drones

Article

Time-Domain Identification Method Based on Data-Driven Intelligent Correction of Aerodynamic Parameters of Fixed-Wing UAV

Dapeng Yang [1,2], Jianwen Zang [3,*], Jun Liu [3] and Kai Liu [3]

[1] Department of Aeronautics and Astronautics, Fudan University, Shanghai 200433, China; dpyang498997537@163.com
[2] Shenyang Aircraft Design & Research Institute, Shenyang 110035, China
[3] School of Aeronautics and Astronautics, Dalian University of Technology, Dalian 116024, China; liujun65@dlut.edu.cn (J.L.); carsonliu@dlut.edu.cn (K.L.)
* Correspondence: zangjw@mail.dlut.edu.cn

Abstract: In order to overcome the influence of complex environmental disturbance factors such as nonlinear time-varying characteristics on the dynamic control performance of small fixed-wing UAVs, the nonlinear expression relationship of neural networks (NNs) is combined with the recursive least squares (RLSs) identification algorithm. This paper proposes a hybrid aerodynamic parameter identification method based on NN-RLS offline network training and online learning correction. The simulation results show that compared with the real value of the identification value obtained by this algorithm, the residual error of the moment coefficient is reduced by 69%, and the residual error of the force coefficient is reduced by 89%. Under the same identification accuracy, the identification time is shortened from the original 0.1 s to 0.01 s. Compared with traditional identification algorithms, better estimation results can be obtained. By using this algorithm to continuously update the NN model and iterate repeatedly, iterative learning for complex dynamic models can be realized, providing support for the optimization of UAV control schemes.

Keywords: time-domain identification method; data-driven; intelligent correction; aerodynamic parameters; fixed-wing UAV

Citation: Yang, D.; Zang, J.; Liu, J.; Liu, K. Time-Domain Identification Method Based on Data-Driven Intelligent Correction of Aerodynamic Parameters of Fixed-Wing UAV. *Drones* **2023**, *7*, 594. https://doi.org/10.3390/drones7090594

Academic Editor: Mostafa Hassanalian

Received: 14 August 2023
Revised: 13 September 2023
Accepted: 16 September 2023
Published: 21 September 2023

1. Introduction

During high altitude and low-speed flight, small fixed-wing UAVs may face a greater measurement noise environment than large aircraft because their smaller weight and size lead to faster response dynamics and are more sensitive to environmental factors such as wind field interference. Moreover, practical factors such as sensor cost control can also easily introduce additional measurement noise. These responses change as disturbances cause changes in the dynamic parameters of the UAV system, increasing the difficulty of controller design [1–3]. Therefore, in the process of flight, if the aerodynamic characteristics and parameters of the UAV can be identified in real time, not only can the robustness and adaptability of the system be enhanced, but also the control accuracy can be effectively improved [4,5].

Traditional identification algorithms are mostly used for models with strong linearization. When the model is highly nonlinear or time-varying quickly, it is necessary to add high-order nonlinear terms. Since the model order of aerodynamic parameters is higher, the curves drawn by it may be more complex or have severe fluctuations and jumps, which will seriously affect the identification results of aerodynamic parameters. In this paper, the neural network can be used to fit any nonlinear function, and the NN-RLS combination identification algorithm is proposed. NN-RLS can not only train the network model offline but also use historical flight data to correct the model. It is also possible to directly use the

trained network to quickly adjust the parameters when online so as to ensure the speed of parameter estimation while ensuring the accuracy of parameter estimation. In recent years, some scholars have conducted research on the identification algorithm.

In the literature [6], Jiang's method is only suitable for simple linear models and cannot be directly used for complex nonlinear motion models. However, the intelligent identification algorithm based on NN can fit nonlinear functions with arbitrary accuracy. In the literature [7,8], Yang and Zhang combined the arbitrary approximation characteristics of the backpropagation NN to conduct offline training of aerodynamic parameters. They used the results of NN training to identify the aircraft rudder effect parameters in real-time. However, compared with the research method of this article, without the process of correcting the original NN with real flight test data, the goal of lifelong learning cannot be achieved. In the literature [9], Li proposed to use the least squares support vector machine (LS-SVM) to identify the nonlinear dynamical system when the aircraft is flying at a high angle of attack and use the method of network search and cross-validation to select the parameters of the support vector machine. The dynamic reference model of aircraft with a high angle of attack is established. However, compared with the research method of this article, the LS-SVM network takes a long time to learn. Additionally, it is difficult to realize online real-time identification due to the low speed of the support vector machine. In the literature [10], the NN model based on the Gauss-based function is established by Wang to determine the structure and the input and output relations of the network. Then, the K-means algorithm is used to train the RBF to obtain the parameters of the network model, which is used to replace the dynamic model of the aircraft. Compared with the research method of this article, Wang's method focuses on using an RBF neural network to replace the aircraft dynamics model and does not extend to fitting aerodynamic parameters. In the literature [11], a sample expansion combined with a support vector machine and the online fast correction method for NN parameters is proposed by Pu. However, his method needs to use the SVM method for data expansion during the simulation process to reach an order of magnitude suitable for neural network training, which introduces data errors to a certain extent. In the literature [12], the selection of the lateral–directional excitation signal and the design method of its parameters are established by Tai based on the requirements for the identifiability of the aerodynamic derivatives. Moreover, a step-by-step method for the identification of aerodynamic force and moment derivatives is established. This method simplifies the solution method for aerodynamic force and aerodynamic moment step by step. It is only suitable for offline simulation and does not have the characteristics of online identification. The algorithmic identification takes a long time.

The intelligent identification algorithm of nonlinear aerodynamic parameters is based on the data-driven multi-level deep learning network proposed in this paper. This approach uses the characteristics of the deep learning network to fit the nonlinear function with arbitrary precision and realizes the modeling of the nonlinear aerodynamic parameter model, which has stronger nonlinearity. The modeling ability can also learn the potential laws and characteristics in the input data and has better generalization ability [13,14]. In addition, the deep learning network performs model training based on the optimization algorithm of gradient descent, which can improve computational efficiency through batch calculation [15]. Furthermore, this approach accounts for the deviation between the real aerodynamic data and the nominal aerodynamic data. The multi-level deep learning network is used to identify the deviation incremental compensation of the aerodynamic parameters so as to realize the accurate correction of the original aerodynamic parameters [16]. The multi-level deep network can realize the weight matrix update iteration of the offline network driven by historical flight test data, which can adapt to different dataset characteristics and changes, better adapt to complex systems, and realize lifelong learning.

2. Offline Training of Aerodynamic Force and Moment NN Model

Since the NN was proposed, it has achieved great development and application in pattern recognition, perceptual learning, model building, and intelligent control. NN can

approximate the input and output characteristics of nonlinear dynamical systems with arbitrary precision and is an effective tool for the identification and modeling of nonlinear dynamical systems. Using it as a nonlinear dynamical system identification model can better solve the identification modeling and problems of complex nonlinear dynamical systems. At present, the most widely used and relatively mature NN is the backpropagation NN, which is a typical feed-forward network in the NN. This is especially suitable for dealing with the nonlinearity, uncertainty, and approximation of the system or structural characteristics of the system, identifying functions, etc.

As shown in Figure 1, the whole algorithm is divided into three stages. In the offline network training stage, the flight state data set of the fixed-wing UAV is obtained through the CFD simulation experiment. The multivariate orthogonal time-domain identification method is used to optimize the reference configuration. The offline NN is trained on the basis of the benchmark dynamic model. The mapping expression of the flight state data to the force coefficient and moment coefficient is established through the NN. The network takes the real-time flight status data as the input and the force and moment coefficients as the output. In the online compensation stage, online learning captures the time-varying deviation of the network model through the nonlinear recursive orthogonal time-domain identification method. In the offline correction stage, combined with historical flight data, a data-driven iterative correction method for offline learning network weights is carried out. With the accumulation of UAV flight test data, the mapping value of the flight state data to the force coefficient and moment coefficient in the offline NN is closer to the real value.

