energies

MDPI

Article

A State Space Modeling Method for Aero-Engine Based on AFOS-ELM

Hongyi Chen, Qiuhong Li *, Shuwei Pang and Wenxiang Zhou

Jiangsu Province Key Laboratory of Aerospace Power System, College of Energy and Power Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China; hongyi_c@nuaa.edu.cn (H.C.); anglepsw@outlook.com (S.P.); zhouwx@nuaa.edu.cn (W.Z.)

* Correspondence: lqh203@nuaa.edu.cn

Abstract: State space models (SSMs) are important for multi-variable performance analysis and controller design of aero-engines. In order to solve the problems of the traditional state space modeling methods that rely on component-level models (CLMs) and cannot be carried out in real time, an aero-engine state space modeling method based on adaptive forgetting factor online sequential extreme learning machine (AFOS-ELM) is proposed in this paper. The structure of the extreme learning machine (ELM) is determined according to the form of the state space model, and the inverse-free ELM algorithm is used to automatically select the appropriate number of hidden nodes to improve the efficiency of offline initialization. The focus of the ELM on current operation performance is enhanced by the adaptive renewed forgetting factor, which reduces the impact of aero-engine history and deviated data on the current output and improves the accuracy of the model. Then, according to the analytical equation of the ELM model, the state space model of an aero-engine at each sampling time is obtained by using the partial derivative method. The simulation results based on engine test data show that the real-time performance and accuracy of the state space model established online in this paper can meet the needs of aero-engine control system requirement.

Keywords: aero-engine; state space model; forgetting factor; online sequential extreme learning machine (OS-ELM)

Citation: Chen, H.; Li, Q.; Pang, S.; Zhou, W. A State Space Modeling Method for Aero-Engine Based on AFOS-ELM. *Energies* **2022**, *15*, 3903. https://doi.org/10.3390/en15113903

Academic Editor: Mojtaba Ahmadieh Khanesar

Received: 9 May 2022
Accepted: 24 May 2022
Published: 25 May 2022

1. Introduction

As a kind of linear mathematical model, the state space model (SSM) is widely used in multi-variable system analysis and controller design processes, especially when the knowledge of modern control theory is applied [1]. Furthermore, in the area of model-predictive control, performance-seeking control, and health evaluation, using SSM can greatly improve the real-time performance of the control system and simplify the process of the optimization and the estimation [2–4].

SSM also plays an important role in the design of aero-engine control systems. Therefore, it is of great significance to research its modeling method, especially the method that can accurately capture the individual aero-engine dynamics in real time.

In view of the strong nonlinearity of an aero-engine, its mechanism model (component-level model, CLM) is very complex [5]; the analytical SSM cannot be derived from it. At present, there are two main modeling methods for aero-engine SSM, namely, fitting method and partial derivative method [6–8]. Both methods belong to the small disturbance method, and the parameters of the SSM are obtained based on the responses of the CLM to the separate small disturbance of the SSM inputs (and states for partial derivative method). The fitting method is carried out with the principle that the linear system output should be consistent with that of the nonlinear system under the same separate small disturbance inputs, and its model parameters are determined by minimizing the errors between the SSM model outputs and the CLM outputs. Therefore, optimization methods

are widely used in the parameters identification process of the fitting method. For example, the SSM of certain turbofan engines is established by the least square optimization in the literature [9], and the model parameters are optimized by genetic algorithm in the literature [10]. The partial derivative method applies disturbances to the control variables and state variables, respectively, on the CLMs, and the partial derivative calculation in the SSM is approximated by the output difference [11,12].

However, both methods have some defects. The fitting method needs to collect the engine dynamic response data offline, and the partial derivative method needs to call the engine CLMs repeatedly for differential calculation. They both cannot be realized in real time and both depend on the CLM. Aero-engine CLM is a complex model, which is composed of multiple component models, co-working equations, and equation solving process [13–15]. Due to the assumptions in modeling, the accuracy of CLM is difficult to ensure, and the CLM is mostly established based on the component characteristics at rated condition, which makes it hard to reflect the individual differences and performance degradation of aero-engine. The fitting method and partial derivative method based on the rated CLMs cannot characterize the accurate characteristics of a specific engine. Although the data used in fitting method can be obtained from the engine test, for multivariable control systems, the test data are a set of responses to the simultaneously varying input variables, while the data obtained from the component-level model are several sets of responses to separately varying input variables, which means the test data cannot clearly reflect the characteristics between the inputs and the outputs, and the model built based on the test data lacks reliability even if it can fit the output curve of the selected test point. In addition, the aero-engine works in a wide flight envelope, and the working state spans from start to maximum reheat rating. The traditional method can only be applied to limited steady-state points. For other working points, the SSMs are more dependent on interpolation or parameter scheduling methods, such as TS fuzzy model [16,17] and equilibrium manifold expansion (EME) model [18,19]. Compared with the SSMs directly established at the corresponding points, the accuracy of the model obtained by interpolation or other methods is relatively lower.

Pang et al. present an online exact partial derivation calculation method to solve this problem, which provides a component-level derivative model of engine, and the partial derivations are calculated by digital solution of the component-level derivative model [20,21]. This method can obtain more accurate SSMs in the whole envelope at any working state of the engine, and the SSM can adapt to the engine degradation with the adaptive CLM used. However, this modeling method is also based on the CLM, the calculation process is complex, and the model accuracy will also be affected by the accuracy of CLM.

With the development of artificial intelligence technology, data-driven methods have developed rapidly in recent years. In the field of aero-engine, data-driven methods have been used in degradation prediction, mathematical model establishment, and so on [22,23]. Compared with the traditional modeling methods that rely on CLMs, data-driven modeling method avoids the complex component-level modeling process and eliminates the influence of CLMs modeling accuracy on SSM. Based on the idea of data-driven, some scholars use an MGD neural network modeling method to establish the adaptive dynamic model of turbofan engine [24,25]. The improved MGD neural network based on batch training is used to establish the onboard model of turbofan engine, but the training process is time-consuming and must be carried out offline, which means the model cannot adapt to the engine degradation. The data-driven modeling method combined with intelligent algorithm is more efficient, but there is still little research on aero-engine state space modeling. Only in reference [23] is a linear parameter-varying (LPV) modeling method of turboshaft engine proposed by using a special construction neural network.

This paper proposes an aero-engine modeling method based on adaptive forgetting factor online sequential extreme learning machine (AFOS-ELM), which aims to model the state space equation online based on data-driven method. The main contributions of this paper are as follows: (1) An online state space modeling method based on AFOS-ELM is

proposed, which can be carried out in real time with the data gathered online. According to the form of SSM, the inputs and outputs of neural network are determined. By using the online training ability of online sequential extreme learning (OS-ELM), the parameters of the network are updated in real time, and the analytical model of the AFOS-ELM is obtained. Then, the SSM is derived by applying the partial derivative method to the analytical model of the AFOS-ELM. (2) The inverse-free extreme learning machine (ELM) is used to initialize the neural network model offline. The number of hidden nodes and the initial weights of the ELM are determined by the inverse-free method to improve the efficiency and accuracy of the offline training. (3) The neural network is trained by the AFOS-ELM method which can extract the current working state characteristics of the engine and improve the accuracy of the SSM model. (4) Compared with the traditional linearization modeling method, the method proposed in this paper can obtain the SSM based on the aero-engine test data at each sampling time, which can better reflect the individual characteristics of the engine.

The paper is organized as follows. Section 2 introduces the inverse-free ELM used in this paper. Section 3 introduces the AFOS-ELM method used in the modeling process. Section 4 introduces the neural network structure for state space modeling and the deduction of SSM. Section 5 gives the simulation results of the modeling. Section 6 concludes this paper.

2. Inverse-Free ELM

Extreme learning machines are feedforward neural networks which were first proposed by Professor Huang of Nanyang University of Technology in Singapore in 2006 [26]. Compared with traditional machine learning algorithms such as backpropagation (BP) neural network, ELM has faster learning speed and similar generalization performance [27]. The traditional neural network has the problem that the number of hidden nodes needs to be determined by trial. The inverse-free ELM can solve this problem, and it has aroused many concerns [28,29]. It adds hidden nodes through iterative method, which can save the trail time and improve the calculation efficiency.

