Proceedings of GT2006
ASME Turbo Expo 2006: Power for Land, Sea and Air
May 8-11, 2006, Barcelona, Spain

GT2006-91099

NEURAL NETWORK MODELS FOR USAGE BASED REMAINING LIFE COMPUTATION

Girija Parthasarathy and Sunil Menon[1]
Vehicle Health Management Lab
Honeywell Aerospace
3660 Technology Drive
Minneapolis, Minnesota 55418

Kurt Richardson, Ahsan Jameel, Dawn McNamee, Tori Desper, Michael Gorelik, and Chris Hickenbottom
Honeywell Engines, Systems and Services
111 South 34th Street
Phoenix, AZ

Email: girija.parthasarathy@honeywell.com (contact author)

ABSTRACT

In engine structural life computations, it is common practice to assign a life of certain number of start-stop cycles based on a standard flight or mission. This is done during design through detailed calculations of stresses and temperatures for a standard flight, and the use of material property and failure models. The limitation of the design phase stress and temperature calculations is that they cannot take into account actual operating temperatures and stresses. This limitation results in either very conservative life estimates and subsequent wastage of good components, or in catastrophic damage because of highly aggressive operational conditions which were not accounted for in design. In order to improve significantly the accuracy of the life prediction, the component temperatures and stresses need to be computed for actual operating conditions. However, thermal and stress models are very detailed and complex, and it could take on the order of a few hours to complete a stress and temperature simulation of critical components for a flight. The objective of this work is to develop dynamic neural network models, that would enable us to compute the stresses and temperatures at critical locations, in orders of magnitude less computation time than required by more detailed thermal and stress models. This work expands on the work done previously [1] where a linear system identification approach was developed. The current paper describes the development of a neural network model and the temperature results achieved in comparison with the original models for Honeywell turbine and compressor components. Given certain inputs such as engine speed and gas temperatures for the flight, the models compute the component critical location temperatures for the same flight in a very small fraction of time it would take the original thermal model to compute.

NOMENCLATURE

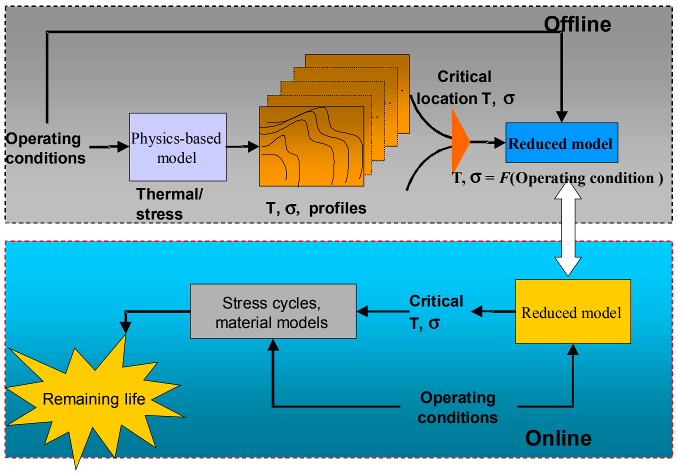
u	Input
y	Output
•	Predicted output
f, F	Activation functions
w, W	Network weights
A, B, F, C, D	Polynomials identifying model structure
q	Time shift operator notation
e	Disturbance signal
t	Time or sample
na	Number of past outputs
nb	Number of past inputs
nk	Time delay
.	Regression vector
.	Vector containing weights
.	Rotational speed
.	Metal temperature at critical location

[1] Currently with Eaton Corporation Innovation Center, 7945 Wallace Road, Eden Prairie, MN 55344.

. .	Gas temperature
W_g	Gas flow

INTRODUCTION

Accurate life prediction for turbine engines has become increasingly important in recent years as military and commercial aircraft fleet age, and engine component failures cause unplanned maintenance and repair. Failure modes such as Low Cycle Fatigue (LCF), Stress Rupture (SR) or Creep are caused by operational stresses (either cyclic or constant) to the engine component material. The common industry practice today is to count engine start stop cycles, and decrement it from a pre-calculated number for life, to arrive at remaining life. This pre-calculated number is overly conservative for obvious safety reasons. Probabilistically, we throw away 1000 components to remove the unknown one that is theoretically predicted to be in a failed state [2]. Therefore, increasing the life prediction confidence is of utmost importance if we are to increase the service life of aircraft turbomachinery components, or if we are to predict imminent failure of an overused engine. Consequently, work is being done to increase our prediction and analysis capability for structural failure. Advances in fatigue prediction material models, damage tolerance life prediction model, modeling crack growth, and assessing strength have been made [3-6]. In all these advances, however, stresses and temperature of a critical component location need to be computed for the operation conditions it sees, before these advanced techniques can yield practical results for health monitoring. Since stresses and temperatures are computed with detailed numerical models, it is impractical to do the same on an every flight basis. In order to be able to capture the important information necessary in a timely manner, a reduced model concept was introduced [1]. The results of using impulse response model for model reduction were demonstrated for a Honeywell engine [1]. The idea of the model reduction (illustrated in Figure 1) was to be able to build a dynamic model of the temperature or stress at a critical location as a function of changing gas temperature or engine speed. In ref (1), the impulse response model was used as the system was close to linear. However, in highly non-linear systems, the temperature or stress response cannot be predicted accurately by an impulse response model. In this paper, we describe the development and use of dynamic neural networks as a tool for model reduction for near real-time computation of critical location temperatures and stresses.



