameters to be updated online and a significant reduction of computational requirements with negligible performance degradation with respect to the MRAN [31]. Even though the EMRAN algorithm is designed for online training, it is worth underlying that in this work, the sequential algorithm is used to train the NN offline in order to have a pretrained NN onboard.

6.1. Training Parameter Setup and NN Downselection. The GRBF training strategy is inspired by [40] where a similar application was studied. The GRBF is trained with the EMRAN algorithm several times in order to achieve a satisfactory convergence of the error. In this work, the GRBF-NNs are retrained 250 times unless a convergence criteria, the training 2σ error $< 1.5°$, is reached. A single sequential training takes about 5 s on a personal computer and, therefore, up to 21 min for the 250 retrainings considered in this work. The learning rates are set to $\eta_w = 1 \times 10^{-2}$, $\eta_\sigma = 1 \times 10^{-3}$, and $\eta_\mu = 1 \times 10^{-3}$ because a faster convergence was noted. Moreover, the three error thresholds introduced in Section 6 are E1 = E2 = 0.75 and E3 = 0.35 to avoid a superfluous growth of hidden units from very early iterations, i.e., processing the first training points.

As far as the radius and overlap parameters are concerned, a priori values are not available and they depend on (1) the number of neurons and (2) the sampling rate of the training points. From a preliminary analysis, it emerged that values proposed in [40] would not be



(a) Best training performances with 10, 13, 25, 50, and 100 neurons adopting the best combination of radius and overlap



(b) 2σ, 3σ, and maximum errors as a function of the radius and overlap parameters for GRBF-NN with 13 neurons

Figure 5: Trade-off of number of neurons, radius, and overlap for GRBF-NN trained with the "original" dataset.

applicable to the present dataset and, therefore, a trade-off was necessary.

The candidate GRBF-NNs are evaluated considering maximum neuron numbers in the range [10,13,25,50,100], where 13 is suggested by the MLP architecture recalled in Section 5. Any GRBF-NN is retrained with radius $\in$ [0.5,1, 2, 5, 7.5,10] and overlap $\in$ [0.05,0.1,0.2,0.5,0.75,1].