In the ELM algorithm, given the training set $\chi = \{\boldsymbol{x}_{ELM,i}, \boldsymbol{T}_i\}_{i=1}^{q}$, where input $\boldsymbol{x}_{ELM,i} = [x_{ELM,i1}, \ldots, x_{ELM,iN}]^{\mathrm{T}} \in \mathbf{R}^N$, target output $\boldsymbol{T}_i = [T_{i1}, \ldots, T_{iM}]^{\mathrm{T}} \in \mathbf{R}^M$.

In ELM with l hidden nodes, the output corresponding to the i^{th} input can be described as

$$\boldsymbol{y}_{ELM,i} = \sum_{j=1}^{l} \boldsymbol{\beta}_j f(\boldsymbol{W}_j \boldsymbol{x}_{ELM,i} + b_j) \tag{1}$$

where $\boldsymbol{W}_j = [W_{j1}, W_{j2}, \ldots, W_{jN}]$ are the weights connecting the hidden layer node j and the input layer nodes. b_j is the bias of the hidden layer node j. $\boldsymbol{\beta}_j = [\beta_{j1}, \ldots, \beta_{jM}]$ are the weights connecting the hidden layer node j and the output layer nodes. $f(x)$ is the hidden layer activation function.

The learning target of neural network is to minimize the output error, so the cost function of ELM can be described as

$$\min_{\boldsymbol{\beta}} J = \left\| \boldsymbol{T} - \boldsymbol{H}\boldsymbol{\beta} \right\| \tag{2}$$

where

$$\begin{cases} \boldsymbol{H} = \left[h_1, \ldots, h_l \right]_{q \times l}^{\mathrm{T}} \\ \boldsymbol{\beta} = [\beta_1, \ldots, \beta_l]_{q \times M}^{\mathrm{T}} \end{cases} \tag{3}$$

$\boldsymbol{H}$ is the hidden layer output matrix, $h_i = f(\boldsymbol{W}\boldsymbol{x}_{ELM,i} + b)$, $i = 1, 2, \cdots, l$.

Output weight $\boldsymbol{\beta}$ can be obtained from the following formula

$$\boldsymbol{\beta} = \boldsymbol{T}\boldsymbol{H}^{+} = \boldsymbol{T}\boldsymbol{H}^{T}(\boldsymbol{H}\boldsymbol{H}^{T})^{-1} \tag{4}$$

where $\boldsymbol{H}^{+}$ is the generalized inverse of $\boldsymbol{H}$.

In order to avoid singular matrix or overfitting, Tikhonov regularization method was used, so Equation (4) becomes

$$\boldsymbol{\beta} = \boldsymbol{T}\boldsymbol{H}^{T}(\boldsymbol{H}\boldsymbol{H}^{T}+\delta^{2}I)^{-1} \tag{5}$$

where $\delta^{2} > 0$ is the regularization factor. Equation (4) can be regarded as a special form of Equation (5) where δ^{2}=0.

If an additional hidden node is added to the ELM with l hidden nodes, the hidden layer output matrix of all l+1 nodes becomes

$$\boldsymbol{H}_{l+1} = \begin{bmatrix} \boldsymbol{H}_{l} \\ h_{l+1} \end{bmatrix} \tag{6}$$

where $h_{l+1} = f(\boldsymbol{W}_{l+1}\boldsymbol{x}_{ELM} + b_{l+1})$ is the output of the (l+1)$^{\text{th}}$ hidden node. The input weight vector $\boldsymbol{W}_{l+1}$ and bias b_{l+1} corresponding to the newly added node are randomly generated.

With l+1 hidden layer nodes, the weights of the ELM output layer can be calculated as

$$\boldsymbol{\beta}_{l+1} = \boldsymbol{T}\boldsymbol{H}_{l+1}{}^{\mathrm{T}}(\boldsymbol{H}_{l+1}\boldsymbol{H}_{l+1}{}^{T}+\delta^{2}\boldsymbol{I})^{-1} = \boldsymbol{T}\boldsymbol{O}_{l+1} \tag{7}$$

where $\boldsymbol{O}_{l+1}$=$\boldsymbol{H}_{l+1}{}^{\mathrm{T}}(\boldsymbol{H}_{l+1}\boldsymbol{H}_{l+1}{}^{T}+\delta^{2}\boldsymbol{I})^{-1}$

We rewrite $\boldsymbol{O}_{l+1}$ as [30].

$$\boldsymbol{O}_{l+1} = \begin{bmatrix} \tilde{O}_{l} & o_{l+1} \end{bmatrix} \tag{8}$$

where

$$o_{l+1} = \frac{h_{l+1} - \boldsymbol{O}_{l}\boldsymbol{H}_{l}h_{l+1}}{h_{l+1}^{T}h_{l+1} + \delta^{2} - h_{l+1}^{T}\boldsymbol{O}_{l}\boldsymbol{H}_{l}h_{l+1}} \tag{9}$$

$$\tilde{\boldsymbol{O}}_{l} = \boldsymbol{O}_{l} - \boldsymbol{o}_{l+1}h_{l+1}^{T}\boldsymbol{O}_{l} \tag{10}$$

Then

$$\boldsymbol{O}_{l+1} = \begin{bmatrix} \tilde{\boldsymbol{O}}_{l} & \boldsymbol{o}_{l+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{O}_{l} - \boldsymbol{o}_{l+1}h_{l+1}^{T}\boldsymbol{O}_{l} & o_{l+1} \end{bmatrix} \tag{11}$$

According to Equation (7),

$$\boldsymbol{\beta}_{l+1} = \boldsymbol{T}\boldsymbol{O}_{l+1} = \begin{bmatrix} \boldsymbol{T}\tilde{\boldsymbol{O}}_{l} & \boldsymbol{T}\boldsymbol{o}_{l+1} \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{\beta}}_{l} & \hat{\beta}_{l+1} \end{bmatrix} \tag{12}$$

where

$$\tilde{\boldsymbol{\beta}}_{l} = \boldsymbol{\beta}_{l} - \hat{\beta}_{l+1}h_{l+1}^{T}\boldsymbol{O}_{l} \tag{13}$$

$$\hat{\beta}_{l+1} = \boldsymbol{T}\boldsymbol{o}_{l+1} \tag{14}$$

Then, the process of updating the output weights can be described as

$$\boldsymbol{\beta}_{l+1} = \begin{bmatrix} \tilde{\boldsymbol{\beta}}_{l} & \hat{\beta}_{l+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\beta}_{l} - \hat{\beta}_{l+1}\boldsymbol{h}_{l+1}^{T}\boldsymbol{O}_{l} & \boldsymbol{T}\boldsymbol{o}_{l+1} \end{bmatrix} \tag{15}$$

The above calculation process is repeated, and gradually increases the hidden layer nodes until the network training error satisfies the requirement or the number of hidden nodes reaches the maximum value. This method greatly saves the trail time and workload of repeated ELM network training used in adjusting the hidden layer nodes.

3. Adaptive Forgetting Factor OS-ELM (AFOS-ELM)

3.1. OS-ELM Neural Network

He ELM neural network belongs to offline batch processing algorithm, and the training data are obtained offline. When new data arrive, the model will discard the results of previous training and be completely rebuilt, which arouses the increase of the calculation burden. In order to meet the needs of real-time dynamic calculation in various fields, OS-ELM came into being, which has a fast learning speed and can recursively renew the output weight with the highest calculation efficiency, which makes it suitable for online learning [31].