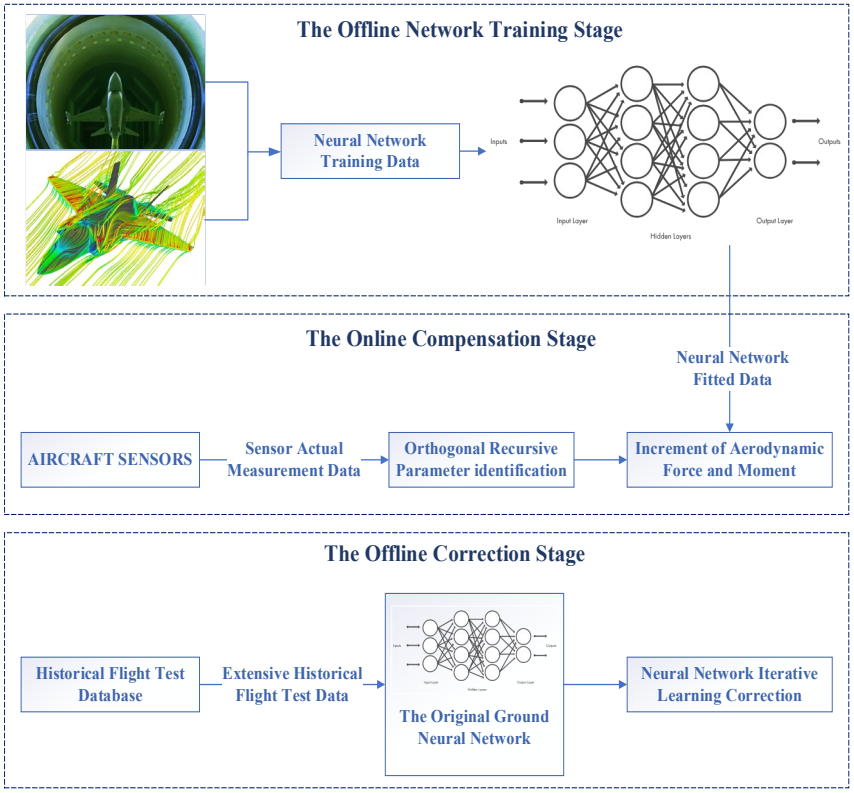


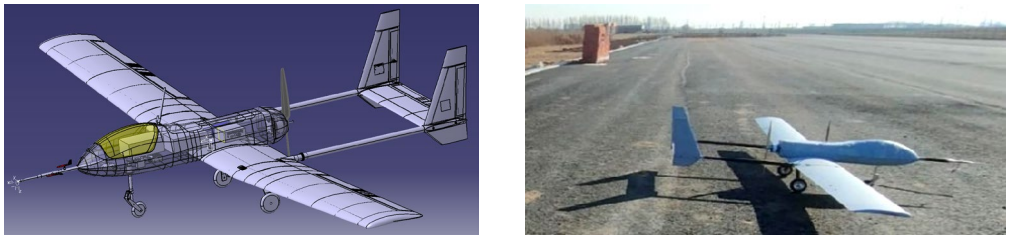
Figure 1. Flow chart of an intelligent nonlinear identification algorithm of aircraft aerodynamic parameters based on data-driven multi-layer deep learning network modification.

ANSYS Fluent in Pointwise V18.3 R1 software is used to obtain the aerodynamic data of the UAV model. The main operation process is as follows: First, the geometric model of the fixed-wing UAV is established using CATIA software and imported into the CFD software, and the body parameters are shown in Table 1. Second, generate a computational mesh that matches the geometric model. Then, determine the boundary conditions of the simulation area and choose an appropriate flow model. In this paper, a single-component

complete gas is selected. The Euler equation is used for the fluid properties. The specific heat ratio is 1.4, the molecular weight of the gas is 2.89×10^{-2}, and the initial partition number of the flow field is 5. CFD software is used to simulate the flow. Additionally, the CFD software solved the velocity, pressure, temperature, and other parameters of the flow field according to the selected flow model, grid, and boundary conditions. We extracted the required aerodynamic data, such as lift coefficient and pitching moment coefficient, from the simulation results. The UAV model established by CATIA software is shown in Figure 2, and the CFD simulation process is shown in Figure 3.

Table 1. Basic parameters of a fixed-wing UAV.

Parameter	Value
span	3.606 m
body length	2.245 m
body height	0.640 m
weight	20 kg
load capacity	2 kg



(**a**) (**b**)

Figure 2. Fixed-wing UAV picture. (**a**) CATIA software 3D modeling picture; (**b**) UAV actual flight scene picture.

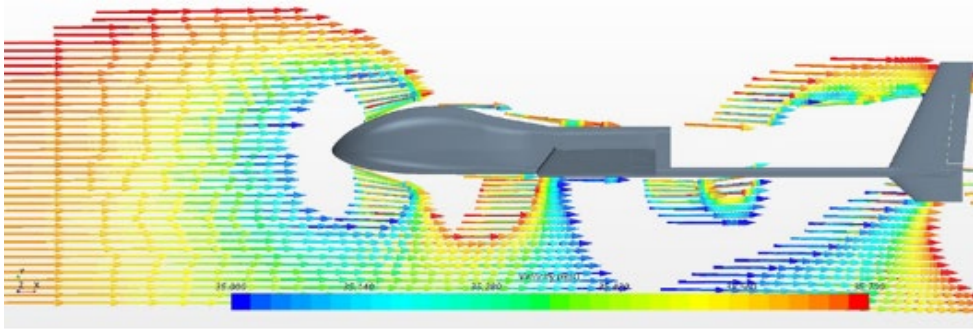


Figure 3. CFD simulation to obtain the NN training set.

The process of training the deep learning network is a process of continuously correcting the network output and the expected output. This adjustment is achieved by backpropagating the error signal from the output end and continuously correcting the weighting coefficients during the propagation process until the output at the output end and the expected value approach to a certain extent. After the adjustment of the network weight coefficients is completed for the sample, it is sent to another sample mode for similar learning until the training and learning of all samples are completed. Finally, the dynamic NN model is obtained [17].

The NN model is shown in the figure, and the given training set $x = [x_1, x_2, \cdots, x_n]^T$ is the input value of the NN; $z = [z_1, z_2, \cdots, z_n]^T$ is the output value of the NN. The NN can be regarded as a nonlinear function, and its input value and predicted value can be regarded as the independent variable and dependent variable of the function, respectively. When the

number of input nodes is n and the number of output nodes is l, the NN is equivalent to the functional mapping relationship from n independent variables to l dependent variables.

For the output layer, its expression is:

$$z = g(W^T y + B_0) \tag{1}$$

In the formula, g is the activation function; y is the neuron in the hidden layer, $y = [y_1, y_2, \ldots, y_m]^T$; W is the weight matrix from the hidden layer to the output layer; W_{jk} represents the connection weight between the j neuron in the hidden layer and the k neuron in the input layer, then W can be written as:

$$W = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1l} \\ w_{21} & w_{22} & \cdots & w_{2l} \\ \cdots & \cdots & \cdots & \cdots \\ w_{m1} & w_{m2} & \cdots & w_{ml} \end{bmatrix} \tag{2}$$

The threshold B_0 expression is:

$$B_0 = \begin{bmatrix} b_{w1} & b_{w2} & \cdots & b_{wl} \end{bmatrix}^T \tag{3}$$

The output layer y can be written as:

$$y = f\left(V^T x + B_1\right) \tag{4}$$

where f is the activation function; x is the neuron in the input layer, $x = [x_1, x_2, \cdots, x_n]^T$; V is the weight matrix from the input layer to the hidden layer; and v_{ij} represents the connection weight between the i neuron in the input layer and the j neuron in the hidden layer, can be written as:

$$V = \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1m} \\ v_{21} & v_{22} & \cdots & v_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ v_{n1} & v_{n2} & \cdots & v_{nm} \end{bmatrix} \tag{5}$$

The threshold B_1 expression is:

$$B_1 = \begin{bmatrix} b_{v1} & b_{v2} & \cdots & b_{vm} \end{bmatrix}^T \tag{6}$$

The above feed-forward NN's expression is:

$$z = g\left(W^T f\left(V^T x + B_1\right) + B_0\right) \tag{7}$$

The activation functions f and g are selected as tanh function and linear function respectively. f can be expressed as:

$$f(x) = \frac{1 - e^{-x}}{1 + e^{-x}} \tag{8}$$

The backpropagation of the error is to first calculate the output error of each layer of neurons from the output layer by layer. After calculating these errors, the weight and threshold of each layer are adjusted according to the error of the objective function so that the final output of the modified network can be close to the expected value. In this section, the objective function of the NN is proposed to be measured by the sum of mean square errors [18].

For a training example, $\left(x^k, z^k\right)$, the mean square error is:

$$E_k = \frac{1}{2}\sum_{j=1}^{l}\left(\hat{z}_j^k - z_j^k\right)^2 \tag{9}$$

Then the total error for all training examples is:

$$MSE = \frac{1}{2p}\sum_{k=1}^{p}\sum_{j=1}^{l}\left(\hat{z}_j^k - z_j^k\right)^2 \tag{10}$$

where p is the number of training samples.

The training method of the NN in this project intends to adopt the L–M training method. The L–M algorithm is an improved Gauss–Newton method, and its form is as follows:

$$\Delta x = -H^{-1}(x)g(x) \tag{11}$$

In the formula, $H(x)$ is the Hessian matrix of the function; $g(x)$ is the gradient. At this point, the Hessian matrix of the gradient and function can be expressed as:

$$\begin{cases} g(x) = J^T(x)e(x) \\ H(x) = J^T(x)J(x) + S(x) \end{cases} \tag{12}$$

$J(x)$ is the Jacobian matrix of the function:

$$J(x) = \begin{bmatrix} \frac{\partial e_1(x)}{\partial x_1} & \frac{\partial e_1(x)}{\partial x_2} & \cdots & \frac{\partial e_1(x)}{\partial x_n} \\ \frac{\partial e_2(x)}{\partial x_1} & \frac{\partial e_2(x)}{\partial x_2} & \cdots & \frac{\partial e_2(x)}{\partial x_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial e_N(x)}{\partial x_1} & \frac{\partial e_N(x)}{\partial x_2} & \cdots & \frac{\partial e_N(x)}{\partial x_n} \end{bmatrix} \tag{13}$$

Through the above calculation, the L–M algorithm calculates Δx to update x at each step by approximating the Hessian matrix [19], where Δx is:

$$\Delta x = \left[J^T(x)J(x) + \mu I\right]^{-1} J^T(x)e(x) \tag{14}$$

We applied this method to repeat the reverse error propagation algorithm for all data, adjust the weights and thresholds of each layer, and complete the training of the NN.

