Neural-network-based observer for turbine engine parameter estimation

P Shankar[1]* and **R K Yedavalli**[2]
[1]Department of Mechanical and Aerospace Engineering, Arizona State University, Tempe, Arizona, USA
[2]Department of Aerospace Engineering, The Ohio State University, Columbus, Ohio, USA

The manuscript was received on 31 March 2009 and was accepted after revision for publication on 4 June 2009.

DOI: 10.1243/09596518JSCE782

Abstract: Accurate estimation of unmeasurable engine parameters such as thrust and turbine inlet temperatures constitutes a significant challenge for the aircraft community. A solution to this problem is to estimate these parameters from the measured outputs using an observer. Currently existing technologies rely on Kalman and extended Kalman filters to achieve this estimation. This paper presents a neural-network-based observer that augments the linear Kalman filter with a neural network to compensate for any non-linearity that is not handled by the linear filter. The implemented neural network is a radial basis function network that is trained offline using a growing and pruning algorithm. The neural-network-based observer is trained and simulated to estimate the high-pressure turbine inlet temperature, thrust, and stall margins at different levels of engine degradation for a two-spool turbofan engine. Simulation results show the ability of the observer to accurately estimate the performance parameters. The advantage of this observer is that it does not need explicit estimation of health parameters to accurately estimate the performance parameters.

Keywords: neural network, observer, performance parameters, aircraft engine, radial basis function network, growing and pruning

1 INTRODUCTION

A knowledge of performance parameters is an important requirement for effective operation of an aircraft engine. Accurate information of these parameters is required for controlling the engine, fault detection, and engine health management. Engine performance parameters such as engine thrust, compressor stall margins, and high-pressure turbine (HPT) inlet temperature cannot be measured explicitly. Since sensor information is not available, they need to be estimated from measured (sensed) quantities. Spang and Brown [**1**] described various jet engine controllers and highlighted the need for observers to accurately estimate the unmeasured parameters. Patton and Chen [**2**] emphasized the use of observers for fault detection in aircraft engines which is especially important for flight safety. Litt *et al.* [**3**] provided a detailed description of aircraft engine control and health management technologies and highlighted the role of parameter estimation for efficient operation of the engine. Estimation of aircraft engine performance parameters for different levels of degradation has been a significant challenge and in many cases highly uncertain resulting in very conservative safety margins [**4**]. Thus, accurate estimation of these parameters is necessary for enhancing efficiency while increasing flight safety.

Current approaches for performance parameter estimation are based on linear Kalman filters. Kobayashi *et al.* [**5**] investigated an approach based on a constant gain extended Kalman filter and on-board engine models for in-flight estimation of parameters such as thrust and stall margins. This estimator requires the engine model to run simultaneously as a part of the observer. Simon and Simon [**6**] developed an analytic method to incorporate state variable inequality constraints in the Kalman filter and applied it to a turbofan engine. The

**Corresponding author: Mechanical and Aerospace Engineering, Arizona State University, Engineering Center G-Wing Room 346, PO Box 876106, Tempe, Arizona AZ85287, USA.*
email: praveen.shankar@asu.edu

resultant filter is a combination of a standard Kalman filter and a quadratic programming problem. Dewallef *et al.* [**7**] presented the development of an unscented Kalman filter for engine diagnostics. Litt *et al.* [**8**] discussed a linear point extended Kalman-filter-based design technique to enable the optimal estimation of thrust in an aircraft engine.

Generally, a mathematical model of a jet engine is either unavailable or too complex to apply standard estimation methods. Physics-based models such as in [**9**] are very limited in their ability to represent the complexity of the engine. However, computer simulations that approximate the physical engine data with high accuracy are available. The fundamental dynamic model of an aircraft engine is highly non-linear. In order to control the engine, real-time estimates of performance parameters are necessary and hence an accurate model of the engine would be very useful. Neural networks have the ability to approximate the non-linear functions when trained with appropriate inputs. In this paper the ability of a radial basis function network to approximate the non-linearity in engine dynamics is explored using engine simulation data. This neural network is allied to a linear Kalman filter to observe augmented engine performance parameters.

Observers based on neural networks have been used for many applications including aircraft engines. Maggiore *et al.* [**10**] discussed a neural-network-based method for estimation of unmeasurable states in an engine simulation model. The paper focused on the importance of input selection for the estimator and its influence on the accuracy of estimation. Volponi [**11**] described the use of hybrid engine modelling using multi-layer perceptrons for performance tracking in turbine engines. Observers using neural networks have been reported for various other applications. Vargas and Hemerly [**12**] described an observer based on linearly parameterized neural networks for estimation of unknown general non-linear systems. Lainiotis and Plataniotis [**13**] discussed an estimator based on dynamic neural networks. A radial basis function network (RBFN) has been used for state estimation in electric power networks [**14**]. Chen *et al.* [**15**] discussed the application of neural networks for state estimation in active vibration control. Meghdari *et al.* [**16**] utilized a neural-network-based observer to estimate the tip over margin for mobile manipulators. They showed that the observer is fast enough for real-time tip over control. Abdollahi *et al.* [**17**] presented the development of a stable neural-network-based observer for general multivariable non-linear systems. The neural network is non-linear in the parameters and hence is able to fit systems with higher degrees of non-linearity. This observer was successfully applied to a flexible-joint manipulator. In this paper, a neural-network-based observer is prepared for the estimation of the states of an aircraft engine using growing and pruning RBFNs. This technique is based on a philosophy similar to [**11**] where a hybrid model of an engine was implemented using a physics-based model augmented with an empirical model consisting of a pretrained neural network. In this work, advantage is taken of the function approximation properties of a RBFN as compared to the multi-layer perceptron used in [**11**]. The RBFN is trained offline using a growing and pruning learning algorithm where hidden-layer neurons are added or deleted based on the inputs arriving at the network. This method was first presented by Shankar and Yedavalli [**18**]. The RBFN is trained using the simulation data from a turbine engine simulation model. The advantage of this approach is that the observer does not require an on-board engine model that runs simultaneously thus decreasing the computational burden. Also, since the network can be trained using training data that consists of various degraded conditions of the engine, the health parameters need not be estimated to get an accurate estimate of the performance parameters at different engine conditions.

This paper is organized as follows: the neural-network-based observer is presented followed by a description of RBFNs and the learning algorithm that has been implemented. The application of this observer to estimation in turbine engines is then explained followed by a discussion of the simulation results and conclusions.