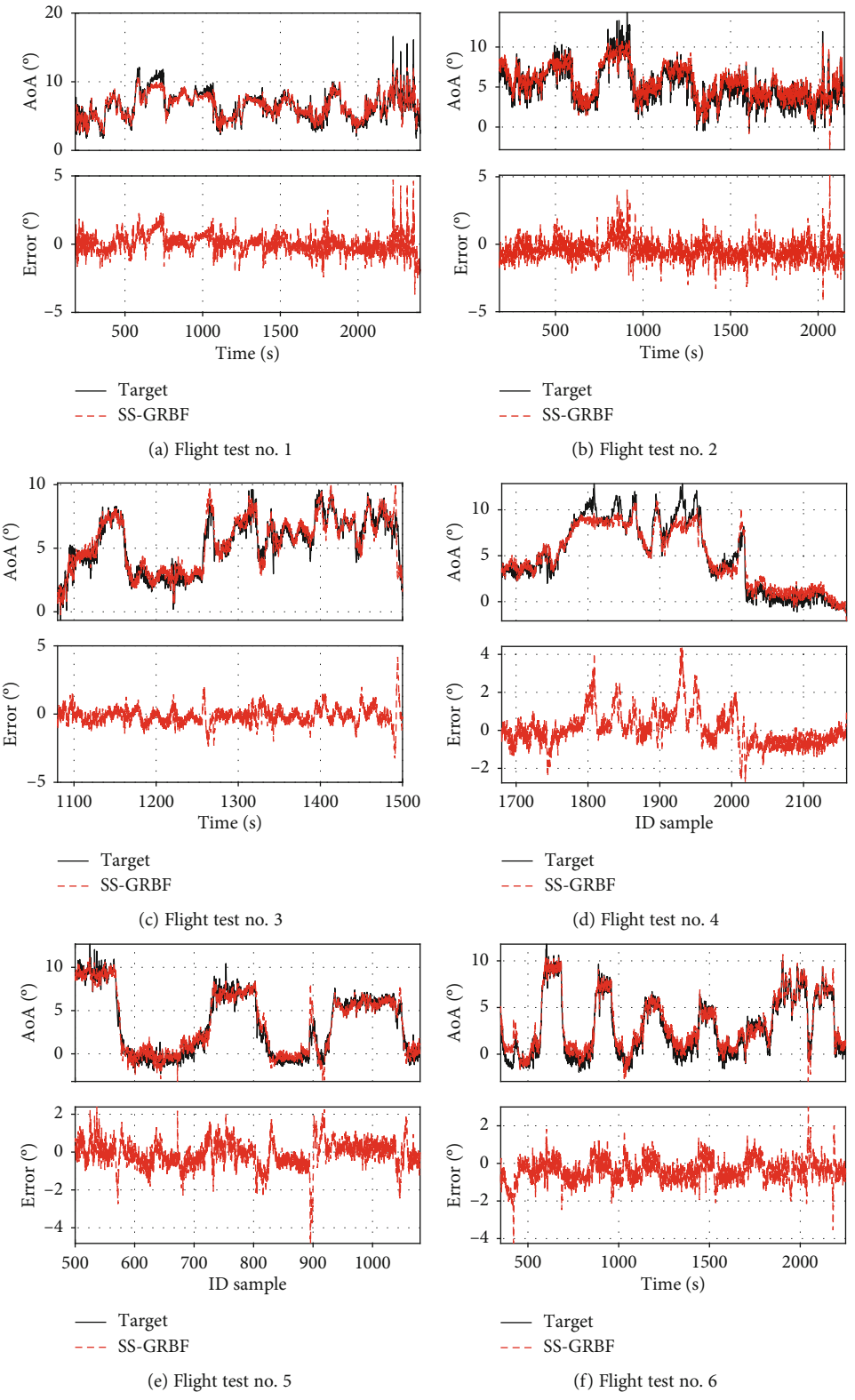
Figure 5(a) represents the best training GRBF-NN performances obtained with all possible combinations of radius and overlap, hereinafter indicated as (R, O). The 2 σ error is the main criteria adopted to choose the maximum number of neurons. In Figure 5(a), the 2σ error is minimised (1.46°) using no more than 13 neurons even though similar results can be achieved with the limit of 50 neurons, whereas the limit of 100 neurons would lead to very large maximum errors and, therefore, it is discarded. Between the maximum number of 13 and 50, considering a possible onboard implementation and the chance to be compared to MLP with the same number of hidden units, the SS-GRBF with a maximum number of 13 neurons is selected.

Best results reported in Figure 5(a) are only the ones obtained with the best couple (R, O) for the training manoeuvres. In order to evaluate their influence on the generalization capabilities, 2σ, 3σ, and max test errors can be analysed in Figure 5(b) for the GRBF-NN with no more than 13 neurons. Considering the 2σ and 3σ errors, it is clear that the radius shall be 2, whereas the overlap can be chosen in the range $0.25 \div 0.75$. Even though the minimum 2σ error (1.12°) is reached for overlap = 0.75, the analysis of the maximum error of Figure 5(a) clearly suggests to choose overlap = 0.5.

Therefore, in this work, the GRBF-NN used for AoA estimation is trained with the EMRAN algorithm using $R = 2$ and O= 0.5 with the maximum number of 13 hidden units. Hereinafter, the latter GRBF-NN is labelled as SS-GRBF.

7. SS-GRBF Performance

The SS-GRBF defined in Section 6.1 is tested on manoeuvres introduced in Section 3, and time histories are reported in Figure 6. Results of AoA estimation, in terms of mean, 2σ, and maximum errors, are collected in Table 3 both for training