3. Dynamic Model Configuration Optimization Based on Multivariate Orthogonal Function

The multivariate orthogonal function nonlinear aerodynamic coefficient modeling based on real-time flight data needs to solve two problems: First, the structure of the model needs to be determined. This requires removing items that have little influence on the model fitting effect from a large number of nonlinear items in the candidate function pool. The minimum prediction mean square error PSE criterion can meet the above requirements. Second, structural parameters need to be identified based on real-time data. The traditional method of parameter estimation based on least squares regression will lead to an ill-conditioned regression matrix due to the linear correlation between variables, resulting in inaccurate parameter estimation.

The coefficient C_j to be identified is used as the dependent variable, and the independent variables are m n-dimensional independent variables such as Mach number, angle of attack, rudder deflection angle, and flap deflection angle. A set of ordered positive integer

sequences $\{k_1, k_2, \cdots, k_m\}$ is specified to obtain the candidate monomial set of independent variables, also known as regression factors:

$$w_i = x_1^{k_1} \cdot x_2^{k_2} \cdot \cdots \cdot x_m^{k_m}, i = 1, 2, \cdots, n \tag{15}$$

In order to meet the conditions of the subsequent matrix orthogonal decomposition, the dimension of the initialization data here depends on the number of monomials n.

We define $\varphi(k) = k_1 + k_2 + \cdots + k_m$ as the order of the monomial [20].

$$\begin{array}{l} \widetilde{k} \Leftrightarrow \{k_1, k_2, \cdots, k_{\mu-1}, k_\mu, k_{\mu+1}, \cdots, k_m\} \\ k \Leftrightarrow \{k_1, k_2, \cdots, k_{\mu-1}, k_\mu + 1, k_{\mu+1}, \cdots, k_m\} \end{array} \tag{16}$$

where k represents the new sequence obtained by updating the $\widetilde{k}$ base sequence. All the index sequences are obtained through the above process, which can be inserted into the above formula to obtain the candidate monomial $W_0 = [w_1, w_2, \cdots, w_n]$.

Decompose W_0 to obtain the initialized orthogonal function matrix Q and coefficient matrix R:

$$W_0 = QR \tag{17}$$

The model to be identified can be represented by a linear combination of w_i:

$$\begin{array}{l} \hat{C}_j = \theta_1 w_1 + \theta_2 w_2 + \ldots + \theta_n w_n + e \\ \quad = W_0 \theta + e \\ \quad = QR\theta + e \\ \quad = Qa + e \end{array} \tag{18}$$

In the formula, $\theta = [\theta_1, \theta_2, \cdots, \theta_n]^T$ and $a = R\theta$.

The ultimate goal of orthogonal modeling is to obtain a set of optimal θ values so that the least squares cost function J of the model is minimized [21], where

$$\begin{array}{l} J = \frac{1}{2}(C_j - W_0\theta)^T (C_j - W_0\theta) \\ = \frac{1}{2}(C_j - Qa)^T (C_j - Qa) \\ = \frac{1}{2}\left[{C_j}^T C_j - \sum_{j=1}^{n} \left(q_j^T C_j \right)^2 \right] \end{array} \tag{19}$$

Calculate the derivative of the cost function in the above formula with respect to θ and solve the equation to obtain the optimal estimate of the unknown parameter θ:

$$\begin{array}{l} W_0^T W_0 \hat{\theta} = W_0^T C_j \\ R\hat{\theta} = Q^T C_j \end{array} \tag{20}$$

By transforming it into a matrix, we have:

$$\begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \cdots & r_{2n} \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & r_{nn} \end{bmatrix} \begin{bmatrix} \hat{\theta}1 \\ \hat{\theta}2 \\ \vdots \\ \hat{\theta}n \end{bmatrix} = \begin{bmatrix} q_1^T C_j \\ q_2^T C_j \\ \vdots \\ q_n^T C_j \end{bmatrix} \tag{21}$$

The second part of the rapid modeling of aerodynamic coefficients is to iteratively update the orthogonal function pool based on real-time data. At the same time, update the

influencing factors on the right side of the above formula. After acquiring the new data $[\xi_1, \xi_2, \cdots, \xi_n]$, the new data are added to the last line of the above formula [22]:

$$\begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \cdots & r_{2n} \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & r_{nn} \\ \xi_1 & \xi_2 & \cdots & \xi_n \end{bmatrix} \begin{bmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \\ \vdots \\ \hat{\theta}_n \end{bmatrix} = \begin{bmatrix} q_1^T C_j \\ q_2^T C_j \\ \vdots \\ q_n^T C_j \\ \zeta \end{bmatrix} \tag{22}$$

where ζ is the new dependent variable data. In order to ensure that the subsequent decomposition conditions are satisfied, the parameter matrix R' must be transformed so that the last row is all zeros. The Givens transformation method is used to triangulate R', and the Givens matrix is defined as follows:

$$G_i = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & & \vdots & \vdots \\ 0 & \cdots & c_i & \cdots & 0 & s_i \\ \vdots & & & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & \cdots & 1 & 0 \\ 0 & \cdots & -s_i & \cdots & 0 & c_i \end{bmatrix} \tag{23}$$

In the formula, $c_i = \frac{r_{ii}}{\sqrt{r_{ii}^2 + \xi_i^2}}, s_i = \frac{\xi_i}{\sqrt{r_{ii}^2 + \xi_i^2}}$, by multiplying all the Givens matrices into the formula [23], we have

$$G_n \cdots G_2 G_1 R' \hat{\theta} = G_n \cdots G_2 G_1 \begin{bmatrix} q_1^T C_j \\ q_2^T C_j \\ \vdots \\ q_n^T C_j \\ \zeta \end{bmatrix} \tag{24}$$

$$\begin{bmatrix} r'_{11} & r'_{12} & \cdots & r'_{1n} \\ 0 & r'_{22} & \cdots & r'_{2n} \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & r'_{nn} \\ 0 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \\ \vdots \\ \hat{\theta}_n \end{bmatrix} = \begin{bmatrix} q'^T_1 C_j \\ q'^T_2 C_j \\ \vdots \\ q'^T_n C_j \\ \varepsilon \end{bmatrix} \tag{25}$$

The above formula is the result of completing a dataset update, in which the symbol with a superscript indicates that it contains information on new data. ε indicates the residual of the new data, which is the residual information that the new data cannot be projected onto an orthogonal basis. In further iterations, it is necessary to replace the last row of the matrix with new data and repeat the above iterative process to update the orthogonal function pool. The whole iterative process only needs to perform simple matrix multiplication and does not need to repeat the orthogonal function generation process, which greatly improves the iterative efficiency. The multivariate orthogonal model screening flow chart is shown in Figure 4.

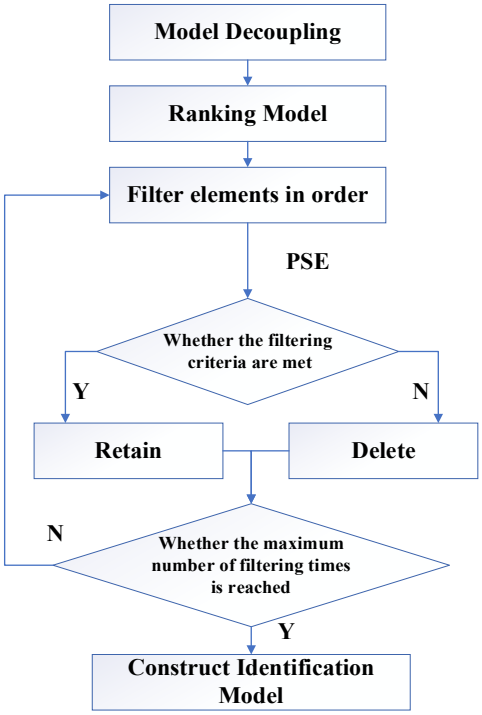


Figure 4. Schematic diagram of the multivariate orthogonal model screening process.

When applied to data not used in the model identification process, the PSE criterion can be used to quantify the squared prediction error of the identified model. The PSE standard is as follows [24]:

$$PSE = \frac{1}{N}(C_j - Q\hat{a})^T(C_j - Q\hat{a}) + \sigma_{\max}^2 \frac{n}{N} = \frac{\hat{J}}{N} + \sigma_{\max}^2 \frac{n}{N} \tag{26}$$

The previous term of PSE is the Mean Squared Fit Error (MSFE)

$$MSFE = \frac{\hat{J}}{N} \tag{27}$$

MSEF decreases monotonically with increasing orthogonal functions.
The second item is the Overfitting Principle (OFP), as follows:

$$OFP = \sigma_{\max}^2 \frac{n}{N} \tag{28}$$

OFP increases monotonically with the increase in the orthogonal function, which can prevent over-parameterization of the model and help improve the accuracy of the model prediction.