The recursive learning output weight of OS-ELM at time k+1 can be expressed as

$$\boldsymbol{\beta}(k+1)=\boldsymbol{\beta}(k)+\left[\boldsymbol{T}(k+1)-\boldsymbol{\beta}(k)\boldsymbol{H}(k+1)\right]\boldsymbol{H}^{\mathrm{T}}(k+1)\boldsymbol{P}(k+1) \tag{16}$$

where the initial matrixes are obtained by the inverse-free OS-ELM method.

$$\boldsymbol{P}(0)=(\delta^2\boldsymbol{I}+\boldsymbol{H}_0\boldsymbol{H}_0^T)^{-1} \tag{17}$$

$$\boldsymbol{P}(k+1)=\boldsymbol{P}(k)-\boldsymbol{P}(k)\boldsymbol{H}(k+1)\left[\boldsymbol{I}+\boldsymbol{H}^{\mathrm{T}}(k+1)\boldsymbol{P}(k)\boldsymbol{H}(k+1)\right]^{-1}\boldsymbol{H}^{\mathrm{T}}(k+1)\boldsymbol{P}(k) \tag{18}$$

With the data input gradually increasing, the output weights $\boldsymbol{\beta}$ are updated online by using Equations (16)–(18). This is the online learning stage of OS-ELM.

3.2. AFOS-ELM Neural Network

Although OS-ELM avoids the offline repeated training of the past samples and achieves the online learning of the new data, the online modeling accuracy is not significantly improved compared with the traditional offline ELM algorithm. The traditional OS-ELM algorithm does not consider the temporal validity of the samples [32]. An aeroengine is a complex aerothermodynamic system with large flight envelope. When the online neural network is applied, the old samples cannot reflect the current dynamic characteristics of the engine over time. If these failure samples are not processed, the neural network cannot reflect the current dynamic characteristics well. Therefore, this paper adopts an OS-ELM with memory mechanism, which uses adaptive forgetting factor to reduce the influence of past failure samples on the current output, so as to improve the modeling accuracy.

OS-ELM with memory mechanism has the same updating principle as the traditional OS-ELM. It only needs to update the output weight at time $k+1$.

The cost function with forgetting factor can be described as

$$J\left[\boldsymbol{\beta}(k)\right]=\sum_{i=1}^{k}\lambda^{k-i}g\left(\boldsymbol{T}(k)-\boldsymbol{H}(k)\boldsymbol{\beta}(k)\right)+\frac{1}{2}\delta^2\left\|\boldsymbol{\beta}(k)\right\| \tag{19}$$

where $g(\,)$ is the least square cost function [33], and $\lambda\in(0,1]$ is the forgetting factor.

If λ is smaller, the proportion of the past samples impact on the cost function is lower, and the influence of past samples on the current outputs will be decreased greatly. According to Equation (19), applying the memory mechanism, the recursive formula of the output weights can be described as [34]

$$\boldsymbol{\beta}(k+1)=\boldsymbol{\beta}(k)-\delta^2(1\text{-}\lambda)\boldsymbol{P}(k+1)\boldsymbol{\beta}(k)+\left[\boldsymbol{T}(k+1)-\boldsymbol{\beta}(k)\boldsymbol{H}(k+1)\right]\boldsymbol{H}^{\mathrm{T}}(k+1)\boldsymbol{P}(k+1) \tag{20}$$

where

$$\boldsymbol{P}(k+1)=\boldsymbol{P}^{*}-\boldsymbol{P}^{*}\boldsymbol{H}^{\mathrm{T}}(k+1)\boldsymbol{H}(k+1)\boldsymbol{P}^{*}\left[1+\boldsymbol{H}(k+1)\boldsymbol{P}(k+1)\boldsymbol{H}^{\mathrm{T}}(k+1)\right] \quad (21)$$

$$\boldsymbol{P}^{*}=\lambda^{-1}\boldsymbol{P}(k)-\delta\frac{\lambda}{\lambda-1}\boldsymbol{P}(k)\left[\boldsymbol{I}+\delta\frac{1-\lambda}{\lambda}\boldsymbol{P}(k)\right]^{-1}\boldsymbol{P}(k) \quad (22)$$

Due to the influence of the measurement noise and the sudden change of the system dynamic, the network output errors show great difference. An adaptive forgetting factor based on the relative error of neural network output is adopted. The forgetting factor of the sample with larger output error is relatively smaller, which can strengthen the forgetting effect and weaken its influence on future output.

The maximum relative output error of multi-output neural network can be described as

$$e_{\mathrm{ELM}}(k)=\max\left(\left|\frac{\boldsymbol{y}_{\mathrm{ELM}}(k)-\boldsymbol{T}(k)}{\boldsymbol{T}(k)}\right|\right) \quad (23)$$

The forgetting factor is adjusted based on the relative error of network output. The calculation method of forgetting factor can be expressed as follows:

$$\begin{cases}\lambda(k)=\lambda_{\max}-\theta\cdot e_{\mathrm{ELM}}(k)\\ if\ \lambda(k)<\lambda_{\min},\ \lambda(k)=\lambda_{\min}\end{cases} \quad (24)$$

where $\lambda_{\max}$ is the maximum value of forgetting factor, which is generally taken as 1 or slightly less than 1. θ is the proportional coefficient. The larger the θ is, the greater the influence of relative error on the forgetting factor. If the forgetting factor is less than the minimum value $\lambda_{\min}$, then it is set to $\lambda_{\min}$.

If the system dynamics change greatly, the error $\boldsymbol{e}_{\mathrm{ELM}}(k)$ will increase significantly and the forgetting factor will decrease, so that the data will be forgotten faster. At the same time, when the neural network tends to converge and the error gradually decreases, $e_{\mathrm{ELM}}(k)$ will approach 0. and $\lambda(k)$ approaches $\lambda_{\max}$. The steady-state accuracy during the online operation can be guaranteed.

The forgetting factor is widely used in the OS-ELM [32–34] and can improve the modeling accuracy, more or less, in the practical problems. It can be said that it is generally effective in online learning. With the adaptive forgetting factor, the adaptive performance of the network on the complex dynamic system is enhanced, and both the static and dynamic accuracy of the system outputs are enhanced.

4. Online State Space Modeling Based on AFOS-ELM

The nonlinear state space model of the aero-engine can be described as follows:

$$\begin{cases}\boldsymbol{x}(k+1)=f_1(\boldsymbol{x}(k),\boldsymbol{u}(k))\\ \boldsymbol{y}(k)=f_2(\boldsymbol{x}(k),\boldsymbol{u}(k))\end{cases} \quad (25)$$

where $\boldsymbol{x}=\left[x_1,\cdots,x_n\right]^T$ is the n-dimensional state vector, $\boldsymbol{y}=\left[y_1,\cdots,y_m\right]^T$ is the m-dimensional output vector, and $\boldsymbol{u}=\left[u_1,\cdots,u_r\right]^T$ is the r-dimensional input vector.

The linearized state space model of the system at the working point (x_{t0},u_{t0},y_{t0}) at time t can be expressed as

$$\begin{cases}\Delta\boldsymbol{x}_t(k+1)=\boldsymbol{A}_t\Delta\boldsymbol{x}_t(k)+\boldsymbol{B}_t\Delta\boldsymbol{u}_t(k)\\ \Delta\boldsymbol{y}_t(k)=\boldsymbol{C}_t\Delta\boldsymbol{x}_t(k)+\boldsymbol{D}_t\Delta\boldsymbol{u}_t(k)\end{cases} \quad (26)$$

where Δ is the increment symbol, $\Delta \boldsymbol{x}_t(k) = x(k) - x_{t0}$, $\Delta \boldsymbol{u}_t(k) = u(k) - u_{t0}$, $\Delta \boldsymbol{y}_m(k) = y(k) - y_{m0}$, where t represents the engine operation time when the state space model is established, and k is the variable used to express a discrete time system.