2 NEURAL-NETWORK-BASED STATE OBSERVER

Consider a non-linear system of the form

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{1}$$

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u}) \simeq \mathbf{C}\boldsymbol{x} + \mathbf{D}\boldsymbol{u} \tag{2}$$

Let the linearized model of the above non-linear system be given by

$$\dot{\boldsymbol{x}} = \mathbf{A}\boldsymbol{x} + \mathbf{B}\boldsymbol{u} \tag{3}$$

$$\boldsymbol{y} = \mathbf{C}\boldsymbol{x} + \mathbf{D}\boldsymbol{u} \tag{4}$$

such that the pairs (**A**, **B**) and (**A**, **C**) are controllable and observable respectively. Since (**A**, **C**) is observable, an observer can be designed with a gain **L** [**19**] that is given by

$$\dot{\hat{\boldsymbol{x}}} = \mathbf{A}\hat{\boldsymbol{x}} + \mathbf{B}\boldsymbol{u} + \mathbf{L}(\boldsymbol{y} - \hat{\boldsymbol{y}}) \tag{5}$$

$$\hat{\boldsymbol{y}} = \mathbf{C}\hat{\boldsymbol{x}} + \mathbf{D}\boldsymbol{u} \tag{6}$$

When the loop is closed around the linear system using the observer it gives the error dynamics that is given by

$$\dot{\boldsymbol{e}} = (\mathbf{A} - \mathbf{LC})\boldsymbol{e} \tag{7}$$

where

$$\boldsymbol{e} = \boldsymbol{x} - \hat{\boldsymbol{x}} \tag{8}$$

By designing a gain **L** such that (**A** − **LC**) is stable, it can be ensured that the error between the actual state and estimated state goes to zero asymptotically. However, when the loop is closed around the non-linear system using the linear observer it is possible to obtain

$$\dot{\boldsymbol{x}} - \dot{\hat{\boldsymbol{x}}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \mathbf{A}\hat{\boldsymbol{x}} - \mathbf{B}\boldsymbol{u} - \mathbf{L}(\boldsymbol{y} - \hat{\boldsymbol{y}}) \tag{9}$$

From the above equation, it is evident that the error goes to zero only when the linear model is a close approximation of the non-linear system. This is possible only at the equilibrium points of the non-linear system around which it has been linearized. The non-linear system and the closed loop can be rewritten as

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) + \mathbf{A}\boldsymbol{x} - \mathbf{A}\boldsymbol{x} + \mathbf{B}\boldsymbol{u} - \mathbf{B}\boldsymbol{u} \tag{10}$$

$$\dot{\boldsymbol{e}} = (\mathbf{A} - \mathbf{LC})\boldsymbol{e} + \Delta(\boldsymbol{x}, \boldsymbol{u}) \tag{11}$$

where

$$\Delta(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \mathbf{A}\boldsymbol{x} - \mathbf{B}\boldsymbol{u} \tag{12}$$

is the linearization error. Let $\boldsymbol{\psi}(\boldsymbol{y}, \boldsymbol{u})$ be a neural network parameterization to approximate the linearization error. Let the equation of the observer be

$$\dot{\hat{\boldsymbol{x}}} = \mathbf{A}\hat{\boldsymbol{x}} + \mathbf{B}\boldsymbol{u} + \mathbf{L}(\boldsymbol{y} - \hat{\boldsymbol{y}}) + \boldsymbol{\psi}(\boldsymbol{y}, \boldsymbol{u}) \tag{13}$$

The closed loop is then given by

$$\dot{\boldsymbol{e}} = (\mathbf{A} - \mathbf{LC})\boldsymbol{e} + \Delta(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{\psi}(\boldsymbol{y}, \boldsymbol{u}) \tag{14}$$

From the above equation, it is apparent that if the neural network component can be implemented to approximate the linearization error, it can be ensured that the error between the actual and estimated states goes to zero asymptotically. The schematic of the observer is shown in Fig. 1.

The neural network component of the observer is composed of a RBFN. The RBFN is trained offline to approximate the linearization error. The next section addresses the structure and learning algorithm of the neural network.

3 RBFNS

Radial basis functions (RBFs) are used for interpolation in multidimensional space. A RBFN is a neural network approached by viewing the design as a curve-fitting (approximation) problem in a high-dimensional space [**20**]. The basic structure of a RBFN is shown in Fig. 2.

The RBFN consists of a hidden layer and an output layer. The neurons in the hidden layer provide a set of non-linear functions that provide a basis for the input vectors when they are expanded into the hidden unit space. The most commonly used function is the Gaussian. The output of the RBFN is a linear combination of the outputs from the hidden units.

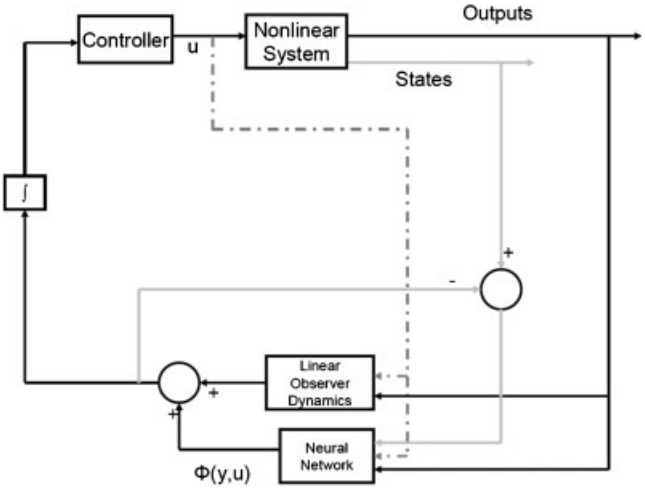

Fig. 1 Neural-network-based observer

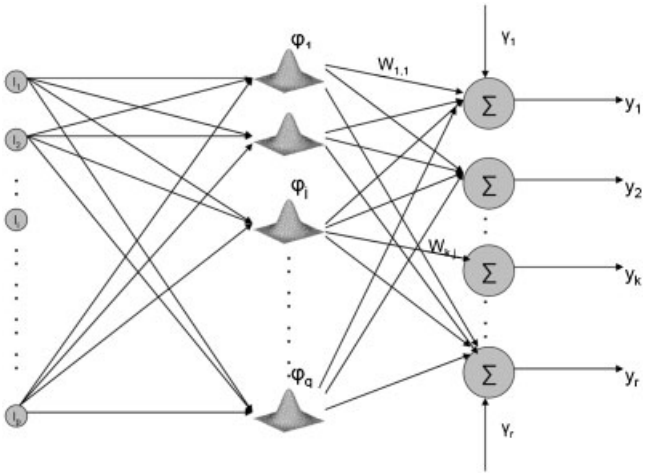

Fig. 2 The RBFN

3.1 Weight update

In most applications, the number of hidden neurons is chosen and the centres and variances of the neurons are selected such that they represent the majority of the input data. The weights of the output layer are updated as follows

$$w_{k,j}(n+1) = w_{k,j}(n) + \eta\left(y_k^{\mathrm{t}} - y_k\right)\phi_j \tag{15}$$

$$\gamma_k(n+1) = \gamma_k(n) + \eta\left(y_k^{\mathrm{t}} - y_k\right) \tag{16}$$

where $w_{k,j}$ represents the weight from the jth neuron of the hidden layer to the kth neuron of the output layer, γ_k is the weight for the bias term (+1) to the kth output neuron, y^{t} is the expected or training output, and η is the learning rate. Also, ϕ_j is the basis function and is given by

$$\phi_j = \exp\left[\sum_{i=1}^{p} -\left(c_{j,i} - I_i\right)^2 \Big/ \sigma_j^2\right] \tag{17}$$

where $c_{j,i}$ represents the centre of the jth neuron corresponding to the ith input I_i and σ_j is its variance. The neural network is trained to minimize the error between the target (desired) output y_k^{t} and the neural network output y_k.