Since MSFE decreases monotonically and OFP increases monotonically, each time an orthogonal function is introduced, MSFE decreases and OFP increases. If each orthogonal function is sorted according to its degree of importance and the orthogonal functions are introduced in order, the PSE must have a global minimum. Selecting this point can determine the selected n orthogonal model functions, taking into account the minimum average square error and good predictive power [25].

4. Online Incremental Compensation Dynamics Identification Method Based on Recursive Least Squares

As shown in Figure 5, during the online identification process, the observation model is used to calculate the pitching moment coefficient and lift coefficient, and the difference

between the online observation value and the offline network mapping value is calculated as the input value of the recursive least squares to identify the error increment.

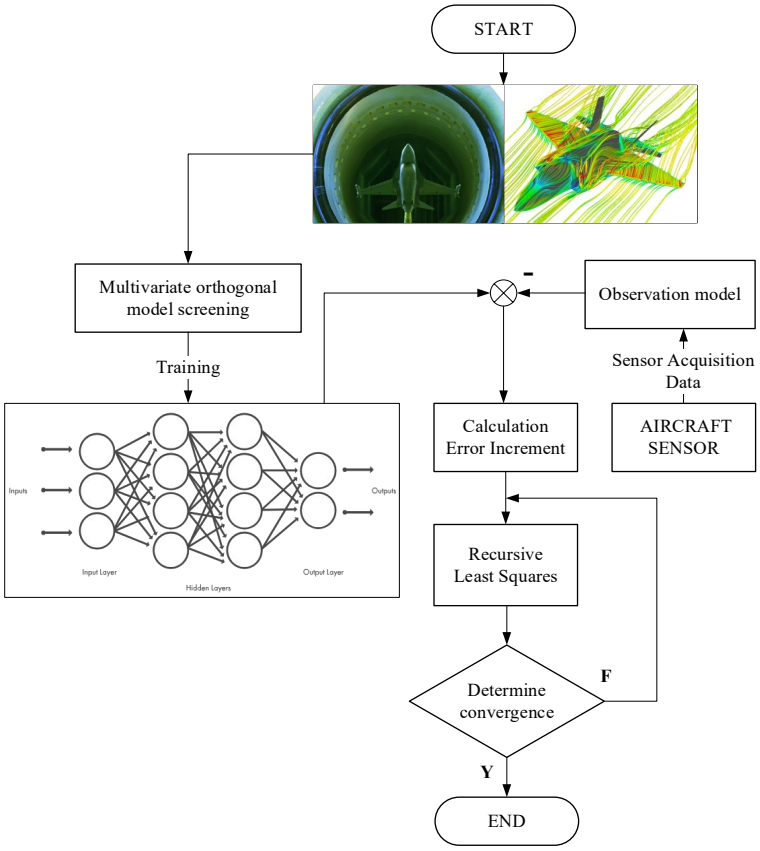


Figure 5. Error increment identification strategy.

For parameter estimation, the relationship between the measured output and the model parameters is more important. The model is called a linear parametric model if the output is given by the following equation:

$$z = H\theta + v \tag{29}$$

For a given z and θ, the best estimate is obtained by minimizing the weighted sum of the squared errors between the measured output and the model output.

$$J(\theta) = \frac{1}{2}(z - H\theta)^T(z - H\theta) \tag{30}$$

Assuming $B(k) = H^T(k)H(k)$, the estimated parameters at the k moments are:

$$\theta(k) = B^{-1}(k)H^T(k)Y(k) \tag{31}$$

The recursive process starts at time $k+1$. In the process of actual use, with the increase in information, the positive definiteness of the information matrix is continuously reduced, and the improvement effect on new information is gradually equal to zero. This phenomenon is called the dataset saturation phenomenon. To solve this problem, a recursive algorithm that introduces the forgetting factor [26] is proposed:

$$\begin{cases} P(k+1) = \frac{1}{\rho^2}[P(k) - K(k)H(k+1)P(k)] \\ \hat{\theta}(k+1) = \hat{\theta}(k) + K(k)\left[z(k+1) - H(k+1)\hat{\theta}(k)\right] \\ K(k) = P(k)H^T(k+1)\left[\rho^2 I + H(k+1)P(k)H^T(k+1)\right]^{-1} \end{cases} \tag{32}$$

In the real flight process, the force and moment coefficients cannot be directly measured by sensors. However, they are calculated based on other observable parameters, so it is necessary to establish an observation model of observable parameters and force and moment coefficients.

$$\begin{cases} C_{m\text{observation}} = \frac{1}{\overline{q}S\overline{c}}[J_y\dot{\omega}_z + (J_x - J_z)\omega_x\omega_y + J_{xz}({\omega_x}^2 - {\omega_y}^2)] \\ C_{\text{Lobservation}} = \frac{ma_y}{\overline{q}S} \end{cases} \tag{33}$$

where C_m is the pitching moment coefficient, $\bar{c}$ is the effective length, and b is the chord length. $\dot{\omega}_z$ is the pitch angular acceleration, J_x, J_{xz}, J_y is the moment of inertia, ω_x is the roll angular rate, and ω_y is the yaw angular rate. C_L is the lift coefficient, $\overline{q}$ is the dynamic pressure, S is the reference area, and a_y is the acceleration in the y axial direction.

In the online real-time identification process, the sensor collects the flight state data in real-time to calculate the observed value of the pitch moment coefficient. It makes a difference with the offline NN mapping value to obtain the error increment of the pitch moment coefficient. According to the recursive least squares identification algorithm, the incremental identification value of each derivative item of the pitching moment coefficient is calculated:

$$\Delta\theta = \left[\Delta C_m^a, \Delta C_m^{\delta_y}, \Delta C_m^{\delta_f}\right] \tag{34}$$

We add the error incremental identification value of the derivative term to the reference value to obtain the identification incremental compensation pitching moment coefficient:

$$z = z_{NN} + \Delta z = z_{NN} + H\Delta\theta + v \tag{35}$$

where z_{NN} is the fitted value of the offline NN based on the flight status data.

5. Data-Driven NN Model Iterative Modification Method

This section is based on the intelligent identification of NN to realize the iterative correction of data-driven offline network weights. With the accumulation of flight test data, the mapping model of flight state data to force coefficient and moment coefficient in the offline learning network is continuously revised, and lifelong learning is finally realized.

Input layer [27]: $\boldsymbol{x}_{\text{Io}}(k) = \boldsymbol{x}_{\text{Ii}}(k)$.

Hidden layer: $\begin{cases} \boldsymbol{x}_{\text{Hi}}(k) = \boldsymbol{W}_{\text{HI}}(k)\boldsymbol{x}_{\text{Io}}(k) \\ \boldsymbol{x}_{\text{Ho}}(k) = \ell[\boldsymbol{x}_{\text{Hi}}(k)] \end{cases}$.

In the formula, $\boldsymbol{x}_{\text{Ho}}(k)$ is the output of the hidden layer of the k step, $\boldsymbol{x}_{\text{Hi}}(k)$ is the input of the hidden layer of the k step, $\boldsymbol{W}_{\text{HI}}(k)$ is the weight matrix for converting the output of the input layer into the input of the hidden layer, and $\ell(\cdot)$ is the activation function of the hidden layer. is the hyperbolic tangent sigmoid function Tanh, and its expression is this:

$$\begin{cases} \ell(x) = (\mathrm{e}^x - \mathrm{e}^{-x})/(\mathrm{e}^x + \mathrm{e}^{-x}) \\ \ell'(x) = 1 - \ell^2(x) \end{cases} \tag{36}$$

Output layer: $\begin{cases} \boldsymbol{x}_{\text{Oi}}(k) = \boldsymbol{W}_{\text{OH}}(k)\boldsymbol{x}_{\text{Ho}}(k) \\ x_{\text{Oo}}(k) = \varphi[\boldsymbol{x}_{\text{Oi}}(k)] \end{cases}$.

In the formula, $\varphi(\cdot)$ is the activation function of the output layer, which is specified as a sigmoid function, and the expression is:

$$\begin{cases} \varphi(x) = 1/(1 + \mathrm{e}^{-x}) \\ \varphi'(x) = \varphi(x)(1 - \varphi(x)) \end{cases} \tag{37}$$

5.1. Establish Loss Function

The update of the weight matrix of the NN should be guided by the minimum loss function J so that the error between the pitching moment coefficient/lift coefficient and

the nominal pitching moment coefficient/lift coefficient of the offline mapping of the NN is minimized.

$$J = \frac{1}{2}\left[C_{m-Offline} - C_{m-Nominal}\right]^2, J = \frac{1}{2}\left[C_{L-Offline} - C_{L-Nominal}\right]^2 \tag{38}$$

5.2. The Weight Matrix W_{OH} Update Algorithm from the Hidden Layer to the Output Layer of NN

The BP update algorithm of the weight matrix is reversed, so the weight matrix $\boldsymbol{W}_{\mathrm{OH}}$ from the hidden layer to the output layer needs to be updated first [28]:

$$\boldsymbol{W}_{\mathrm{OH}}(k+1) = \boldsymbol{W}_{\mathrm{OH}}(k) - \eta\frac{\partial J}{\partial \boldsymbol{W}_{\mathrm{OH}}(k)} + \alpha\Delta\boldsymbol{W}_{\mathrm{OH}}(k) \tag{39}$$

In the formula, we take the negative gradient direction, that is, $-\partial J/\partial \boldsymbol{W}_{\mathrm{OH}}(k)$, to seek the minimum value of the loss function J; η is the learning rate; the last momentum item is added here: α is the inertia coefficient; $\Delta\boldsymbol{W}_{\mathrm{OH}}(k)$ is the weight matrix correction amount of the k step.