(a) Flight test no. 1

(b) Flight test no. 2

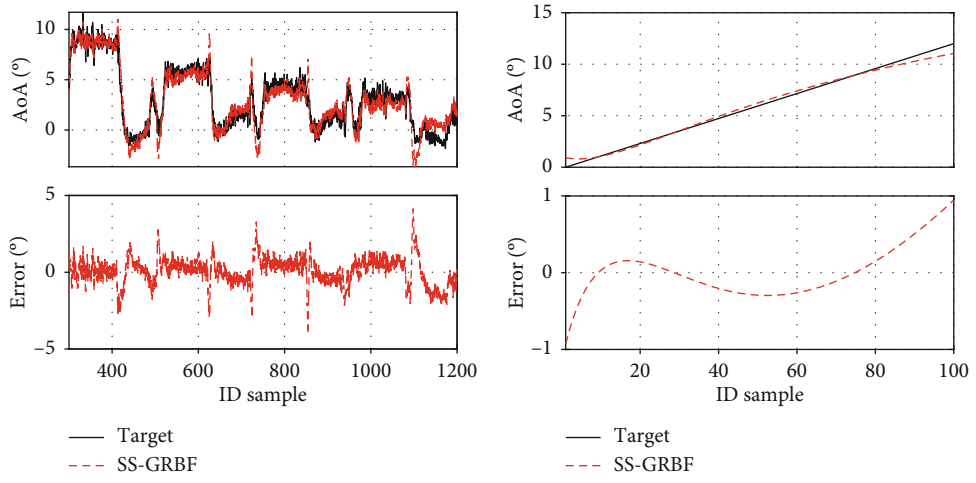
(c) Flight test no. 3

(d) Flight test no. 4

(e) Flight test no. 5

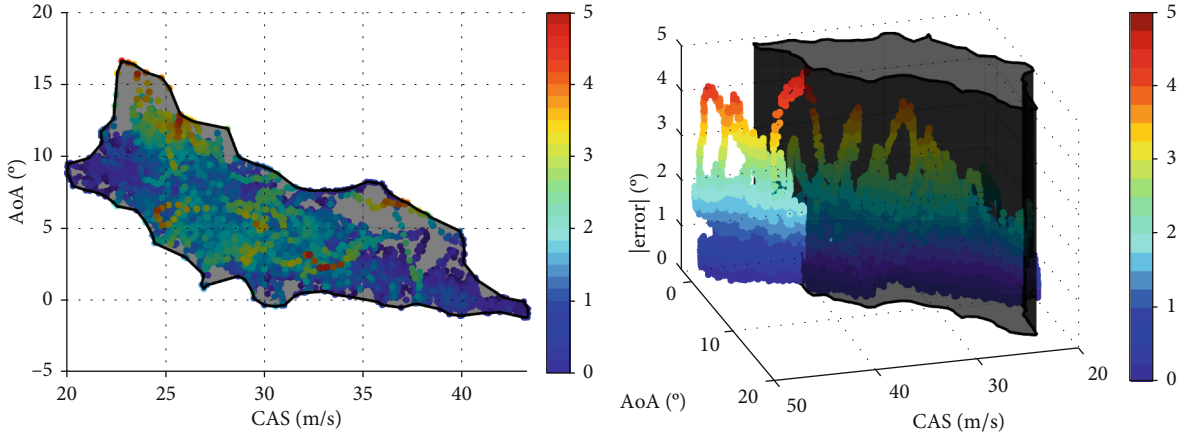
(f) Flight test no. 6

Figure 6: Continued.



(g) Flight test no. 7 (h) Flight test no. 8

Figure 6: True AoA, estimated AoA, and estimation errors for training and validation manoeuvres.



(a) (b)

Figure 7: AoA estimation errors in the plane AoA-CAS for the (a) training and (b) validation dataset.

and validation manoeuvres to be compared to results obtained with SS-MLP and SS-MLP-M introduced in Section 5.

The limited 2σ errors suggest a general good agreement between the true AoA values and SS-GRBF's estimations even if maximum errors around 5° can be observed (e.g., in flight tests 1, 2, and 5).

The SS-GRBF shows worse training performances (relative to flight test nos. 1-4) if compared to SS-MLP (Table 3) in terms of mean, maximum, and 2σ errors. In fact, as known, the batch training algorithm can be very effective on the training with respect to the sequential one. On the other side, as far as validation manoeuvres are concerned, it can be noted that the 2σ error is below the acceptance limits of 1.5° defined in Section 4.1 except for flight test number 4. The maximum error during the simulated steady-state conditions is compliant with respect to the required 1.0°, and it is smaller than the SS-MLP one. Recalling the qualitative nature of the requirements presented in Section 4.1, both noncomplaint 2σ (flight tests 4 and 7) and max errors (flight test 2) do not invalidate the SS-GRBF. In fact, the SS-GRBF shows similar estimation capabilities of the SS-MLP-M-trained ad hoc for AoA estimations with a "modified" training dataset. Moreover, large maximum errors can be also observed in SS-MLP and SS-MLP-M but they not represent an issue as they are very limited in time (often spike errors).