3.2 Growing and pruning algorithm

The algorithm incorporates a simple strategy to add and delete neurons in the hidden layer based on the input that is received by the network in addition to updating the weights of the output layer. This means that the algorithm must implement a means of allocating and deleting hidden-layer neurons while updating the centres of the remaining hidden-layer neurons. The update of the centres of the remaining neurons is accomplished by the method of back-propagation and it is given by

$$c_{j,i}(n+1) = c_{j,i}(n) + 2\frac{\eta}{\sigma_j}\left(I_i - c_{j,i}(n)\right)\phi_j\left[\sum_{k=1}^{r}\left(y_k^{\mathrm{t}} - y_k\right)w_{k,j}\right] \tag{18}$$

The growing and pruning algorithm is illustrated for different cases in Fig. 3. The x-axis of the figure is a representation of the distance of the neuron centres from the input.

3.3 Growing strategy

If the input, say I appearing at the neural network is such that it is at a distance more than a prespecified absolute distance d_{add} from each of the hidden layer neuron centres, a new neuron with that input as the centre is added. The centres, variances, and initial weights of the new neuron is given by

$$c_{\mathrm{new},i} = I_{\mathrm{new}} \tag{19}$$

$$\sigma_{\mathrm{new}} = K_{\mathrm{add}} d_{\mathrm{min}} \tag{20}$$

$$w_{k,\mathrm{new}} = y_k^{\mathrm{t}} - y_k \tag{21}$$

$$\gamma_{k,\mathrm{new}} = 0 \tag{22}$$

where K_{add} is a predefined constant and d_{min} is the minimum distance between the centres of the existing hidden neurons, and the input I_{new}. The first neuron in the hidden layer is added with its centre at the first input coming into the network.

3.4 Pruning strategy

If inputs arriving at the network are such that they are close to the centres of some of the neurons while being far away from the other neurons, the latter set of neurons are deleted. The implementation is accomplished by checking the absolute distances of the input from the neuron centres with respect to a threshold beyond which any neuron receiving this input will not produce any output. The threshold for

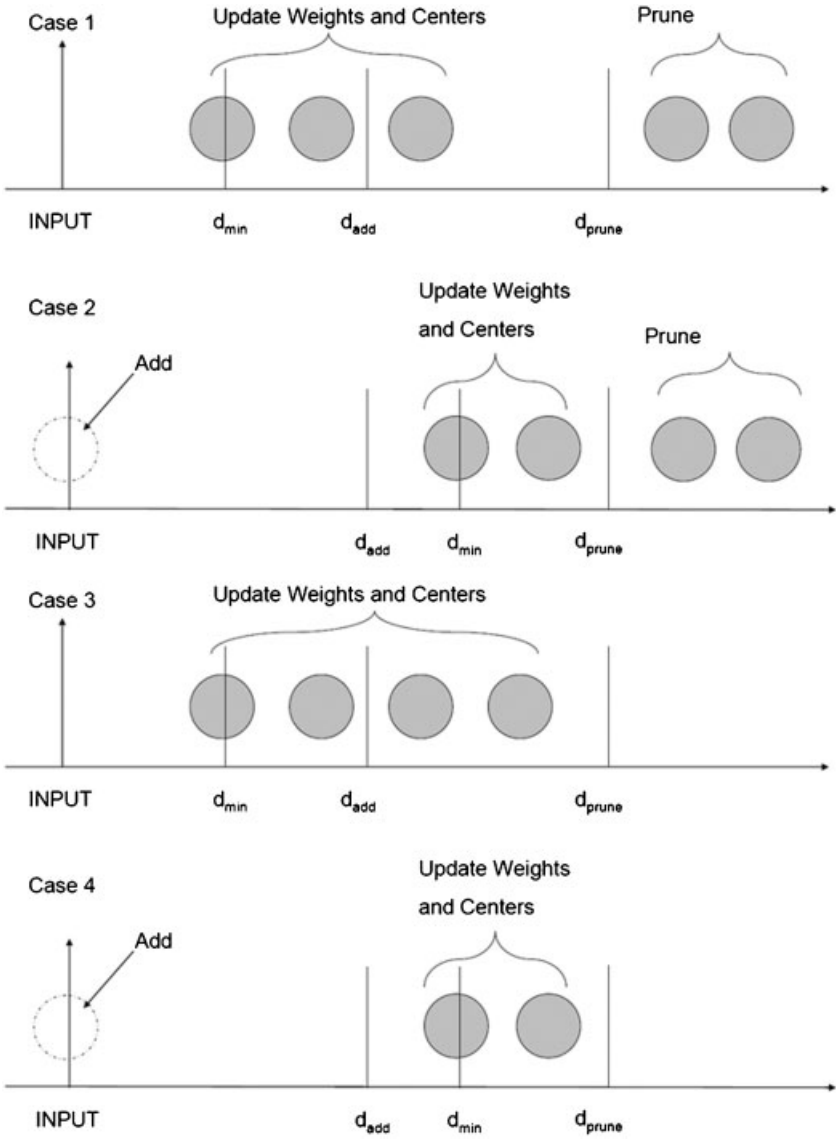

Fig. 3 Growing and pruning algorithm

pruning a neuron is selected to be $d_{\text{prune}} = M_{\text{prune}} \times d_{\min}$ where $M_{\text{prune}} > 1$ and $d_{\min}$ is the minimum distance of the neurons from the input. While a neuron is added every time an input that satisfies case 2 in Fig. 3 arrives at the network, pruning is accomplished only when it is ensured that a particular neuron has not produced an output for a majority of inputs in the training set. Therefore, pruning is executed at the end of every training cycle.

Note that, during the growing and pruning strategy, the weights of the output layer and the centres of the remaining hidden-layer neurons are continuously updated. The advantage of this algorithm is that it is relatively simple to implement and is not computationally intensive.

4 APPLICATION TO AIRCRAFT ENGINE

This section describes the application of the neural-network-based observer to an aircraft engine. A brief introduction to the engine simulation model is followed by a detailed description of the various components of the neural-network-based observer. Lastly, the training strategy for the growing and pruning RBFN is explained.

4.1 Turbine engine simulation model

The engine model used in this paper is a non-linear simulation of an advanced high-bypass turbofan engine. This engine model has been constructed as a component level model (CLM), which consists of the

major components of an aircraft engine and is similar to the engine model used in [**8**]. The CLM represents highly complex engine dynamics and simulates real-time data. It is also possible to simulate the engine under degraded conditions. In this paper the engine is simulated from idle to take-off.