The weight matrix correction amount of the $k+1$ step is:

$$\Delta\boldsymbol{W}_{\mathrm{OH}}(k+1) = \boldsymbol{W}_{\mathrm{OH}}(k+1) - \boldsymbol{W}_{\mathrm{OH}}(k) = -\eta\frac{\partial J}{\partial \boldsymbol{W}_{\mathrm{OH}}(k)} + \alpha\Delta\boldsymbol{W}_{\mathrm{OH}}(k) \tag{40}$$

The above formula $\partial J/\partial \boldsymbol{W}_{\mathrm{OH}}(k)$ is unknown; the following uses the chain rule to derive the expression of $\partial J/\partial \boldsymbol{W}_{\mathrm{OH}}(k)$:

$$\frac{\partial J}{\partial \boldsymbol{W}_{\mathrm{OH}}(k)} = \frac{\partial J}{\partial y(k+1)}\frac{\partial y(k+1)}{\partial u(k)}\frac{\partial u(k)}{\partial \boldsymbol{x}_{\mathrm{Oo}}(k)}\frac{\partial \boldsymbol{x}_{\mathrm{Oo}}(k)}{\partial \boldsymbol{x}_{\mathrm{Oi}}(k)}\frac{\partial \boldsymbol{x}_{\mathrm{Oi}}(k)}{\partial \boldsymbol{W}_{\mathrm{OH}}(k)} \tag{41}$$

where

$$\frac{\partial J}{\partial y(k+1)} = -e(k+1),\ e(k+1) = y(k+1) - r(k+1) = m_{znominal}(k+1) - m_{zOffline}(k+1) \tag{42}$$

For $\partial y(k+1)/\partial u(k)$, the symbolic function is used to simplify the expression:

$$\frac{\partial y(k+1)}{\partial u(k)} \triangleq \mathrm{sign}\left(\frac{\partial y(k+1)}{\partial u(k)}\right) \tag{43}$$

The bias produced by the sign function substitution is compensated by the learning rate and inertia coefficient [29]. If the controlled quantity is torque, $\frac{\partial u(k)}{\partial \boldsymbol{x}_{\mathrm{Oo}}(k)} = 1$.

According to the output layer activation function

$$\frac{\partial \boldsymbol{x}_{\mathrm{Oo}}(k)}{\partial \boldsymbol{x}_{\mathrm{Oi}}(k)} = \frac{\partial \varphi\left[\boldsymbol{x}_{\mathrm{Oi}}(k)\right]}{\partial \boldsymbol{x}_{\mathrm{Oi}}(k)} = \varphi'\left[\boldsymbol{x}_{\mathrm{Oi}}(k)\right] \tag{44}$$

and $\begin{cases} \boldsymbol{x}_{\mathrm{Oi}}(k) = \boldsymbol{W}_{\mathrm{OH}}(k)\boldsymbol{x}_{\mathrm{Ho}}(k) \\ x_{\mathrm{Oo}}(k) = \varphi\left[\boldsymbol{x}_{\mathrm{Oi}}(k)\right] \end{cases}$ and after approximation and substitution, it can be simplified [30]:

$$\frac{\partial J}{\partial \boldsymbol{W}_{\mathrm{OH}}(k)} = -\left\{e(k+1)\mathrm{sign}\left(\frac{\partial y(k+1)}{\partial u(k)}\right) \quad \frac{\partial u(k)}{\partial \boldsymbol{x}_{\mathrm{Oo}}(k)} \cdot \varphi'\left[\boldsymbol{x}_{\mathrm{Oi}}(k)\right]\right\}\left[\boldsymbol{x}_{\mathrm{Ho}}(k)\right]^{\mathrm{T}} = -\delta_O\left[\boldsymbol{x}_{\mathrm{Ho}}(k)\right]^{\mathrm{T}} \tag{45}$$

The above formula $\boldsymbol{x}_{\mathrm{Ho}}(k)$ is the output of the first hidden layer.

The final result $\partial J/\partial \boldsymbol{W}_{\mathrm{OH}}(k)$ is a $3 \times n_{\mathrm{h}}$ dimensional matrix, where n_{h} is the number of neurons in the hidden layer.

Then, the weight matrix correction amount $\Delta \boldsymbol{W}_{\mathrm{OH}}$ can be rewritten as:

$$\Delta\boldsymbol{W}_{\mathrm{OH}}(k+1) = \eta\delta_{\mathrm{O}}\left[\boldsymbol{x}_{\mathrm{Ho}}(k)\right]^{\mathrm{T}} + \alpha\Delta\boldsymbol{W}_{\mathrm{OH}}(k) \tag{46}$$

5.3. Update the Algorithm of the Weight Matrix $W_{HI}(k)$ from the Input Layer to the Hidden Layer of NN

$$\frac{\partial J}{\partial \boldsymbol{W}_{\mathrm{OH}}(k)} = \frac{\partial J}{\partial \boldsymbol{x}_{\mathrm{Oi}}(k)} \frac{\partial \boldsymbol{x}_{\mathrm{Oi}}(k)}{\partial \boldsymbol{W}_{\mathrm{OH}}(k)} = \frac{\partial J}{\partial \boldsymbol{x}_{\mathrm{Oi}}(k)} \frac{\partial \boldsymbol{W}_{\mathrm{OH}}(k)\boldsymbol{x}_{\mathrm{Ho}}(k)}{\partial \boldsymbol{W}_{\mathrm{OH}}(k)} = \frac{\partial J}{\partial \boldsymbol{x}_{\mathrm{Oi}}(k)} [\boldsymbol{x}_{\mathrm{Ho}}(k)]^{\mathrm{T}} \tag{47}$$

Comparing formula $\Delta \boldsymbol{W}_{\mathrm{OH}}(k+1) = \eta \delta_{\mathrm{O}} \left[\boldsymbol{x}_{\mathrm{Ho}}(k)\right]^{T} + \alpha \Delta \boldsymbol{W}_{\mathrm{OH}}(k)$, it is found that the following relational formula exists [31]:

$$\xi_{\mathrm{O}} = \frac{\partial J}{\partial \boldsymbol{x}_{\mathrm{Oi}}(k)} = -\boldsymbol{\delta}_{\mathrm{O}}, \frac{\partial J}{\partial \boldsymbol{W}_{\mathrm{HI}}(k)} = \xi_{\mathrm{H}} [\boldsymbol{x}_{\mathrm{Io}}(k)]^{\mathrm{T}} \tag{48}$$

Therefore, the weight matrix correction value $\Delta \boldsymbol{W}_{\mathrm{HI}}$ from the input layer to the hidden layer of the $k+1$ step can be written as:

$$\Delta \boldsymbol{W}_{\mathrm{HI}}(k+1) = -\eta \xi_{\mathrm{H}} [\boldsymbol{x}_{\mathrm{Io}}(k)]^{\mathrm{T}} + \alpha \Delta \boldsymbol{W}_{\mathrm{HI}}(k) = \eta \boldsymbol{\delta}_{\mathrm{H}} [\boldsymbol{x}_{\mathrm{Io}}(k)]^{\mathrm{T}} + \alpha \Delta \boldsymbol{W}_{\mathrm{HI}}(k) \tag{49}$$

where

$$\xi_{\mathrm{H}} = \frac{\partial J}{\partial \boldsymbol{x}_{\mathrm{Hi}}(k)} = \ell' [\boldsymbol{x}_{\mathrm{Hi}}(k)] \cdot \left[(\xi_{\mathrm{O}})^{\mathrm{T}} \boldsymbol{W}_{\mathrm{OH}}(k) \right]^{\mathrm{T}} \tag{50}$$

6. Simulation Example

In order to verify the effectiveness of the method proposed in this paper, this section obtains aerodynamic data based on the aerodynamic function of a fixed-wing UAV and trains an offline NN, which reflects the mapping relationship between the flight state and the force and moment coefficients.