The parameters of the state space matrixes in Equation (26) can be expressed in the form of partial derivative, and the matrixes $\boldsymbol{A}_t$, $\boldsymbol{B}_t$, $\boldsymbol{C}_t$, $\boldsymbol{D}_t$ are also called Jacobian matrixes.

$$\begin{aligned} \boldsymbol{A}_t &= \left[\begin{array}{ccc} \frac{\partial x_1(k+1)}{\partial x_1(k)} & \cdots & \frac{\partial x_1(k+1)}{\partial x_n(k)} \\ \vdots & \ddots & \vdots \\ \frac{\partial x_n(k+1)}{\partial x_1(k)} & \cdots & \frac{\partial x_n(k+1)}{\partial x_n(k)} \end{array}\right]_{k=t} \quad \boldsymbol{B}_t = \left[\begin{array}{ccc} \frac{\partial x_1(k+1)}{\partial u_1(k)} & \cdots & \frac{\partial x_1(k+1)}{\partial u_r(k)} \\ \vdots & \ddots & \vdots \\ \frac{\partial x_n(k+1)}{\partial u_1(k)} & \cdots & \frac{\partial x_n(k+1)}{\partial u_r(k)} \end{array}\right]_{k=t} \\ \boldsymbol{C}_t &= \left[\begin{array}{ccc} \frac{\partial y_1(k)}{\partial x_1(k)} & \cdots & \frac{\partial y_1(k)}{\partial x_n(k)} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_m(k)}{\partial x_1(k)} & \cdots & \frac{\partial y_m(k)}{\partial x_n(k)} \end{array}\right]_{k=t} \quad \boldsymbol{D}_t = \left[\begin{array}{ccc} \frac{\partial y_1(k)}{\partial u_1(k)} & \cdots & \frac{\partial y_1(k)}{\partial u_r(k)} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_m(k)}{\partial u_1(k)} & \cdots & \frac{\partial y_m(k)}{\partial u_r(k)} \end{array}\right]_{k=t} \end{aligned} \tag{27}$$

To obtain the linear state space model with AFOS-ELM, the data used to train the ELM are normalized as

$$\overline{x} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{28}$$

The inverse-free ELM neural network described in Section 2 is used in the ELM initialization. The inputs of the AFOS-ELM are the input variables $\boldsymbol{u}(k)$ and the state variables $\boldsymbol{x}(k)$ of the state space model in Equation (25). The outputs of the ELM are the state variables $\boldsymbol{x}(k+1)$ and the output variables $\boldsymbol{y}(k)$ of the SSM. Then, the construction of the AFOS-ELM is shown in Figure 1.

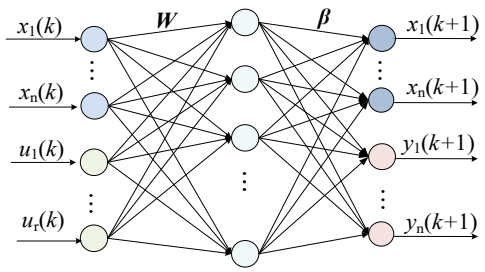


Figure 1. AFOS-ELM neural network construction.

The OS-ELM adopts linear output nodes, and the analytical expression of network output can be described as

$$\boldsymbol{y}_{ELM}(k) = \sum_{i=1}^{l} \boldsymbol{\beta}_i(k) f(\boldsymbol{W}_i \boldsymbol{x}_{ELM}(k) + b_i) \tag{29}$$

where $\boldsymbol{y}_{ELM}(k) = \left[x^{\mathrm{T}}(k+1) \quad y^{\mathrm{T}}(k)\right]$, $\boldsymbol{x}_{ELM}(k) = \left[x^{\mathrm{T}}(k) \quad u^{\mathrm{T}}(k)\right]$ are the neural network output vector and input vector.

The partial derivative of network output to network input is calculated by

$$\frac{\partial y_{ELM}(k)}{\partial x_{ELM}(k)} = \sum_{j=1}^{l} \frac{\partial y_{ELM}(k)}{\partial h_j(k)} \frac{\partial h_j(k)}{\partial x_{ELM}(k)} = \sum_{j=1}^{l} \beta_j(k) W_j \dot{f}(W_j x_{ELM}(k) + b_j) \tag{30}$$

$$\dot{f}(W_j x_{ELM}(k)+b_j)=\frac{e^{-W_j x_{ELM}(k)-b_j}}{(1+e^{-W_j x_{ELM}(k)-b_j})^2} \quad (31)$$

Therefore, the parameters in Equation (27) can be calculated according to Equations (30) and (31), and then the SSM in Equation (26) is obtained at time *k*.

5. Simulation and Discussion

5.1. Data Process

In order to verify the method of establishing SSM online based on the neural network proposed in this paper, the test data of a twin spool hybrid exhaust afterburner turbofan aero-engine are used for modeling and validation. The research is carried out in the environment of Win11, Intel(R) CORE(R) i5-12600Kf CPU, and 32G RAM.

Since the data come from the engine test, the measurement noise is inevitable. To avoid the impact of outliers on modeling accuracy, the local singularity estimation capability of wavelets and the residuals of the signal wavelet transform can be used to determine the residual threshold for outlier discrimination [35,36].

The residual of the original signal *S* after *L*-layer wavelet decomposition can be described by

$$R_e = S - S_S \quad (32)$$

where S_S are the low-frequency reconstructed signal after *L*-layer decomposition.

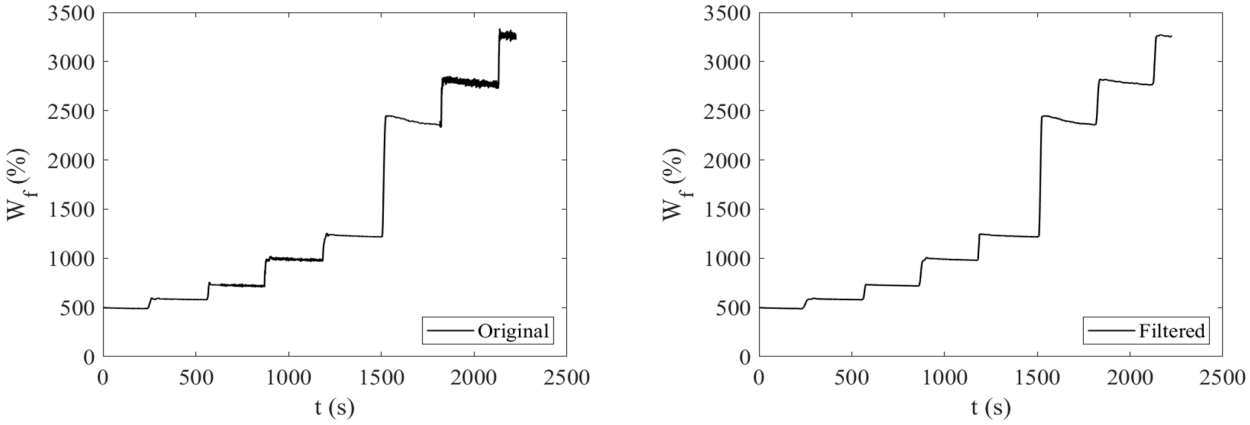
When the residual is small, the low-frequency reconstructed signal approaches the original signal. When the residual is large, there is a relatively large-amplitude high-frequency noise superimposed on the measured signal, which can be seen as an outlier. Taking the fuel data of an engine test as an example, the "db8" wavelet with scale 7 is used to detect the outliers. A total number of 111,406 data are collected in this test, and the residual distribution of fuel flow W_f is shown in Table 1.

Table 1. Statistics of wavelet transform residual distribution of W_f.

	Residual Distribution									
Residual interval/%	[−∞, −7.04]	[−7.04, −4.94]	[−4.94, −2.83]	[−2.83, −0.721]	[−0.721, 1.39]	[1.39, 3.49]	[3.49, 5.6]	[5.6, 7.71]	[7.71, 9.82]	[9.82, ∞]
Number of data	9	7	16	1348	109,786	207	17	8	2	6

It can be seen from Table 1 that the data in the range of residual [−0.721%, 1.39%] account for 98.54% of the total data, and the data in the range of residual [−2.83%, 3.49%] account for 99.94% of the total data. The data distribution in the range of residual greater than 3.49% or less than −2.83% does not tend to decrease with the increase of residual. Therefore, the threshold for outlier identification is set to −2.83% and 3.49%. The data with residuals beyond this range are considered outliers, which are replaced by the average value of the data before and after them. For continuous outliers, they are replaced by linear interpolation of the reasonable data before and after them. After replacing the outlier data, the "db8" wavelet with scale 7 is adopted to filter the signal. Taking fuel data as an example, the comparison of data before and after filtering is shown in Figure 2a,b.