The non-linear dynamics of the engine is given by

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{23}$$

$$\boldsymbol{y} = \mathbf{C}\boldsymbol{x} + \mathbf{D}\boldsymbol{u} \tag{24}$$

where $\boldsymbol{x} \in \Re^6$, $\boldsymbol{u} \in \Re^3$, $\boldsymbol{y} \in \Re^{12}$. Here it is assumed that the outputs of the engine are linear combinations of the states. This is a deliberate approximation to show the capability of the observer to approximate the non-linearity in the differential equation and estimate the states of the non-linear system. If a non-linear transformation from the states to the outputs were to be considered, the estimated output would not match the actual output even though the error between the estimated and actual states is zero. While not being addressed in this paper, it is possible to implement a second neural network to approximate the non-linearity in the output equation.

The states, inputs, and outputs of the model are shown in Table 1 and are defined in the Appendix.

The state space matrices needed to design the observer are derived by linearizing the engine simulation model at the required operating point. The linear state space model is given by

$$\dot{\boldsymbol{x}} = \mathbf{A}\boldsymbol{x} + \mathbf{B}\boldsymbol{u} \tag{25}$$

$$\boldsymbol{y} = \mathbf{C}\boldsymbol{x} + \mathbf{D}\boldsymbol{u} \tag{26}$$

The output $\boldsymbol{y}$ includes the measured ($\boldsymbol{y}_\mathbf{m}$) and unmeasured ($\boldsymbol{y}_\mathbf{c}$) outputs where

$$\boldsymbol{y}_\mathrm{m} = \mathbf{C}_\mathrm{m}\boldsymbol{x} + \mathbf{D}_\mathrm{m}\boldsymbol{u}, \quad \boldsymbol{y}_\mathrm{m} \in \Re^7 \tag{27}$$

$$\boldsymbol{y}_\mathrm{c} = \mathbf{C}_\mathrm{c}\boldsymbol{x} + \mathbf{D}_\mathrm{c}\boldsymbol{u}, \quad \boldsymbol{y}_\mathrm{c} \in \Re^5 \tag{28}$$

4.2 Implementation of neural-network-based observer

The non-linear dynamics of the aircraft engine can be rewritten as

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) + \mathbf{A}\boldsymbol{x} + \mathbf{B}\boldsymbol{u} - \mathbf{A}\boldsymbol{x} - \mathbf{B}\boldsymbol{u}$$

4.2.1 Linear component

The linear component of the neural-network-based observer is designed using a Kalman filter. Since the linearized model is observable from the seven measured outputs that constitute $\boldsymbol{y}_\mathrm{m}$, it is possible to implement a gain $\mathbf{L}$ such that when the loop is closed around the linearized model using the Kalman filter, the error between the actual and estimated states tends to zero asymptotically. The sensor standard deviation (to determine noise covariance matrices) required to design the Kalman filter are given in Table 2, where σ is the per cent of the steady-state values at ground maximum power condition. The linear observer dynamics is given by

$$\dot{\hat{\boldsymbol{x}}} = \mathbf{A}\hat{\boldsymbol{x}} + \mathbf{B}\boldsymbol{u} + \mathbf{L}(\boldsymbol{y}_\mathrm{m} - \hat{\boldsymbol{y}}_\mathrm{m}) \tag{29}$$

$$\hat{\boldsymbol{y}}_\mathrm{m} = \mathbf{C}_\mathrm{m}\hat{\boldsymbol{x}} + \mathbf{D}_\mathrm{m}\boldsymbol{u} \tag{30}$$

When the linear observer is used to close the loop around the non-linear system

Table 1 Engine model variables

Type	Variable
States (x)	XN2, XN25, TMHS23, TMHS3, TMHS41, TMHS49
Inputs (u)	WF36, AE24, STP25
Outputs (y)	T25, P25, T3, P3, T41, T49, FN, SMW12, SMW2, SMW25, XN2, XN25
Sensed or measured outputs (y_m)	T25, P25, T3, P3, T49, XN2, XN25
Unmeasured outputs (y_c)	FN, SMW12, SMW2, SMW25, T41

Table 2 Sensor standard deviation

	Sensor						
	XN2	XN25	P25	T25	P3	T3	T49
Standard deviation (σ)	0.25	0.25	0.5	0.75	0.5	0.75	0.75

$$\dot{\boldsymbol{e}} = (\mathbf{A} - \mathbf{L}\mathbf{C}_{\mathrm{m}})\boldsymbol{e} + \begin{pmatrix} \Delta_1(\boldsymbol{x}, \boldsymbol{u}) \\ \Delta_2(\boldsymbol{x}, \boldsymbol{u}) \\ \Delta_3(\boldsymbol{x}, \boldsymbol{u}) \\ \Delta_4(\boldsymbol{x}, \boldsymbol{u}) \\ \Delta_5(\boldsymbol{x}, \boldsymbol{u}) \\ \Delta_6(\boldsymbol{x}, \boldsymbol{u}) \end{pmatrix} \tag{31}$$

$$\boldsymbol{e} = \boldsymbol{x} - \hat{\boldsymbol{x}} \tag{32}$$

where $\Delta_i(\boldsymbol{x}, \boldsymbol{u})$ is the linearization error in each of the six non-linear equations and $\mathbf{C}_{\mathrm{m}}$ and $\mathbf{D}_{\mathrm{m}}$ corresponds to the output matrices for the measured outputs.

4.2.2 *Neural network component*

The neural network component consists of a RBFN the outputs of which approximate the linearization error in each of the six channels. The neural network observer is given by

$$\dot{\hat{\boldsymbol{x}}} = \mathbf{A}\hat{\boldsymbol{x}} + \mathbf{B}\boldsymbol{u} + \mathbf{L}(\boldsymbol{y}_{\mathrm{m}} - \widehat{\boldsymbol{y}_{\mathrm{m}}}) + \begin{pmatrix} \psi_1(\boldsymbol{y}_{\mathrm{m}}, \boldsymbol{u}) \\ \psi_2(\boldsymbol{y}_{\mathrm{m}}, \boldsymbol{u}) \\ \psi_3(\boldsymbol{y}_{\mathrm{m}}, \boldsymbol{u}) \\ \psi_4(\boldsymbol{y}_{\mathrm{m}}, \boldsymbol{u}) \\ \psi_5(\boldsymbol{y}_{\mathrm{m}}, \boldsymbol{u}) \\ \psi_6(\boldsymbol{y}_{\mathrm{m}}, \boldsymbol{u}) \end{pmatrix} \tag{33}$$

The closed-loop system is now given by

$$\dot{\boldsymbol{e}} = (\mathbf{A} - \mathbf{L}\mathbf{C}_{\mathrm{m}})\boldsymbol{e} + \begin{pmatrix} \Delta_1(\boldsymbol{x}, \boldsymbol{u}) \\ \Delta_2(\boldsymbol{x}, \boldsymbol{u}) \\ \Delta_3(\boldsymbol{x}, \boldsymbol{u}) \\ \Delta_4(\boldsymbol{x}, \boldsymbol{u}) \\ \Delta_5(\boldsymbol{x}, \boldsymbol{u}) \\ \Delta_6(\boldsymbol{x}, \boldsymbol{u}) \end{pmatrix} - \begin{pmatrix} \psi_1(\boldsymbol{y}_{\mathrm{m}}, \boldsymbol{u}) \\ \psi_2(\boldsymbol{y}_{\mathrm{m}}, \boldsymbol{u}) \\ \psi_3(\boldsymbol{y}_{\mathrm{m}}, \boldsymbol{u}) \\ \psi_4(\boldsymbol{y}_{\mathrm{m}}, \boldsymbol{u}) \\ \psi_5(\boldsymbol{y}_{\mathrm{m}}, \boldsymbol{u}) \\ \psi_6(\boldsymbol{y}_{\mathrm{m}}, \boldsymbol{u}) \end{pmatrix} \tag{34}$$

The overall structure of the neural-network-based observer for turbine engine is given in Fig. 4.