The form of the high-dimensional identification model established offline is as follows [32]:

$$\begin{aligned} C_m &= C_{m0} + C_m^{\alpha}\alpha + C_m^{q}\frac{qc}{2V0} + C_m^{\delta_y}\delta_y + C_m^{\delta_f}\delta_f + C_m^{\alpha^2}\alpha^2 + C_m^{{\delta_y}^2}{\delta_y}^2 \\ &\quad + C_m^{\alpha\delta_y}\alpha\delta_y + C_m^{\alpha^3}\alpha^3 + C_m^{{\delta_y}^3}{\delta_y}^3 \\ C_L &= C_{L0} + C_L^{\alpha}\alpha + C_L^{q}\frac{qc}{2V0} + C_L^{\delta_y}\delta_y + C_L^{\delta_f}\delta_f + C_L^{\alpha^2}\alpha^2 + C_L^{{\delta_y}^2}{\delta_y}^2 \\ &\quad + C_L^{\alpha\delta_y}\alpha\delta_y + C_L^{\alpha^3}\alpha^3 + C_L^{{\delta_y}^3}{\delta_y}^3 \end{aligned} \tag{51}$$

The form of the low-dimensional identification model after screening by the orthogonal model is as follows:

$$\begin{aligned} C_m &= C_m^{\alpha}\alpha + C_m^{\delta_y}\delta_y + C_m^{\delta_f}\delta_f \\ C_L &= C_L^{\alpha}\alpha + C_L^{\delta_y}\delta_y + C_L^{\delta_f}\delta_f \end{aligned} \tag{52}$$

Based on the flight state data, an offline NN is built. Moreover, the NN is trained to map the pitching moment coefficient to the flight state data, such as angle of attack and rudder deflection. The fitting effect of the offline NN is measured by the fitting degree R^2 and the mean square error MSE, and the closer the fitting degree R^2 is to 1, the better the fitting effect is.

6.1. Simulation and Analysis of Offline Training of Aerodynamic Force and Aerodynamic Moment NN Model

Using Pointwise V18.3 R1 software from ANSYS Fluent to calculate the aerodynamic data set of the UAV, a total of 20,000 sets of data are used to build an offline deep learning network. The NN is trained to map the pitching moment coefficient to the flight state data, such as angle of attack and rudder deflection. An offline neural network is a feed-forward network composed of nonlinearly changing units. It is generally composed of five layers of neurons: one input layer, three hidden layers, and one output layer. The neurons in each layer form weighted interconnections, while the neurons located in the same layer have

no connections and do not influence each other. Therefore, the neurons in each layer are only sensitive to the input of the neurons in the previous layer. The output of the neurons in each layer only affects the output of the next layer. The BP neural network structure is determined based on the characteristics of the fitted nonlinear function. This paper studies the nonlinear functions of the lift coefficient and pitching moment coefficient. It has five input parameters: angle of attack, full-motion tail, flap deflection angle, speed, and altitude, and two output parameters: lift coefficient and pitching moment coefficient. In this section, the nonlinear link of the hidden layer of the BP neural network uses the "sigmoid" nonlinear function, and the neuron type of the output layer is a linear output neuron. This algorithm selects 70% of the training data as the training set, and the verification set and the test set each account for 15% of the data. The fitting effect of the offline NN is measured by the fitting degree R^2 and the mean square error MSE, and the closer the fitting degree R^2 is to 1, the better the fitting effect is, as shown in Figure 6.

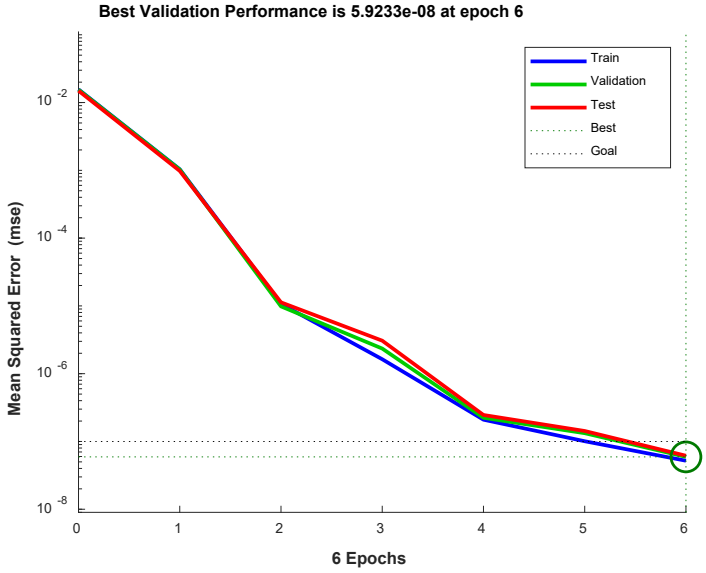


Figure 6. Offline NN test set mean square error.

The offline deep learning network was built according to the CFD flight status data. As shown as Table 2, the fitting degree R^2 was 0.99948, and the mean square error of the training data was 10^{-7}, which indicated that the NN built offline had a good fitting mapping effect on the parameters to be estimated.

Table 2. Goodness of Fit for each set.

	Goodness of Fit
Training	$R^2 = 0.99946$
Validation	$R^2 = 0.99969$
Test	$R^2 = 0.99905$
All	$R^2 = 0.99948$

6.2. Simulation and Analysis of Dynamic Identification Based on Offline Network Learning and Online Compensation

Based on the benchmark NN model of aerodynamic force and aerodynamic moment established in the offline stage and the response data of the online process, the deviation data of the benchmark NN model is calculated and obtained. The increment in aerodynamic force and aerodynamic moment is obtained using the time domain recursive least squares identification method. In this section, we conduct an identification simulation analysis of the pitching moment coefficient and lift coefficient. This analysis is based on the dynamic

identification method of offline network learning and online compensation. We evaluate the aspects of identification accuracy and identification efficiency, respectively.

It can be seen from Figures 7 and 8 that, under the same conditions of identification accuracy, the force coefficient and moment coefficient identified by the hybrid intelligent method converge faster than the force coefficient and moment coefficient obtained by the traditional identification method.The residual and relative error curves of the pitching moment coefficient and lift coefficient are shown in Figures 9 and 10.

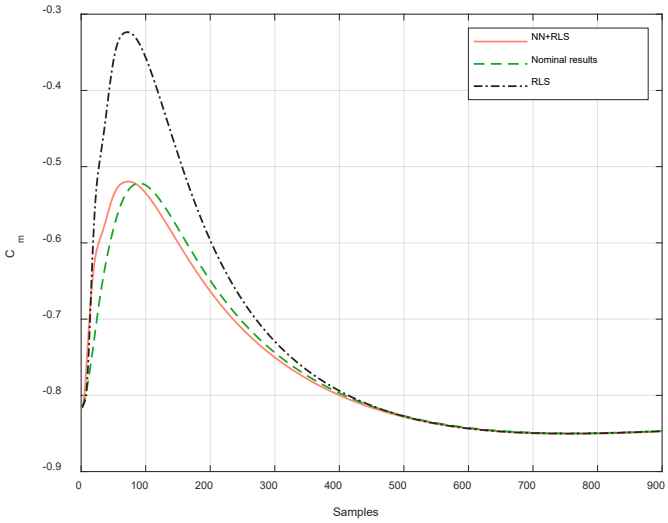


Figure 7. Comparison results of pitching moment coefficients between incremental compensation and network mapping.

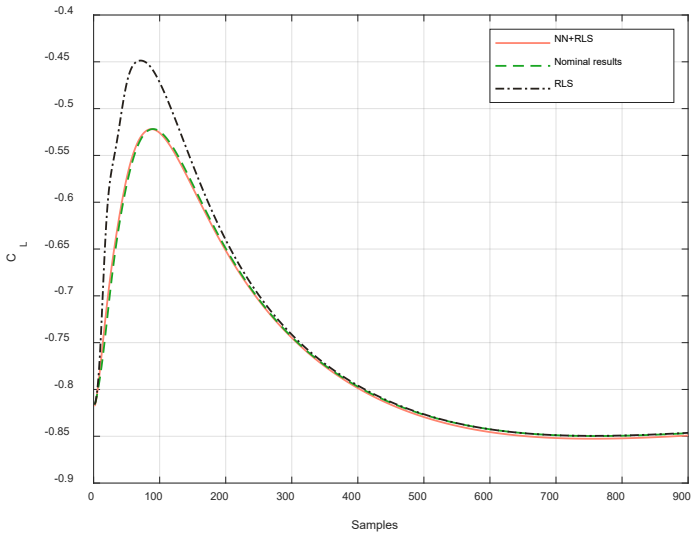


Figure 8. Comparison results of lift coefficients between incremental compensation and network mapping.

In the offline process, the research method in this paper uses the QR decomposition matrix according to the multivariate orthogonal function. It uses the Givens matrix to iterate the data to determine the identification reference configuration. Compared with the traditional online iterative identification configuration, the time consumed by the online iterative process is greatly reduced. Moreover, online learning captures the time-varying deviation of the network model through the nonlinear recursive orthogonal time-domain identification method. These deviations are then integrated with the force coefficient and moment coefficient fitted by the NN, which improves the identification accuracy of the

force coefficient and moment coefficient. As shown in Table 3, it can be seen from the table that under the same identification conditions, the time-consuming task of identifying the moment coefficient is reduced from the original 0.438 s to 0.0674 s. Comparing the incremental compensation identification and the recursive least squares identification methods, the identification time-consuming of the force coefficient is reduced from the original 0.576 s to 0.0753 s.