It can be seen from Figure 2 that the noise in the filtered data is significantly suppressed, which is helpful to establish the SSM in a data-driven way. Then, the data are normalized with Equation (27).



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

Figure 2. Fuel data comparison before and after filtering. (**a**) Original fuel data; (**b**) filtered fuel data.

5.2. AFOS-ELM Validation

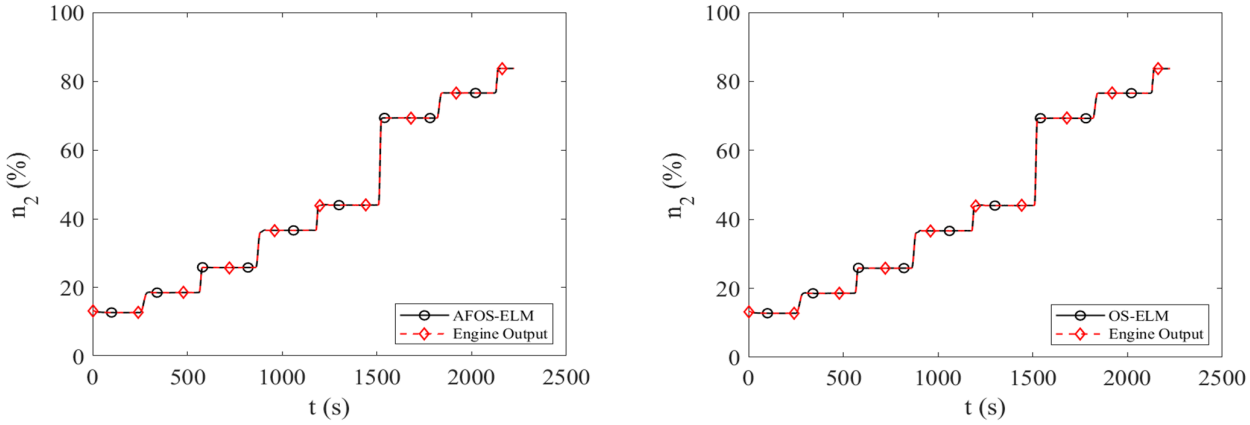
The input vector for a twin spool turbo-fan engine control system is $u(k) = [W_f(k), A_8(k)]^T$, and A_8 is the nozzle throat area. The measured output vector besides rotor speeds is $y(k) = [P_3(k), T_6(k), P_6(k), F(k)]^T$, where P_3 is the total pressure at compressor outlet, T_6, P_6 are the total temperature and total pressure at low-pressure turbine outlet, and F is the thrust. The state vector is $x(k) = [n_1(k), n_2(k)]^T$; n_1 and n_2 are the low-pressure rotor speed and the high-pressure rotor speed.

According to the construction of AFOS-ELM in Figure 1, the input vector of the ELM is $\boldsymbol{x}_{ELM}(k) = \begin{bmatrix} W_f(k) & A_8(k) & n_1(k) & n_2(k) \end{bmatrix}^T$. The output vector of the ELM is $\boldsymbol{y}_{ELM}(k) = [n_1(k+1),\ n_2(k+1),\ P_3(k),\ T_6(k),\ P_6(k),\ F(k)]^T$.

The first 30% of the engine test data sequence are adopted to initialize the ELM based on the inverse-free ELM offline. When the root mean square error is less than 6.5% or the number of hidden layer nodes increases to more than 100, the offline training stops. The number of hidden layer nodes is automatically set to 86 in this research.

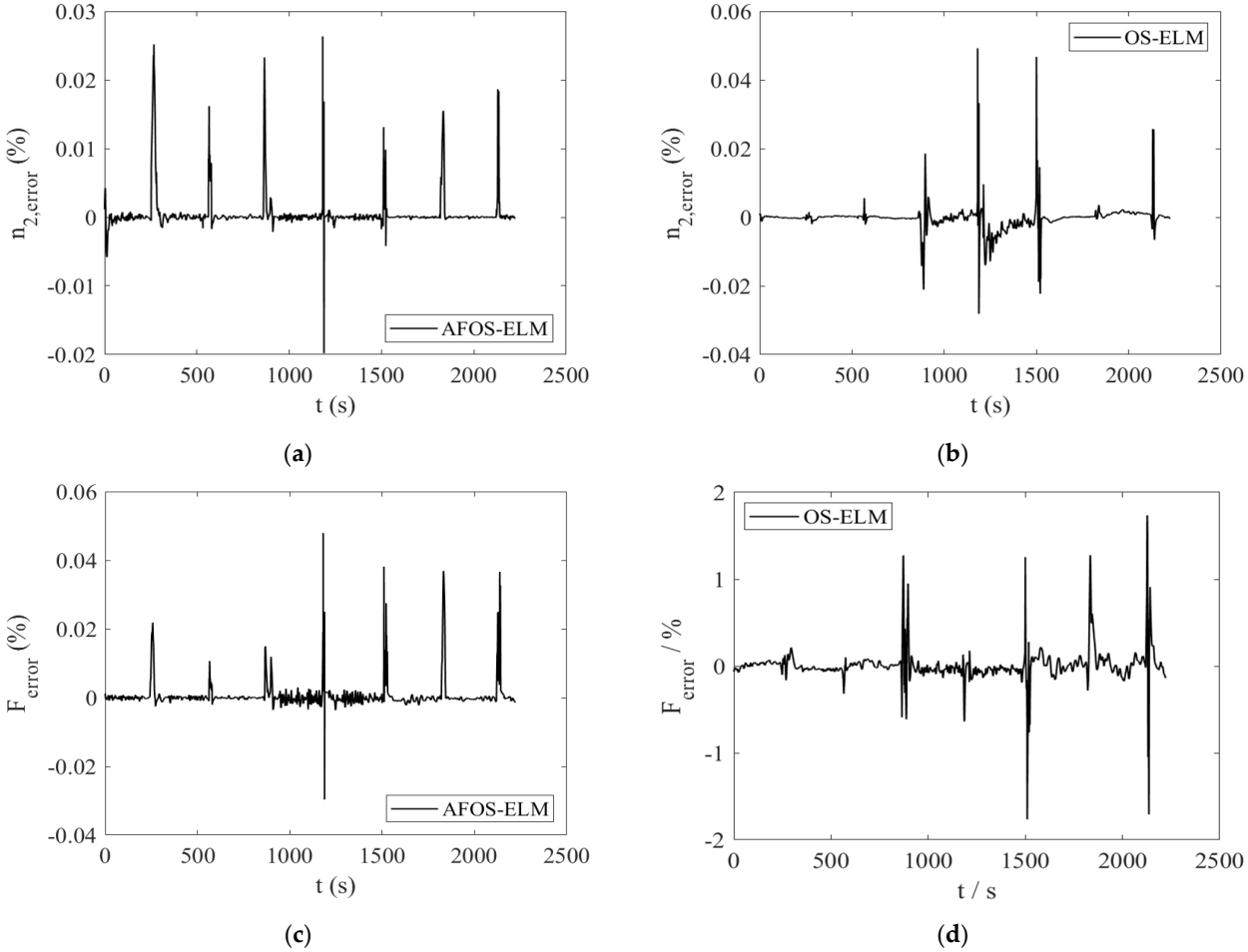
The AFOS-ELM is used to update the output weights online for all the engine test data. The n_2 output of the AFOS-ELM is compared with the engine test data in Figure 3. The relative output errors of n_2 and F are compared with that of the traditional OS-ELM without the forgetting factor in Figure 4. The detail errors of the ELM outputs are listed in Table 2.