The latter consideration is corroborated by analysis of Figure 7(b) where AoA estimation errors are plotted in the plane AoA-CAS and the training boundary is extruded in order to highlight the location of the maximum errors. It can be noted that the largest errors are located across or outside the training boundary, whereas the maximum error inside the training boundary is about 3.5°. This behaviour is expected because the SS-GRBF is extrapolating instead of working as a regressor.

From Figure 7(a), the AoA estimation error for training manoeuvres reveals, as expected, that the largest errors are

close to the training perimeter even though the maximum errors (up to 5.12° for flight test 2) are inside the training boundary. However, the largest errors are spike errors (as can be seen in Figure 6(b)) that do not usually represent real issues to be handled.

Considerations presented in this section show the potentiality of the GRBF networks to be used as the AoA estimator adopting the proposed approach for AoA estimation dealing with experimental flight test data. Similar performances can be achieved with MLP networks but some actions shall be considered to prepare the training dataset, whereas the GRBF is more tolerant to noisy data and sparse and unbalanced training domain. However, even though the SS-GRBF shows similar 2σ errors of SS-MLP-M, larger maximum errors are always observed.

To conclude, as far as the AoA estimation is concerned, the choice between MLP and GRBF cannot be generic and it is related to the specific application. In fact, if the GRB shows better capability than MLP networks to cope with issues related to operative scenarios, in some circumstances, the MLP can also lead to better results if the training dataset is adequately preprocessed.

8. Conclusion

Training a synthetic sensor based on neural network with experimental flight test data can be challenging when recorded data is noisy and not regularly distributed on the network definition domain. Some issues can arise, and an adequately "modified" training dataset can be adopted. The present work explores the use of a radial basis function network trained sequentially using the EMRAN algorithm with the entire training dataset as alternative to a previous MLP-NN trained with a "modified" training dataset. The GRBF's optimal training parameters and the optimal neuron number are identified. The GRBF-NN showed acceptable generalization capabilities dealing with experimental flight test data comparable to those obtained with MLP-NN trained with a "modified" training dataset. To conclude, as far as the NN-based AoA estimation is concerned, the choice between MLP-NN and GRBF-NN in operative scenarios is not obvious because of their comparable performances and the decision should be weighted according to the specific application. However, the proposed analysis suggests that the GRBF should be used when approaching a new research topic, e.g., choosing the right training manoeuvres, whereas, once the training is defined, a performance increase can be obtained with MLP with adequate experimental data manipulations. In fact, from the present work, it emerges that the GRBF are more effective with real flight datathat is often noisy and sparse with variable density in the input domain. On the other hand, MLP can achieve similar, or better, performances if trained using ad hoc modified training dataset.

Nomenclature

ADS: Air data system
AHRS: Attitude and heading reference system
AoA: Angle of attack
AoS: Angle of sideslip
CAS: Calibrated airspeed
EMRAN: Extended minimal resource-allocating network
FTI: Flight test instrumentation
GNSS: Global navigation satellite system
GRBF: Generalized radial basis function
MLP: Multilayer perceptron
MRAN: Minimal resource-allocating network
NARX: Nonlinear autoregressive exogenous
NN: Neural network
RAN: Resource-allocating network
RBF: Radial basis function
SS: Synthetic sensors
UAM: Urban air mobility
UAV: Unmanned aerial vehicles
ULM: Ultralight machine.

Data Availability

The flight data used to support the findings of this study have not been made available because of company confidentiality.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] J. Flottau, "Boeing 737 max return decision in January," Aviation Week & Space Technology, 2019, https://aviationweek.com/air-transport/easas-director-expects-boeing-737-max-returndecision-january.

[2] European Aviation Safety Agency, EASA, "Easy access rules for unmanned aircraft systems," 2015.

[3] European Aviation Safety Agency, EASA, "Proposed special condition for small-category VTOL aircraf SC-VTOL-01," 2019.

[4] J. Gertler, "Analytical redundancy methods in fault detection and isolation - survey and synthesis," *IFAC Proceedings Volumes*, vol. 24, no. 6, pp. 9–21, 1991.

[5] T. J. Rohloff, S. A. Whitmore, and I. Catton, "Fault-tolerant neural network algorithm for flush air data sensing," *Journal of Aircraft*, vol. 36, no. 3, pp. 541–549, 1999.