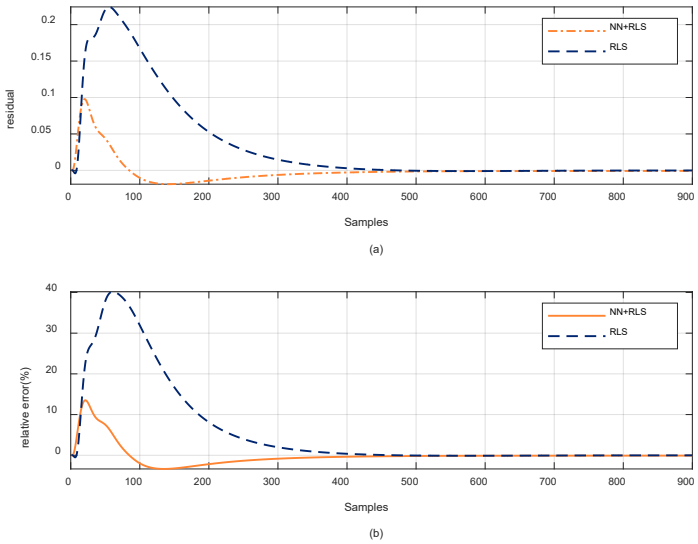


Figure 9. Comparison results of incremental compensation and network mapping pitching moment coefficient residual and relative error. (**a**) Residual curve of incremental compensation and network mapping pitching moment coefficient; (**b**) relative error curve of incremental compensation and network mapping pitching moment coefficient.

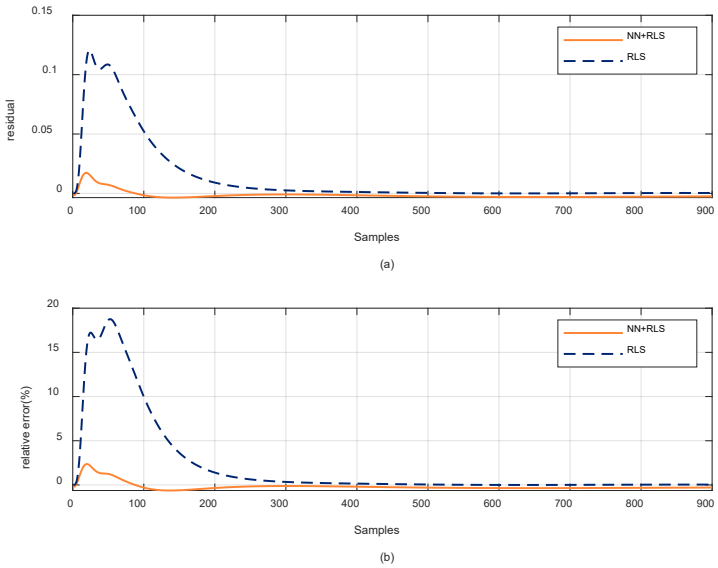


Figure 10. Comparison results of incremental compensation and network mapping lift coefficient residual and relative error. (**a**) Residual curve of incremental compensation and network mapping lift coefficient; (**b**) relative error curve of incremental compensation and network mapping lift coefficient.

Table 3. Comparison table of identification average time between RLS and RLS + NN.

	Pitching Moment Coefficient (s)	Lift Coefficient (s)
RLS	0.438	0.576
RLS + NN	0.0647	0.0753

6.3. Simulation and Analysis of an Iterative Learning Correction Method Based on a Data-Driven NN Model

This section considers using the flight test data of a real fixed-wing UAV to modify the weight matrix of the original NN offline so as to realize the iterative learning of the offline network. By updating and iterating the weight matrix of the NN, the fitting accuracy of the NN to the force coefficient and moment coefficient is greatly improved, thereby realizing lifelong learning for complex dynamic models. Figures 11 and 12 are respectively the identification results of the pitching moment coefficient and lift coefficient after correction by the neural network.

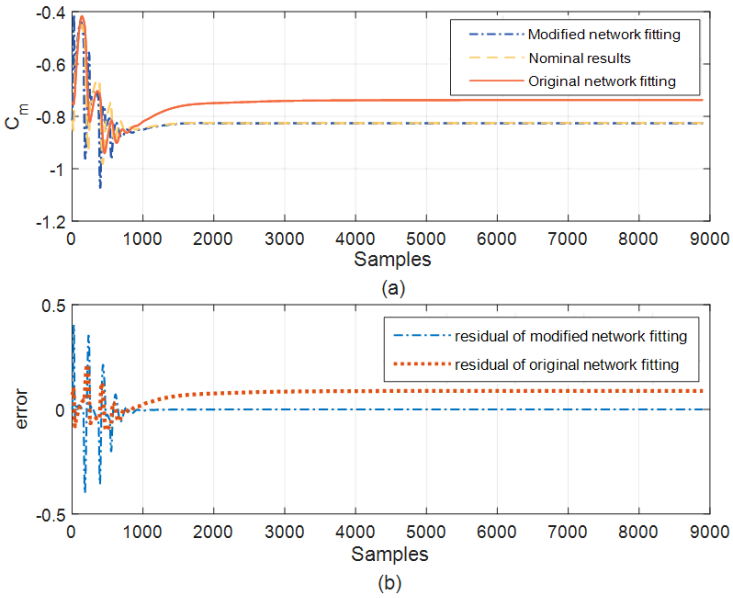


Figure 11. The results of fitting the pitching moment coefficient between the modified network and the original network. (**a**) Fitting pitching moment coefficient curve between the modified network and the original network; (**b**) the residual curve of the modified network and the original network fitting value and nominal value.

The training data of the original NN were obtained through CFD tests, and there is a big difference with the real flight data. Therefore, it is necessary to correct the original NN weight matrix established offline through the network weight iterative correction algorithm based on the real flight test data. This adjustment ensures that the NN model established by it can better reflect the real dynamic model. According to the simulation results as shown in Table 4, compared with the original NN, the variance of the mapping value of the modified network to the moment coefficient is reduced by 83.68%, and the residual error is reduced by 69.23%. After the correction, the variance of the mapping value of the network to the force coefficient is reduced by 39.63%, and the residual error is reduced by 89.89%.

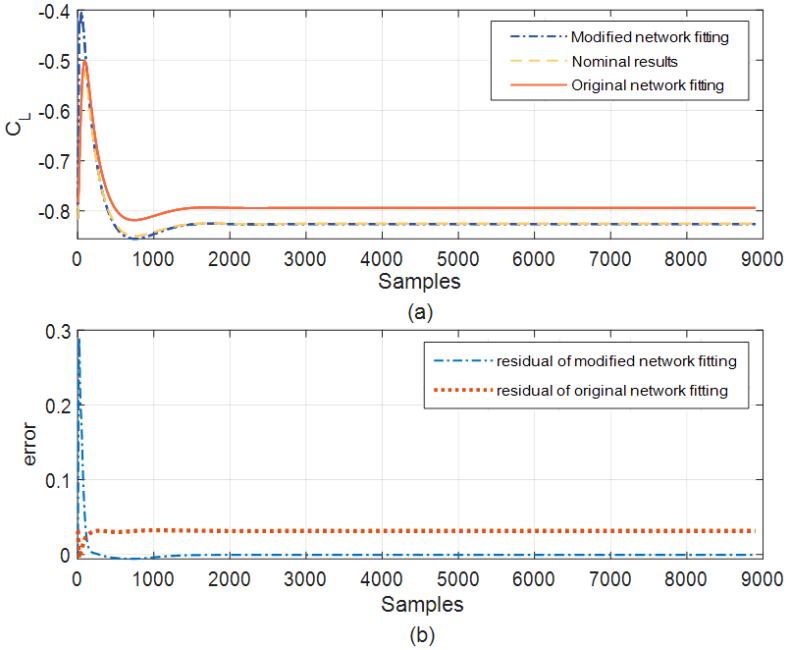


Figure 12. The results of fitting the lift coefficient between the modified network and the original network. (**a**) Fitting lift coefficient curve between the modified network and the original network; (**b**) the residual curve of the modified network and the original network fitting value and nominal value.

Table 4. Comparison of the accuracy of identifying force coefficient and moment coefficient before and after NN correction.

Serial Number	Moment Coefficient				Force Coefficient			
	Original Network		Corrected Network		Original Network		Corrected Network	
	Variance	Residual	Variance	Residual	Variance	Residual	Variance	Residual
1	0.2137	0.1713	0.0391	0.0514	0.0314	0.0313	0.0177	0.0041
2	0.2353	0.1747	0.0314	0.0551	0.0354	0.0414	0.0211	0.0034
3	0.2200	0.1704	0.0371	0.0541	0.0305	0.0351	0.0204	0.0035
4	0.2259	0.1690	0.0369	0.0495	0.0321	0.0393	0.0184	0.0039
5	0.2325	0.1807	0.0397	0.0563	0.0322	0.0409	0.0198	0.0041
Average	0.2255	0.1732	0.0368	0.0533	0.0323	0.0376	0.0195	0.0038

7. Conclusions

In this paper, an intelligent identification method of aerodynamic parameters based on deep learning network correction and compensation is used to identify the pitching moment coefficient and lift coefficient. The CFD simulation data of fixed-wing UAVs are used to train the deep learning network offline. The network training dataset pair consists of flight state data, force coefficients, and moment coefficients. Secondly, based on the recursive orthogonal time-domain algorithm, the recursive least squares incremental identification compensation is performed according to the offline trained network mapping force coefficient, moment coefficient, and coefficients. Finally, the method of establishing a loss function is used to correct the output layer to the hidden layer and the weights from the hidden layer to the input layer. The deep learning networks are corrected offline according to the data collected in the real flight state so that the corrected NN mapping force coefficient and torque coefficient are closer to the real values.