4.3 Training

It is evident from equation (34) that if the neural network can been trained to approximate the

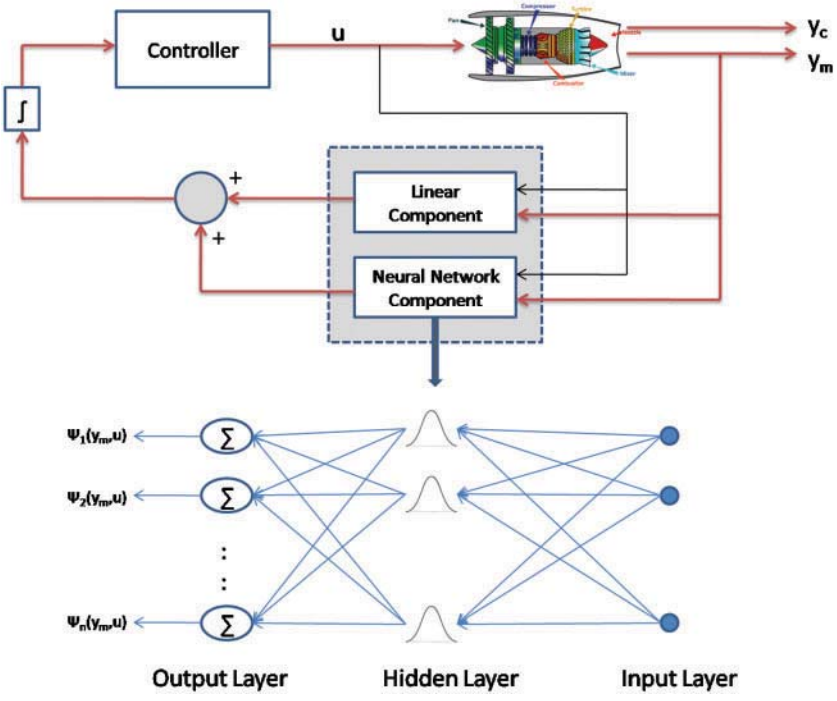

Fig. 4 Neural-network-based observer for aircraft engine

linearization error accurately that is

$$\psi_i(\boldsymbol{y}_m, \boldsymbol{u}) = \Delta_i(\boldsymbol{x}, \boldsymbol{u}) \tag{35}$$

then the closed-loop system will simply reduce to

$$\dot{\boldsymbol{e}} = (\mathbf{A} - \mathbf{L}\mathbf{C}_m)\boldsymbol{e} \tag{36}$$

If the gain $\mathbf{L}$ is chosen such that $(\mathbf{A} - \mathbf{L}\mathbf{C}_m)$ is stable, then the estimate of the state will approach the actual state asymptotically.

The training of the RBFN is performed for the transient simulation from idle to take-off. In the engine simulation model that was previously described, the controller regulates the low-pressure spool speed to a predefined set-point. The training data for the neural network are the measured outputs $\boldsymbol{y}_m$ and the inputs (to the engine) $\boldsymbol{u}$. The neural network is trained offline using the growing and pruning algorithm to minimize the error between the actual state of the engine (obtained from the computer simulation) and the state estimated by the observer since the error between the actual non-linearity (linearization error) and output of the network (estimated non-linearity) is generally not available. Suitable choices are made for the learning rate, neuron addition threshold d_{add}, and the pruning threshold parameter M_{prune} by looking at the simulation data. Generally the learning rate is kept small.

The training cycle consists of transient engine simulation data representing two sets of engine conditions:

(a) no deterioration;
(b) full deterioration.

Each cycle of training consists of presenting both sets of engine condition data sequentially to the network. During this cycle, the network weights and centres are updated and new neurons are added as needed (growing). At the end of every cycle, the significance of each neuron is evaluated using the pruning criterion and insignificant neurons are deleted (pruning). During the next cycle of training, the neuron addition threshold parameter is reduced. This will enable the network to cover a larger portion of the input space and accurate selection of RBF centres. The neural network is trained for a total of about 100 cycles. After the training is complete, the structure of the neural network, i.e. number, centres, and variances of hidden-layer neurons and weights and biases of the output layer is retained for validation.

5 RESULTS

The neural-network-based observer is used to estimate the unmeasurable performance parameters including HPT inlet temperature T41, thrust FN, and the stall margins SMW12, SMW2, SMW25 from the measured outputs. The input to the engine for the idle to take-off flight condition is shown in Fig. 5.

The following subsections illustrate the capability of the observer to estimate the performance parameters for different conditions of engine degradation during idle to take-off. The following subsections illustrate the capability of the observer to estimate the performance parameters for different conditions of engine degradation during idle to take-off (data normalized).

5.1 Case 1: No deterioration in engine

The engine model along with the neural-network-based observer is simulated for the case when there is no deterioration in the engine components. This implies that all the components are operating at their peak efficiencies. Figure 6 illustrates the engine performance parameters including the turbine inlet temperature, thrust, and various stall margins. These variables comprise the unmeasured variables vector $\boldsymbol{y}_c$.

Figure 7 describes the measured output variables that comprise the measured variables vector $\boldsymbol{y}_m$ including the temperatures and pressures at various points in the engine.

Figure 8 show the plots of the various state variables that encompass the dynamic model of the turbofan engine.