[6] M. Perhinschi, G. Campa, M. Napolitano, M. Lando, L. Massotti, and M. L. Fravolini, "Modelling and simulation of a fault-tolerant flight control system," *International Journal of Modelling and Simulation*, vol. 26, no. 1, pp. 1–10, 2006.

[7] A. D. Pouliezos and G. S. Stavrakakis, "Analytical redundancy methods," in *Real Time Fault Monitoring of Industrial Processes*, vol. 12, pp. 93–178, Springer, Netherlands, Dordrecht, 1994.

[8] L. Garbarino, G. Zazzaro, N. Genito, G. Fasano, and D. Accardo, "Neural network based architecture for fault detection and isolation in air data systems," in *2013 IEEE/AIAA 32nd Digital Avionics Systems Conference (DASC)*, pp. 2D4–1–2D4–11, East Syracuse, NY, USA, October 2013.

[9] R. Eubank, E. Atkins, and S. Ogura, "Fault detection and fail-safe operation with a multiple-redundancy air-data system,"

in *AIAA Guidance, Navigation, and Control Conference*, pp. 1–14, Toronto, Ontario, Canada, August 2012.

[10] P. Lu, L. Van Eykeren, E.-J. Van Kampen, and Q. Chu, "Air data sensor fault detection and diagnosis with application to real flight data," in *AIAA Guidance, Navigation, and Control Conference*, pp. 590–604, Kissimmee, Florida, January 2015.

[11] F. A. P. Lie and D. Gebre-Egziabher, "Synthetic air data system," *Journal of Aircraft*, vol. 50, no. 4, pp. 1234–1249, 2013.

[12] A. Calia, E. Denti, R. Galatolo, and F. Schettini, "Air data computation using neural networks," *Journal of Aircraft*, vol. 45, no. 6, pp. 2078–2083, 2008.

[13] K. Sun, C. D. Regan, and D. Gebre-Egziabher, "Observability and performance analysis of a model-free synthetic air data estimator," *Journal of Aircraft*, vol. 56, no. 4, pp. 1471–1486, 2019.

[14] A. Lerro, A. Brandl, and P. Gili, "Model-free scheme for angle-of-attack and angle-of-sideslip estimation," *Journal of Guidance, Control, and Dynamics*, vol. 44, no. 3, pp. 595–600, 2021.

[15] P. Tian, H. Chao, M. Rhudy, J. Gross, and H. Wu, "Wind sensing and estimation using small fixed-wing unmanned aerial vehicles: a survey," *Journal of Aerospace Information Systems*, vol. 18, no. 3, pp. 132–143, 2021.

[16] A. Lerro, A. Brandl, M. Battipede, and P. Gili, "A data-driven approach to identify flight test data suitable to design angle of attack synthetic sensor for flight control systems," *Aerospace*, vol. 7, no. 5, p. 63, 2020.

[17] S. Fabri and V. Kadirkamanathan, "Dynamic structure neural networks for stable adaptive control of nonlinear systems," *IEEE Transactions on Neural Networks*, vol. 7, no. 5, pp. 1151–1167, 1996.

[18] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303–314, 1989.

[19] J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function networks," *Neural Computation*, vol. 3, no. 2, pp. 246–257, 1991.

[20] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice Hall PTR, USA, 2nd edition, 1998.

[21] M. Maruyama, F. Girosi, and T. Poggio, "Connection between GRBF and MLP," *Artificial Intelligence Memo*, vol. 1, no. 1291, 1992.

[22] M. A. Arbib and J. A. Robinson, *Natural and Artificial Parallel Computation*, MIT Press, Cambridge, MA, USA, 1990.

[23] T. Rajkumar and J. Bardina, "Prediction of aerodynamic coefficients using neural networks for sparse data," in *Proceedings of the Fifteenth International Florida Artificial Intelligence Research Society Conference*, pp. 242–246, AAAI Press, 2002.

[24] J. L. Bernier, J. Ortega, E. Ros, I. Rojas, and A. Prieto, "A quantitative study of fault tolerance, noise immunity, and generalization ability of MLPS," *Neural Computation*, vol. 12, no. 12, pp. 2941–2964, 2000.