Since the model established based on ground simulation data has errors and it is difficult to simulate uncertain factors such as complex real-world environments, this method improves the control performance of the UAV. It improves the fitting accuracy of network mapping force coefficients and torque coefficients. At the same time, the problem

of reduced control accuracy caused by the existence of time-varying coupling is solved. In addition, flight tests have accumulated a large number of test flight data. It is of great significance to use test flight data to achieve model correction in real flight environments, laying the foundation for obtaining more accurate complex control models.

Author Contributions: Conceptualization, D.Y. and J.Z.; methodology, J.Z.; software, J.Z.; validation, K.L.; formal analysis, D.Y., J.Z. and K.L.; investigation, D.Y. and J.Z.; resources, D.Y. and J.Z.; data curation, J.Z. and K.L.; writing—original draft preparation, J.Z.; writing—review and editing, J.Z., K.L. and J.L.; visualization, J.Z.; supervision, D.Y.; project administration, J.Z.; funding acquisition, K.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China, grant number U2141229 and foundation under grant JCJQ, grant number 2019-JCJQ-DA-001-131.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Silvestrini, S.; Lavagna, M. Deep Learning and Artificial NNs for Spacecraft Dynamics, Navigation and Control. *Drones* **2022**, *6*, 270. [CrossRef]
2. Dou, L.; Du, M.; Zhang, X.; Wang, Y. Aerodynamic parameter identification of the RLV reentry process based on the EM-EKF algorithm. *J. Tianjin Univ.* **2019**, *52*, 1285–1292.
3. Snyder, S.; Bacon, B.; Morelli, E.A.; Frost, S.; Teubert, C.; Okolo, W. Online control design for learn-to-fly. In Proceedings of the 2018 Atmospheric Flight Mechanics Conference, Kissimmee, FL, USA, 8–12 January 2018; pp. 1–27.
4. Morelli, E.A. Determining Aircraft Moments of Inertia from Flight Test Data. *AIAA SciTech 2021 Forum* **2021**, *45*, 4–14.
5. Jafari, S.; Ioannou, P.; Rudd, L.E. What is L1 adaptive control. In Proceedings of the AIAA Guidance, Navigation, and Control (GNC) Conference, Portland, Oregon, 10–15 August 2013; p. 4513.
6. Jiang, Y.M.; Wang, C.Q.; Xu, C. Online Identification of Aircraft Model Parameters Based on Recursive Least Squares Method. *Control Inf. Technol.* **2019**, *4*, 58–64.
7. Yang, G.H.; Du, L.F.; Li, H.; Liu, X.D. Parameter Identification and Adaptive Control of Aircraft Based on BP NN. *Aerosp. Control* **2021**, *39*, 3–7.
8. Zhang, Y.; Tang, J.; Huang, H. An Intelligent Correction Method of Aerodynamic Parameters Based on Machine Learning. *Aerosp. Control* **2021**, *39*, 49–51.
9. Li, Z.Q. A High Angle of Attack Dynamical System Identification Algorithm Based on LS-SVM. *Comput. Simul.* **2019**, *1*, 28–30.
10. Wang, J.; Liang, H.Z.; Wang, J.C.; Li, Q.; Wang, L.L. Online Intelligent Identification Method of Hypersonic Vehicles Dynamic System. *Tactical Missile Technol.* **2019**, *5*, 19–22.
11. Pu, J.L.; Han, Y.P.; Zhang, L. Research on Intelligent Online Identification Technology for Aerodynamic Parameters of Aircraft. *Astronaut. Syst. Eng. Technol.* **2018**, *2*, 1–9.
12. Tai, S.; Wang, L.; Wang, Y.; Lu, S.; Bu, C.; Yue, T. Identification of Lateral-Directional Aerodynamic Parameters for Aircraft Based on a Wind Tunnel Virtual Flight Test. *Aerospace* **2023**, *10*, 350. [CrossRef]
13. Wang, L.; Zhao, R.; Xu, K.; Zhang, Y.; Yue, T. Identification and Modeling Method of Longitudinal Stall Aerodynamic Parameters of Civil Aircraft Based on Improved Kirchhoff Stall Aerodynamic Model. *Aerospace* **2023**, *10*, 333. [CrossRef]
14. Xue, J.; Yang, Y.J.; Liu, Y.; Li, L.T. Lateral Control of UAV Based on L1 Adaptive Control. *J. Northwest Poly Tech. Univ.* **2015**, *33*, 40–44.
15. Zaheer, S.; Anjum, N.; Hussain, S.; Algarni, A.D.; Iqbal, J.; Bourouis, S.; Ullah, S.S. A Multi Parameter Forecasting for Stock Time Series Data Using LSTM and Deep Learning Model. *Mathematics* **2023**, *11*, 590. [CrossRef]
16. Sieberling, S.; Chu, Q.P.; Mulder, J.A. Robust Flight Control Using Incremental Nonlinear Dynamic Inversion and Angular Acceleration Prediction. *J. Guid. Control Dyn.* **2010**, *33*, 1732–1742. [CrossRef]
17. Li, Q.; Liu, X. Simulation research on system parameter identification of improved Kalman filtering algorithm. *Comput. Simul.* **2012**, *29*, 172–175.
18. Su, Z.Y. Development and Application of Aerodynamic Parameter Identification Technology for Aircraft. *Sci. Technol. Innov.* **2018**, *29*, 64–65.
19. Snyder, S. Autopilot Design with Learn-to-Fly. *AIAA SciTech 2020 Forum* **2020**, *1*, 25–29.
20. Murphy, P.C.; Hatke, D.B.; Aubuchon, V. Preliminary Steps in Developing Rapid Aero Modeling Technology. *AIAA SciTech Forum* **2020**, 6–10.
21. Gong, X.Y.; Fu, W.; Bian, X.A.; Fei, J.T. Adaptive Backstepping Terminal Sliding Mode Control of Nonlinear System Using Fuzzy Neural Structure. *Mathematics* **2023**, *11*, 1094. [CrossRef]
22. Cao, Y.H.; Tan, W.Y. The Effects of Icing on Aircraft Longitudinal Aerodynamic Characteristics. *Mathematics* **2020**, *8*, 1171. [CrossRef]

23. Wang, L.; Zhang, Z.; Zhu, Q. Automatic Flight Control Design Considering Objective and Subjective Risks During Carrier Landing. *The Institution of Mechanical Engineers, Part I: J. Syst. Control Eng.* **2020**, *234*, 446–461. [CrossRef]
24. Lee, B.; Saj, V.; Benedict, M. Machine Learning Vision and Nonlinear Control Approach for Autonomous Ship Landing of Vertical Flight Aircraft. In Proceedings of the 77th Annual National Forum of the Vertical Flight Society, Online, 10–14 May 2021.
25. Leshikar, C.; Gosnell, S.; Gomez, E. System Identification Flight Testing of Inverted V-Tail Small Unmanned Air System. In Proceedings of the AIAA SciTech 2022 Forum, San Diego, CA, USA, 3–7 January 2022.
26. Ingen, J.V.; Visser, C.C.; Pool, D.M. Stall Model Identification of a Cessna Citation II from Flight Test Data Using Orthogonal Model Structure Selection. In Proceedings of the AIAA Scitech 2021 Forum, Online, 11–15 & 19–21 January 2021.
27. Boschetti, P.J.; Neves, C.; Ramirez, P.J. Nonlinear Aerodynamic Model in Dynamic Ground Effect at High Angles of Attack. In Proceedings of the AIAA SCITECH 2022 Forum, San Diego, CA, USA, 3–7 January 2022.
28. Wang, L.; Zhang, N.; Liu, H.; Yue, T. Stability characteristics and airworthiness requirements of blended wing body aircraft with podded engines. *Chin. J. Aeronaut.* **2022**, *35*, 77–86. [CrossRef]
29. Wang, L.; Zuo, X.; Liu, H.; Yue, T.; Jia, X.; You, J. Flying qualities evaluation criteria design for scaled-model aircraft based on similarity theory. *Aerosp. Sci. Technol.* **2019**, *90*, 209–221. [CrossRef]
30. Shafi, I.; Mazhar, M.F.; Fatima, A.; Alvarez, R.M.; Miró, Y.; Espinosa, J.C.M.; Ashraf, I. Deep Learning-Based Real Time Defect Detection for Optimization of Aircraft Manufacturing and Control Performance. *Drones* **2023**, *7*, 31. [CrossRef]
31. Dai, W.; Liang, K.; Wang, B. State Monitoring Method for Tool Wear in Aerospace Manufacturing Processes Based on a Convolutional Neural Network (CNN). *Aerospace* **2021**, *8*, 335. [CrossRef]
32. Zheng, Q.; Yang, M.; Yang, J.; Zhang, Q.; Zhang, X. Improvement of generalization ability of deep CNN via implicit regularization in two-stage training process. *IEEE Access* **2018**, *6*, 15844–15869. [CrossRef]