From Figure 3 we can see that both the AFOS-ELM and OS-ELM output $n_2(k+1)$ track the engine test data well in the acceleration process. From Figure 4 we can see that the relative errors of $n_2(k+1)$ and $F(k)$ are both smaller than 0.05% with the AFOS-ELM method, which is much smaller than that of the OS-ELM method, and the adaptive forgetting factor enhances the modeling accuracy both at the dynamic state and at the steady state.



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

Figure 3. Comparison of neural network $n_2(k+1)$ output. (**a**) AFOS-ELM $n_2(k+1)$ output; (**b**) OS-ELM $n_2(k+1)$ output.



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

(**c**) (**d**)

Figure 4. Comparison of relative error between AFOS-ELM and OS-ELM. (**a**) Error of AFOS-ELM $n_2(k+1)$ output; (**b**) error of OS-ELM $n_2(k+1)$ output; (**c**) error of AFOS-ELM $F(k+1)$ output; (**d**) error of OS-ELM $F(k+1)$ output.

Table 2. Relative error of neural network output (%).

Method	Error Type	Variables					
		$n_1(k+1)$	$n_2(k+1)$	$P_3(k)$	$T_6(k)$	$P_6(k)$	$F(k)$
AFOS-ELM	Average error	0.0013	0.0008	0.0016	0.002	0.0014	0.0013
	Maximum error	0.0649	0.0263	0.2084	0.2747	0.0866	0.0479
	Standard deviation	0.0043	0.0026	0.0055	0.0051	0.004	0.0037
OS-ELM	Average error	0.0019	0.0014	0.3218	0.5854	0.2815	0.0831
	Maximum error	0.1419	0.0492	5.8795	16.52	6.8424	1.7649
	Standard deviation	0.0042	0.0028	0.4385	1.0432	0.5142	0.1572

It can be seen from Table 2 that the errors of two rotor speeds are very small for both AFOS-ELM and OS-ELM, the maximum error occurs at n_1 with OS-ELM method, which is 0.1419%, and the average error is less than 0.002%, which is much better than that of the component-level model [37]. For the other output variables, the AFOS-ELM method shows greater advantage than the OS-ELM method. The maximum error of AFOS-ELM occurs at T_6, which is 0.2747%, and the max average error occurs also occurs at T_6, which is 0.002%, while for OS-ELM, the maximum error and the max average error also occur at T_6, which are 16.52% and 0.5854%, respectively, and the figures are much bigger than that of the AFOS-ELM. AFOS-ELM can model the state space configuration nonlinear model well. It can process the engine failure data and ensure the accuracy of the current output. The adaptive forgetting factor can also deal with the rapid change caused by the large dynamic and maintain a high accuracy.

5.3. State Space Model Validation

Based on the analytical expression of the AFOS-ELM model established above, the SSM can be derived by partial derivative calculation. Taking the calculation of parameter $a_{11}(k)$ as an example, according to Equations (3) and (27), we have

$$a_{11}(k) = \frac{\partial n_1(k+1)}{\partial n_1(k)} = \frac{\partial n_1(k+1)}{\partial \boldsymbol{H}(k)} \frac{\partial \boldsymbol{H}(k)}{\partial n_1(k)} = \sum_{j=1}^{l} \beta_{1j} W_{1j} \dot{f}(W_{1j} n_1(k) + b_j) \tag{33}$$

Applying the calculation process of a_{11} o the other parameters of the SSM, the calculation of all state space matrix parameters at time k can be achieved, and the SSM at time k is established.

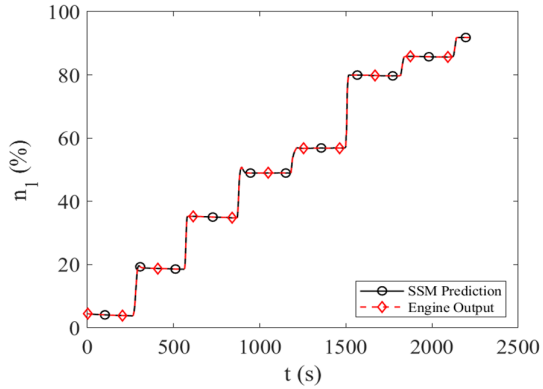
In order to validate the effectiveness of the SSM established in this paper, the models established at different times are selected to demonstrate their output prediction ability over future time, which is essential for model-predictive control and performance-seeking control of aero-engines.

The prediction output of SSM at future time can be obtained through the response calculation of a discrete-time system.

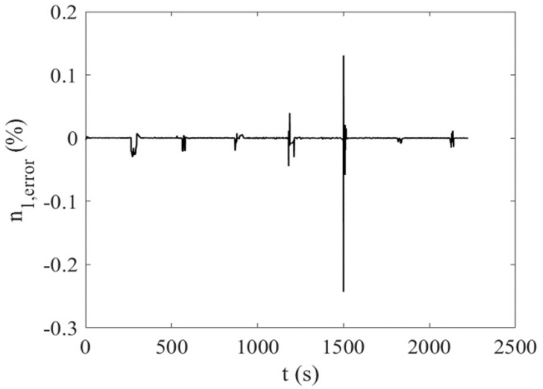
$$\begin{aligned} x(k+1) &= \boldsymbol{A}(k)x(k) + \boldsymbol{B}(k)u(k) \\ x(k+2) &= \boldsymbol{A}(k)x(k+1) + \boldsymbol{B}(k)u(k+1) \\ &= \boldsymbol{A}^2(k)x(k) + \boldsymbol{A}(k)\boldsymbol{B}(k)u(k) + \boldsymbol{B}(k)u(k+1) \\ &\cdots\cdots \\ x(k+p) &= \boldsymbol{A}^p(k)x(k) + \sum_{i=k}^{k+p-1} \boldsymbol{A}^{k+p-1-i}(k)\boldsymbol{B}(k)u(i) \\ y(k+p) &= \boldsymbol{C}(k)\boldsymbol{A}^p(k)x(k) + \sum_{i=k}^{k+p-1} \boldsymbol{C}(k)\boldsymbol{A}^{k+p-1-i}(k)\boldsymbol{B}(k)u(i) + \boldsymbol{D}(k)u(k+p) \end{aligned} \tag{34}$$

Taking p = 5 as an example, the error between the SSM prediction output and the engine output is compared. The results of prediction are shown in Figure 5a–l, where the

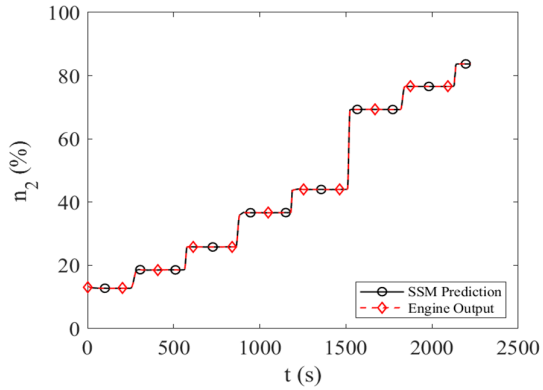
"SSM Prediction" represents the prediction output of the SSM proposed in this paper, and "Engine Output" represents the engine test data at $k + p$.