[25] J. L. Bernier, A. F. Díaz, F. J. Fernández et al., "Assessing the noise immunity and generalization of radial basis function networks," *Neural Processing Letters*, vol. 18, no. 1, pp. 35–48, 2003.

[26] P. A. Samara, G. N. Fouskitakis, J. S. Sakellariou, and S. D. Fassois, "Aircraft angle-of-attack virtual sensor design via a functional pooling NARX methodology," in *2003 European Control Conference (ECC)*, pp. 1816–1821, Cambridge, UK, September 2003.

[27] M. Battipede, P. Gili, and A. Lerro, "Neural networks for air data estimation: test of neural network simulating real flight instruments," in *Engineering Applications of Neural Networks*, C. Jayne, S. Yue, and L. Iliadis, Eds., pp. 282–294, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[28] K. Kufieta, K. Sivamoorthy, and Vörsmann, "Application and comparison of neural networks and optimization algorithms as a virtual angle of attack sensor," in *2013 IEEE International Conference on Computational Intelligence and Cybernetics (CYBERNETICSCOM)*, pp. 6–10, Yogyakarta, Indonesia, December 2013.

[29] A. Lerro, M. Battipede, P. Gili, and A. Brandl, "Aerodynamic angle estimation: comparison between numerical results and operative environment data," *CEAS Aeronautical Journal*, vol. 11, no. 1, pp. 249–262, 2020.

[30] A. Lerro, M. Battipede, and P. Gili, "System and process for measuring and evaluating air and inertial data," 2013, Patent No. EP3022565A2.

[31] M. R. Napolitano, V. Casdorph, C. Neppach, S. Naylor, M. Innocenti, and G. Silvestri, "Online learning neural architectures and cross-correlation analysis for actuator failure detection and identification," *International Journal of Control*, vol. 63, no. 3, pp. 433–455, 1996.

[32] L. Trainelli and A. Rolando, "Reliable and cost-effective flight testing of ultralight aircraft," *Journal of Aircraft*, vol. 48, no. 4, pp. 1342–1350, 2011.

[33] J. Tukey, *Exploratory Data Analysis, Chapter 2*, Addison-Wesley series in behavioral science.Addison-Wesley Publishing Company, 1977.

[34] B. Etkin and L. Reid, *Dynamics of Flight: Stability and Control*, Wiley, Third edition, 1995.

[35] A. Lerro, M. Battipede, P. Gili, and A. Brandl, "Advantages of neural network based air data estimation for unmanned aerial vehicles," *International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, vol. 11, no. 5, pp. 1090–1099, 2017.

[36] SAE International, *Minimum Performance Standards, for Angle of Attack (AoA) and Angle of Sideslip (AoS)*, 2019.

[37] A. Lerro, A. Brandl, M. Battipede, and P. Gili, "Preliminary design of a model-free synthetic sensor for aerodynamic angle estimation for commercial aviation," *Sensors*, vol. 19, no. 23, p. 5133, 2019.

[38] R. M. Sanner and J. J. E. Slotine, "Gaussian networks for direct adaptive control," *IEEE Transactions on Neural Networks*, vol. 3, no. 6, pp. 837–863, 1992.

[39] J. Platt, "A resource-allocating network for function interpolation," *Neural Computation*, vol. 3, no. 2, pp. 213–225, 1991.

[40] G. Campa, M. L. Fravolini, B. Seanor et al., "On-line learning neural networks for sensor validation for the flight control system of a b777 research scale model," *International Journal of Robust and Nonlinear Control*, vol. 12, no. 11, pp. 987–1007, 2002.

[41] M. Basseville and A. Benveniste, "Design and comparative study of some sequential jump detection algorithms for digital signals," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 31, no. 3, pp. 521–535, 1983.

[42] P. Simpson, *Artificial Neural Systems: Foundations, Paradigms, Applications, and Implementations. Neural Networks*, Elsevier Science Limited, 1990.

[43] C. S. Chen and R. S. Nutter, "An extended back-propagation learning algorithm by using heterogeneous processing units," in *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*, vol. 3, pp. 988–993, Baltimore, MD, USA, June 1992.

[44] P. Saratchandran, N. Sundararajan, and Y. Li, "Analysis of minimal radial basis function network algorithm for real-time identification of nonlinear dynamic systems," *EE Proceedings - Control Theory and Applications*, vol. 147, no. 4, pp. 476–484, 2000.