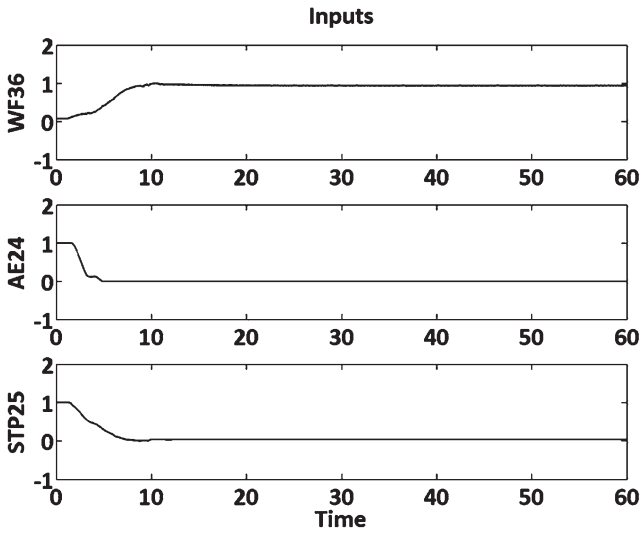

Fig. 5 Engine input variables

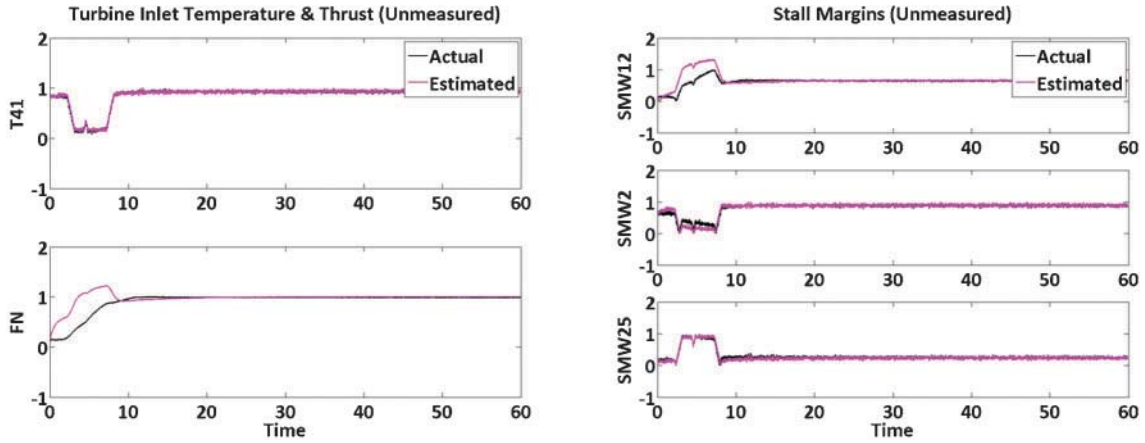

Fig. 6 Undeteriorated engine performance parameters

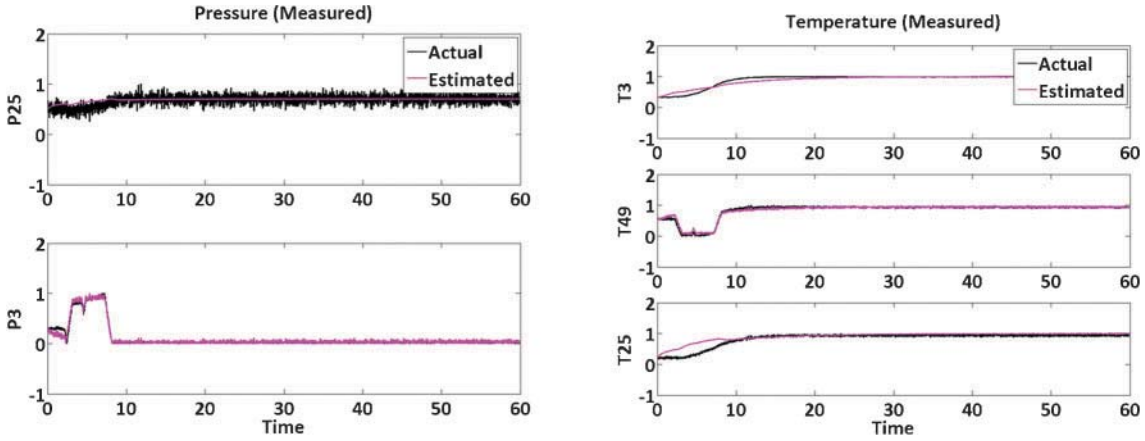

Fig. 7 Undeteriorated engine pressure and temperature

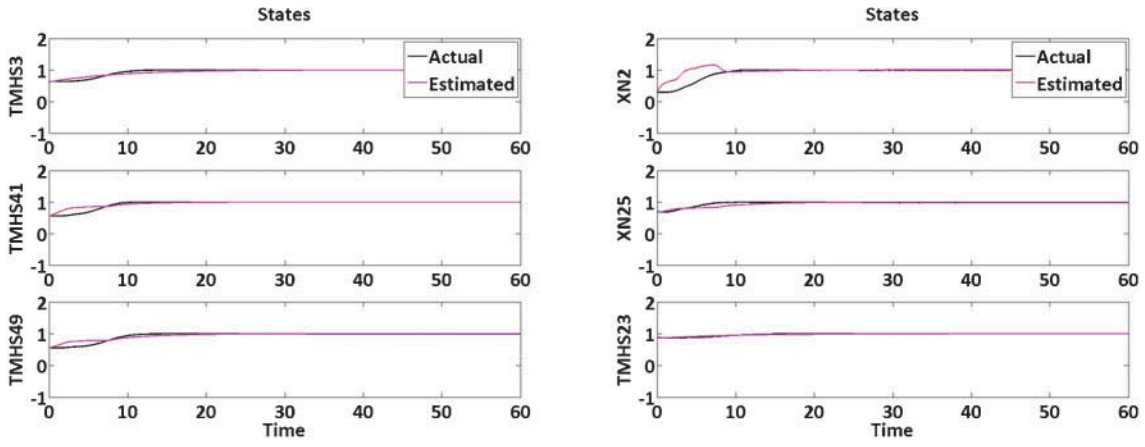

Fig. 8 Undeteriorated engine state variables

5.2 Case 2: Full deterioration in engine

In this subsection, the performance of the neural-network-based observer when the engine is in the fully deteriorated condition is considered. The efficiencies of the various engine components are assumed to be at their lowest value. Figure 9 represents the unmeasured ($\boldsymbol{y}_{\mathbf{c}}$) output variables of the fully deteriorated engine.

Figures 10 and 11 represent the measured output variables ($\boldsymbol{y}_{\mathbf{m}}$) and state variables of the fully deteriorated engine.

6 DISCUSSION

The results for the two cases show the ability of the neural-network-based observer to accurately predict the steady-state values of the unmeasured quantities. The transient response shows a comparatively larger error but this phenomenon can be related to the choice of the linear Kalman filter gain matrix. Redesigning the gain matrix appropriately will modify the bandwidth of the observer and improve the transient response characteristics. The ability of the neural-network-based observer, however, can be understood more clearly from Fig. 12. The error in the estimation of performance parameters of the undeteriorated engine is plotted for both the neural-network-based observer and the linear Kalman filter alone. It is important to note that the Kalman gain matrices ($\mathbf{L}$) utilized for both observers are the same. It is evident that the performance of the neural-network-based observer is much improved over the linear Kalman filter alone in both the transient and steady-state regions.