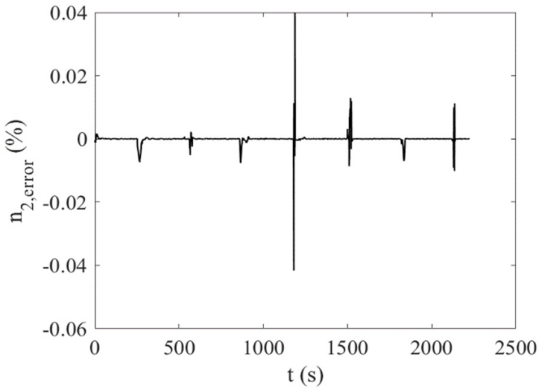
(**a**)



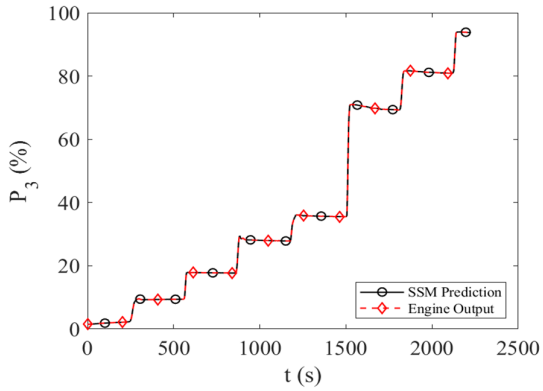
(**b**)



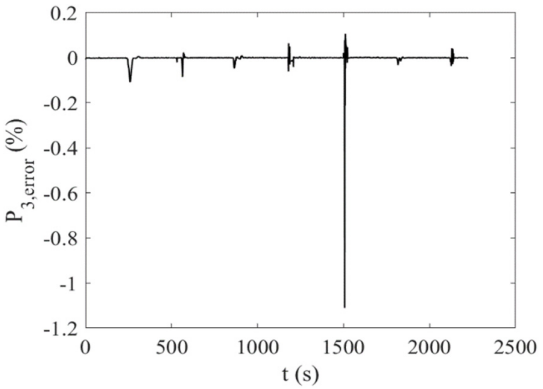
(**c**)



(**d**)



(**e**)



(**f**)

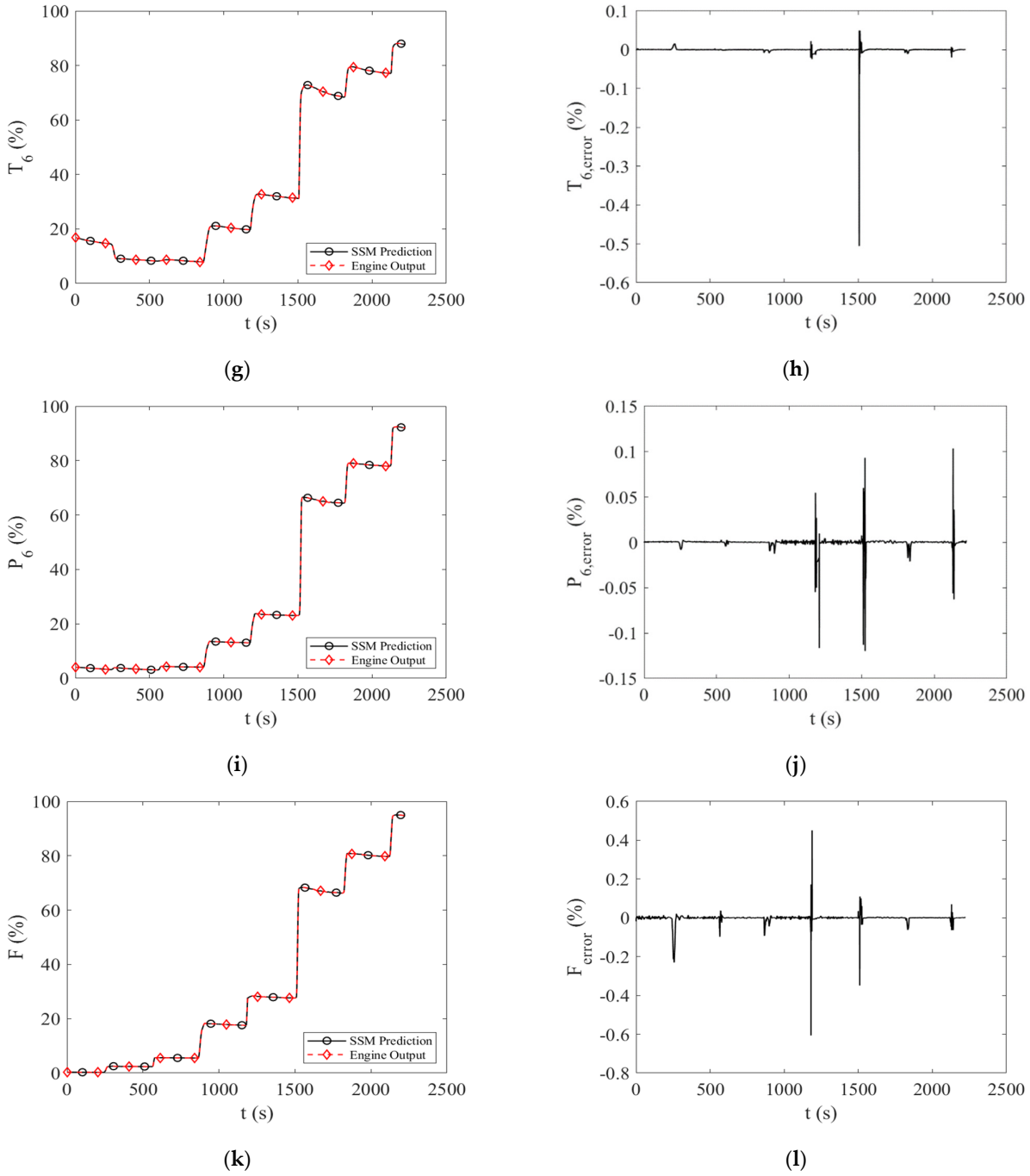


Figure 5. Prediction result of SSM at time k+5. (**a**) n_1 prediction; (**b**) prediction error of n_1; (**c**) n_2 prediction; (**d**) prediction error of n_2; (**e**) P_3 prediction; (**f**) prediction error of P_3; (**g**) T_6 prediction; (**h**) prediction error of T_6; (**i**) P_6 prediction; (**j**) prediction error of P_6; (**k**) F prediction; (**l**) prediction error of F.

As can be seen from Figure 5, the maximum prediction error at time k+5 occurs at P_3 around 1500 s, where a sudden acceleration occurs and the thrust F increases about 40.6%. Except for F, the other variables all reach their biggest prediction error at this moment. Under such a big dynamic, the max prediction error occurred at P_3 and is about 1.1%. The max prediction error of the low-pressure rotor speed is less than 0.3% when the low-

pressure rotor speed n_1 increases about 22.9%, while the error is more than 0.4% in the component-level-model-based online state space modeling method in reference [21] for time $k + 5$ output prediction when the low-pressure rotor speed decreases by about 12% (there is no other common variable that can be compared in reference [21]). The prediction accuracy of the SSM established by the method described in this paper shows great advantage, and it can predict the varying of the engine test data well.

In order to validate the real-time performance of the SSM proposed in this paper, the time consumption of the AFOS-ELM online modeling process and the SSM modeling process are evaluated. The simulation platform is as described above; the modeling process is carried out three times. The neural network is established online based on 111,406 groups of engine test data, and 111,405 SSMs are obtained from the first data to the last-1 data, and the average time for modeling the 111,405 SSMs based on AFOS-ELM is 183.958 s. The online network weights updating and online SSM calculation process of each step takes an average of 1.7 ms. It is lower than the sampling step of the aero-engine, which is about 20 ms. Therefore, this modeling method can meet the real-time requirements of the aero-engine.

6. Conclusions

An online data-driven state space modeling method based on AFOS-ELM is proposed in this paper, which provides a mechanism-model-free state space modeling method for a complex nonlinear system. The inputs and outputs of the AFOS-ELM are determined according to the description of the SSM. The partial derivative calculation based on the analytical model of the AFOS-ELM provides an alternative for deference-based calculation in the Jacobian matrix (state space matrix) calculation of SSM. The OS-ELM with adaptive forgetting factor shows great system dynamic capture ability and achieves high modeling accuracy at both steady state and dynamic state on the engine test dataset, which means the SSMs obtained from it have the ability to predict the system output in future time accurately.

The inverse free-ELM algorithm used in the OS-ELM initialization can automatically determine the most appropriate number of the hidden layer nodes, which reduces the trail time and enhances the modeling efficiency.

The SSM is established online at each sampling time of the engine operation and has high real-time property, which gives it the potential to be further used in model-based control and health management research.

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

Funding: This research was funded by Postgraduate Research & Practice Innovation Program of NUAA (xcxjh20210214), National Natural Science Foundation of China (51976089), Enhanced Foundation Project (2017-JCJQ-ZD-047-21), the National Science and Technology Major Project (2017-V-0004-0054).

Data Availability Statement: Not applicable.

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

References

1. Zhou, W. Research on Object-Oriented Modeling and Simulation for Aeroengine and Control System. Ph.D. Thesis, Nanjing University of Aeronautics and Astronautics, Nanjing, China, 2006.
2. Alag, G.; Gilyard, G. A proposed Kalman filter algorithm for estimation of unmeasured output variables for an F100 turbofan engine. In Proceedings of the 26th Joint Propulsion Conference, Orlando, FL, USA, 16–18 July 1990. https://doi.org/10.2514/6.1990-1920.
3. Pang, S.; Li, Q.; Ni, B. Improved nonlinear MPC for aircraft gas turbine engine based on semi-alternative optimization strategy. *Aerosp. Sci. Technol.* **2021**, *118*, 106983.

4. DeCastro, J.A. Rate-based model predictive control of turbofan engine clearance. *J. Propuls. Power* **2007**, *23*, 804–813.
5. Roberts, R.A.; Eastbourn, S.M. Modeling techniques for a computational efficient dynamic turbofan engine model. *Int. J. Aerosp. Eng.* **2014**, *2014*, 283479. https://doi.org/10.1155/2014/283479.
6. Feng, Z.; Sun, J.; Huang, J.; Cai, H. A New Method for Establishing a State Variable Model of Aeroengine. *J. Aerosp. Power* **1998**, *13*, 435–438, 463.
7. Sun, J. Prospects of the Aeroengine Control Development in the Early Time of the 21st Century. *J. Aerosp. Power* **2001**, *16*, 97–102.
8. Sugiyama, N. Derivation of ABCD system matrices from nonlinear dynamic simulation of jet engines. In Proceedings of the 28th Joint Propulsion Conference and Exhibit, Nashville, TN, USA, 6–8 July 1992. https://doi.org/10.2514/6.1992-3319.
9. Feng, Z.; Sun, J. Modeling of small perturbation state variable model for aeroengines. *J. Propuls. Technol.* **2001**, *22*, 54–57.
10. Li, Q.; Sun, J. Aero-Engine State Variable Modeling Based on the Genetic Algorithm. *J. Aerosp. Power* **2006**, *21*, 427–431.
11. Sugiyama, N. Derivation of system matrices from nonlinear dynamic simulation of jet engines. *J. Guid. Control Dyn.* **1994**, *17*, 1320–1326. https://doi.org/10.2514/3.21350.
12. Henrion, D.; Reberga, L.; Bernussou, J.; Vary, F. Linearization and identification of aircraft turbofan engine models. *IFAC Proc. Vol.* **2004**, *37*, 1055–1060.
13. Lytle, J.K. The Numerical Propulsion System Simulation: An Overview. In Proceedings of the Computational Aerosciences Workshop, Moffett Field, CA, USA, 15–17 February 2000; p. E-12152.
14. Parker, K.I.; Guo, T. *Development of a Turbofan Engine Simulation in a Graphical Simulation Environment*; NASA/TM-2003-212543; NASA: Washington, DC, USA, 2003.
15. Ma, J.; Chang, J.; Ma, J.; Bao, W.; Yu, D. Mathematical modeling and characteristic analysis for over-under turbine based combined cycle engine. *Acta Astronaut* **2018**, *148*, 141–152.
16. Diao, Y.; Passino, K.M. Stable fault-tolerant adaptive fuzzy/neural control for a turbine engine. *IEEE Trans. Control Syst. Technol.* **2001**, *9*, 494–509.
17. Pan, M.; Wang, H.; Huang, J. T–S Fuzzy Modeling for Aircraft Engines: The Clustering and Identification Approach. *Energies* **2019**, *12*, 3284.
18. Liu, X.; Yuan, Y.; Shi, J.; Zhao, L. Adaptive modeling of aircraft engine performance degradation model based on the equilibrium manifold and expansion form. *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.* **2014**, *228*, 1246–1272.
19. Liu, X.; Zhang, L.; Luo, C. Model reference adaptive control for aero-engine based on system equilibrium manifold expansion model. *Int. J. Control* **2021**. https://doi.org/10.1080/00207179.2021.2016979.
20. Pang, S.; Li, Q.; Zhang, H. An Exact Derivative Based Aero-Engine Modeling Method. *IEEE Access* **2018**, *6*, 34516–34526.
21. Pang, S.; Li, Q.; Zhang, H. A new online modelling method for aircraft engine state space model. *Chin. J Aeronaut* **2020**, *33*, 1756–1773.
22. Wang, C.; Zhu, Z.; Lu, N.; Cheng, Y.; Jiang, B. A data-driven degradation prognostic strategy for aero-engine under various operational conditions. *Neurocomputing* **2021**, *462*, 195–207.
23. Gu, Z.; Pang, S.; Zhou, W.; Li, Y.; Li, Q. An Online Data-Driven LPV Modeling Method for Turbo-Shaft Engines. *Energies* **2022**, *15*, 1255.
24. Zheng, Q.; Zhang, H.; Li, Y.; Hu, Z. Aero-Engine On-Board Dynamic Adaptive MGD Neural Network Model within a Large Flight Envelope. *IEEE Access* **2018**, *6*, 45755–45761.
25. Zheng, Q.; Fang, J.; Hu, Z.; Zhang, H. Aero-engine on-board model based on batch normalize deep neural network. *IEEE Access* **2019**, *7*, 54855–54862.
26. Huang, G.; Zhu, Q.; Siew, C. Extreme learning machine: Theory and applications. *Neurocomputing* **2006**, *70*, 489–501.
27. Huang, G.; Chen, L.; Siew, C.K. Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Trans. Neural Netw.* **2006**, *17*, 879–892.
28. Li, S.; You, Z.; Guo, H.; Luo, X.; Zhao, Z. Inverse-free extreme learning machine with optimal information updating. *IEEE Trans. Cybern.* **2015**, *46*, 1229–1241.
29. Jin, B.; Jing, Z.; Zhao, H. Incremental and decremental extreme learning machine based on generalized inverse. *IEEE Access* **2017**, *5*, 20852–20865.
30. Zhu, H.; Wu, Y. Inverse-Free Incremental Learning Algorithms with Reduced Complexity for Regularized Extreme Learning Machine. *IEEE Access* **2020**, *8*, 177318–177328.
31. Liang, N.; Huang, G.; Saratchandran, P.; Sundararajan, N. A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Trans. Neural Netw.* **2006**, *17*, 1411–1423.
32. Du, Z.; Li, X.; Zheng, Z.; Zhang, G.; Mao, Q. Extreme learning machine based on regularization and forgetting factor and its application in fault prediction. *Chin. J. Sci. Instrum.* **2015**, *36*, 1546–1553.
33. Guo, W. Robust adaptive online sequential extreme learning machine for predicting nonstationary data streams with outliers. *J. Algorithms Comput. Technol.* **2019**, *13*, 174830261989542.
34. Guo, W.; Xu, T.; Yu, J.; Tang, K. Online Sequential Extreme Learning Machine Based on M-estimator and Variable Forgetting Factor. *J. Electron. Inf. Technol.* **2018**, *40*, 1360–1367. https://doi.org/10.11999/JEIT170800.
35. Torrence, C.; Compo, G.P. A practical guide to wavelet analysis. *Bull. Am. Meteorol. Soc.* **1998**, *79*, 61–78.
36. Bilen, C.; Huzurbazar, S. Wavelet-based detection of outliers in time series. *J. Comput. Graph. Stat.* **2002**, *11*, 311–327.
37. Pang, S.; Li, Q.; Feng, H.; Zhang, H. Joint steady state and transient performance adaptation for aero engine mathematical model. *IEEE Access* **2019**, *7*, 36772–36787.