Another point of discussion is the choice of training data and parameters. It is evident that the method will be highly dependent on the training data and the parameters used in the algorithm. Application of this estimation technique in practical situations will require extensive training data sets that cover a wide range of operating flight conditions

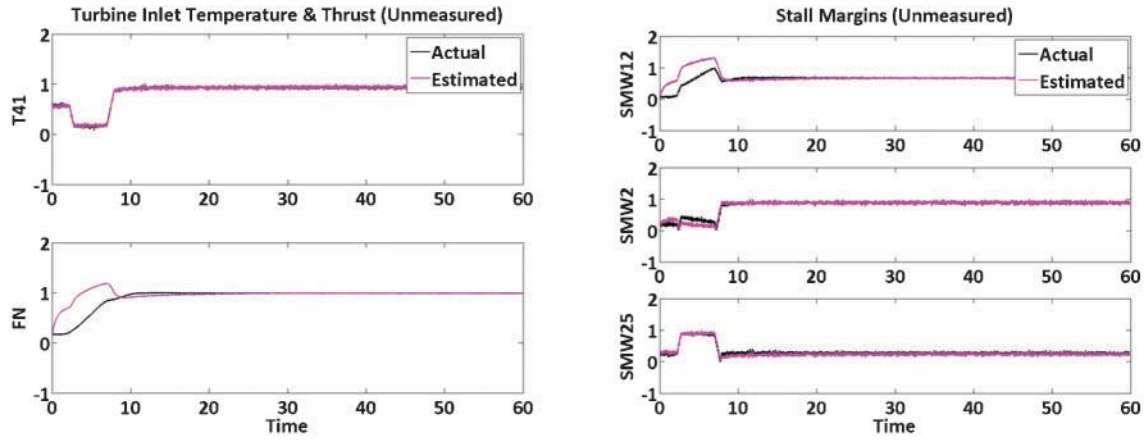

Fig. 9 Fully deteriorated engine performance parameters

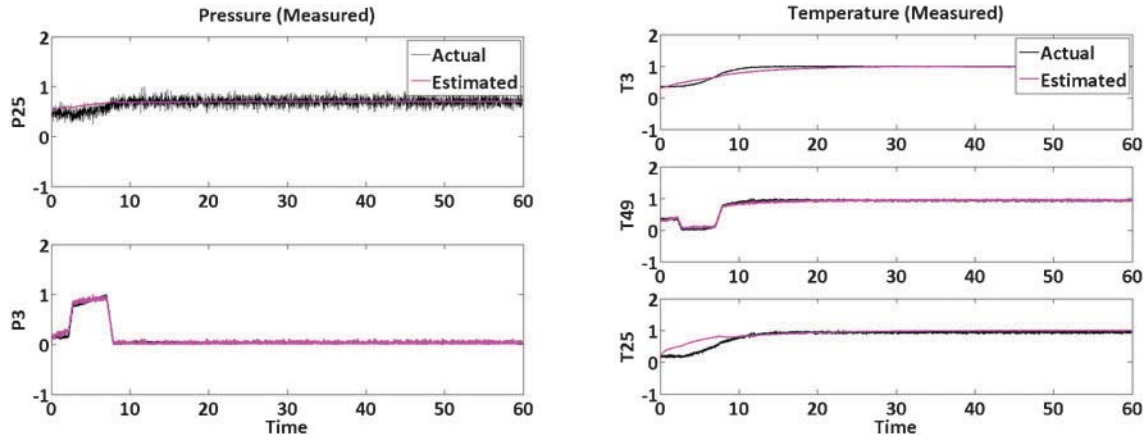

Fig. 10 Fully deteriorated engine pressure and temperature

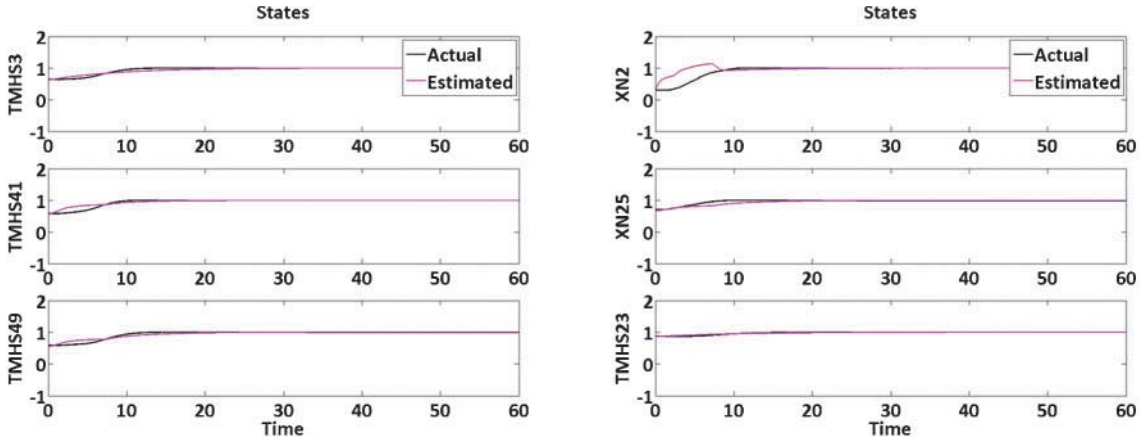

Fig. 11 Fully deteriorated engine state variables

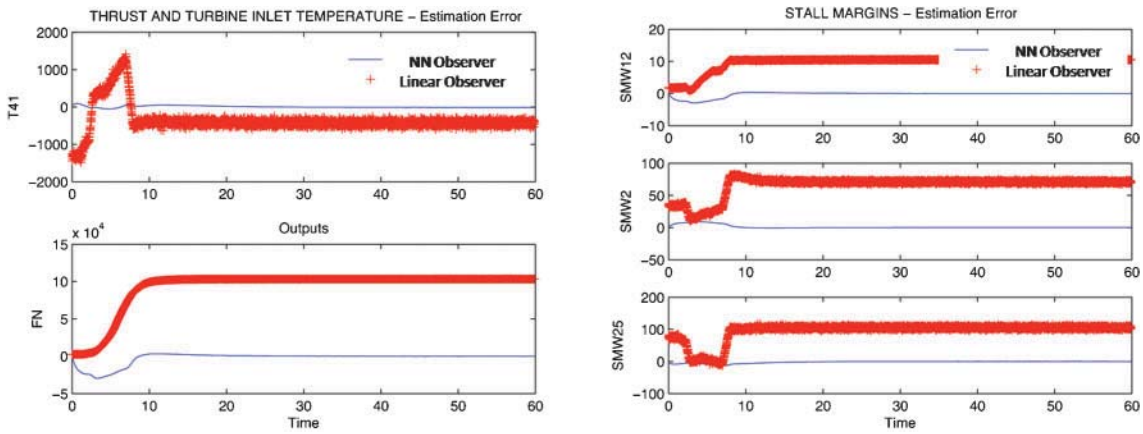

Fig. 12 Performance comparison of linear observer with neural-network-based observer

as well as different levels of engine degradation. In this paper, the observer was simulated for idle to take-off flight condition. It is critical to note that the performance of the neural-network-based observer may deteriorate for a different flight condition. It is therefore important to train the neural network not only at different levels of engine degradation at a particular flight condition but also using data from various flight conditions. One drawback of this approach, however, would be that the neural network may significantly increase in size, increasing the level of required computational resources. It is therefore suggested that multiple neural networks are trained each at a particular flight condition and a selected method of scheduling the network between each of these conditions be implemented.

7 CONCLUSIONS

This paper presented an estimation technique that augments the linear Kalman filter with a neural network to compensate for errors arising out of the non-linearities in the engine dynamic model. The HPT inlet temperature, thrust, and stall margins of the engine are estimated from the measured outputs using this neural-network-based observer during idle to take-off condition. The neural network implemented is a RBFN that is trained using a growing and pruning algorithm. By training the network with engine data at various degraded conditions of the engine, the explicit estimation of engine health parameters may be avoided.

ACKNOWLEDGEMENT

This research was performed under a research grant from NASA Glenn Research Center as a part of the Propulsion 21 initiative

REFERENCES

1 **Spang, H. A.** and **Brown, H.** Control of jet engines. *Control Engng Pract.*, 1999, **7**, 1043–1059.

2 **Patton, R. J.** and **Chen, J.** Observer-based fault detection and isolation: robustness and applications. *Control Engng Pract.*, 1997, **5**, 671–682.

3 **Litt, J. S., Simon, D. L., Garg, S., Guo, T.-H., Mercer, C., Millar, R., Behbahani, A., Bajwa, A.,** and **Jensen, D. T.** A survey of intelligent control and health management technologies for aircraft propulsion systems. *J. Aerosp. Comput., Inform. Commun.*, 2004, **1**, 543–563.

4 **Gupta, S., Ray, A., Sarkar, S.,** and **Yasar, M.** Fault detection and isolation in aircraft gas turbine engines. Part 1: underlying concept. *Proc. IMechE, Part G: J. Aerospace Engineering*, 2008, **222**(G3), 307–318. DOI: 10.1243/09544100JAERO311.

5 **Kobayashi, T., Simon, D. L.,** and **Litt, J.** Application of constant gain extended Kalman filter for in-flight estimation of aircraft engine performance parameters. ASME Turbo Expo: Power for Land, Sea and Air, Reno-Tahoe, NV, 2005.

6 **Simon, D.** and **Simon, D. L.** Aircarft turbofan engine health estimation using constrained Kalman filtering. *Trans. ASME, J. Engng Gas Turbines Power*, 2005, **127**, 323–328.

7 **Dewallef, P., Leonard, O.,** and **Mathioudakis, K.** Online aircraft engine diagnostic using a soft-constrained Kalman filter. ASME Turbo Expo 2004: Power for Land, Sea and Air, Vienna, Austria, 2004.

8 **Litt, J. S.** An optimal orthogonal decomposition method for Kalman-filter based turbofan engine thrust estimation. *Trans. ASME, J. Engng Gas Turbines Power*, 2008, **130**, 011601-1–011601-12.

9 **Yedavalli, R. K., Shankar, P., Siddiqi, M.,** and **Behbahani, A.** Modeling, diagnostics and prognostics of a two-spool turbofan engine. The 41st AIAA/ASME/SAE/ASEE Joint Propulsion Conference, Tucson, AZ, 2005.

10 **Maggiore, M., Ordonez, R., Passino, K. M.,** and **Adibhatla, S.** Estimator design in jet engine applications. *Engng Applic. Artif. Intell.*, 2003, **16**, 579–593.

11 **Volponi, A. J.** Use of hybrid engine modeling for on-board module performance tracking. ASME Turbo Expo 2005: Power for Land, Sea and Air, Reno-Tahoe, NV, 2005.

12 **Vargas, J. A. R.** and **Hemerly, E. M.** Adaptive observers for unknown general nonlinear systems. *IEEE Trans. Syst., Man Cybern.*, 2001, **31**, 683–690.

13 **Lainiotis, D. G.** and **Plataniotis, K. N.** Adaptive dynamic neural network estimators. IEEE World Congress on Computational Intelligence, Orlando, Florida, 1994.

14 **Singh, D., Pandey, D. P.,** and **Chauhan, D. S.** Radial basis neural network state estimation of electric power networks. IEEE International Conference on Electric Utility Deregulation, Restructuring and Power Technologies, Hong Kong, 2004.

15 **Chen, C. I., Napolitano, M. R.,** and **Chen, C.-L.** A new learning algorithm for neural network state estimation in active vibration control. *Smart Mater. Struct.*, 1992, **1**, 250–257.

16 **Meghdari, A., Naderi, D.,** and **Alam, M. R.** Neural-network based observer for real-time tipover estimation. *Mechatronics*, 2005, **15**, 989–1004.

17 **Abdollahi, F., Talebi, H. A.,** and **Patel, R. V.** A stable neural network based observer with application to flexible-joint manipulators. *IEEE Trans. Neural Netw.*, 2006, **17**(1), 118–128.

18 **Shankar, P.** and **Yedavalli, R. K.** A neural network based adaptive observer for turbine engine parameter estimation. ASME Turbo Expo 2006: Power for Land, Sea and Air, Barcelona, Spain, 2006.

19 **Bay, J. S.** *Fundamentals of linear state space systems*, 1999 (McGraw Hill, New York).

20 **Haykin, S.** *Neural networks: a comprehensive foundation,* second edition, 1999 (Prentice Hall, New Jersey).

APPENDIX

Notation

A	plant matrix $\mathbf{A} \in \Re^{n \times n}$
B	input/control matrix $\mathbf{B} \in \Re^{m \times n}$
C	output matrix $\mathbf{C} \in \Re^{p \times n}$
D	feedthrough matrix $\mathbf{D} \in \Re^{p \times m}$
$\boldsymbol{e}$	error vector $\boldsymbol{e} \in \Re^{n}$
L	observer gain matrix $\mathbf{L} \in \Re^{n \times p}$
$\boldsymbol{u}$	input/control vector $\boldsymbol{u} \in \Re^{m}$
$\boldsymbol{x}$	state vector $\boldsymbol{x} \in \Re^{n}$
$\boldsymbol{y}$	output vector $\boldsymbol{y} \in \Re^{p}$

Engine model variables

AE24	variable bleed valve
FN	net thrust, N
P3	combustor inlet pressure (Pa)
P25	high-pressure compressor (HPC) inlet pressure (Pa)
SMW2	low-pressure compressor stall margin
SMW12	fan stall margin
SMW25	HPC stall margin
STP25	variable stator vane
T3	combustor inlet temperature (°C)
T25	HPC inlet temperature (°C)
T41	HPT inlet temperature (°C)
T49	low-pressure turbine (LPT) inlet temperature (°C)
TMHS3	HPC metal temperature (°C)
TMHS23	booster metal temperature (°C)
TMHS41	HPT nozzle metal temperature (°C)
TMHS49	LPT metal temperature (°C)
XN2	low-pressure spool speed (r/min)
XN25	high-pressure spool speed (r/min)
WF36	fuel flow (kg/s)