The objective of model reduction is to take an existing numerical model (thermal, stress, fluid flow) that predicts detailed spatial and temporal quantities, and condense it so that instead of the detailed predictions, only task-specific targeted quantities are predicted, albeit at a fraction of the cost to computational resources. This usage of the knowledge of design phase numerical model in an online system is novel. Another unique aspect of this work is the creation of a dynamic model, as opposed to a static model, since dynamic stresses and temperatures are very important in the calculation of remaining life.

Model reduction as explained above, can be achieved in various ways such as system identification methods, neural networks, regression techniques, and so on. In the case of finite element models, however, there is literature in the area of *reduced-order* modeling, which is distinct from the approach in this paper. Algorithms have been developed for solving transient thermal problems in a reduced subspace of the original discretization space [7]. Mass and stiffness matrices can be reduced with the Guyan reduction method [8], decreasing the size of the model. However, later literature indicates that it cannot be successfully applied to dynamic thermal problems and that in fact, cruder finite element approximations do better [9]. Model size reduction using component mode synthesis has been applied to turbine engine disc temperature calculation [10]. In this method, nodal degrees of freedom are divided into two sets . active and omitted. Active degrees of freedom are translated into the reduced model, while omitted ones are replaced by the most important modal shapes.

The above model-order reduction techniques require all inputs to the finite element model including nodal boundary conditions, which can be in the hundreds or more, and are applicable when the output required is of the same magnitude as the full numerical model. These reduced order models are strictly applicable to only the full finite element model part of the process. In other words, we cannot include in the reduced order model, models that provide the boundary conditions themselves, such as the performance model. These reduced order modeling techniques have been generally applied for ease of use of complex models or combination of models. In our case, however, the reduced model needs to predict temperature and stress in certain locations, and also reduce the input space as much as possible.

With these considerations in mind, work was done towards developing system identification techniques for the purpose of computing stress and temperature at the life-critical locations Honeywell propulsion engines [1]. The impulse response method was used to compute temperatures and stresses. This technique lends itself well to systems that exhibit linear or close to linear behavior. For components that show non-linear behavior, however, other methods need to be explored. We present the use of *dynamic neural networks* for computing stresses and temperatures at critical locations of Honeywell turbomachinery components. The following sections provide a brief background of neural networks and describe the approaches taken in developing the current method.

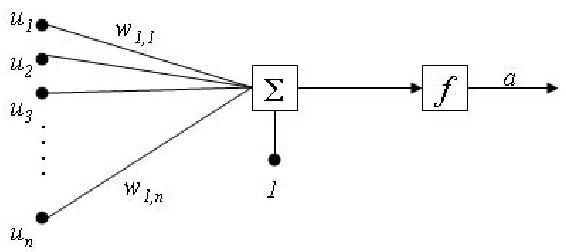
NEURAL NETWORKS

Neural Networks are non-linear empirical model structures. Neural networks have been used in deriving models for complex and nonlinear systems, where developing physics-based models could be extremely time-consuming and not very accurate. Neural networks have been used for diverse fields such as control, insurance, banking, image processing and medicine. In our case, we wish to derive a reduced dynamic model for computation of critical location temperature and stress of an engine component for application such as health or usage monitoring of aircraft systems. A very brief introduction to neural networks is given below. For more in-depth study, other references should be consulted [11, 12].

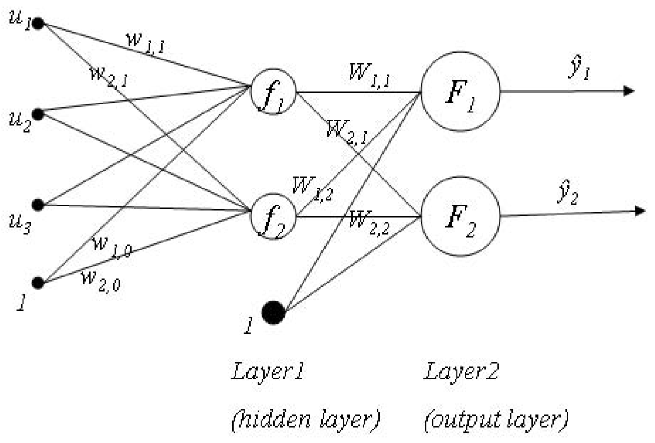
Figure 2(a) illustrates a neuron, the basic processing element for a neural network. A neuron takes a number of inputs (*u*), multiplies them by a constant (weight), then sums them up and uses the result as input to a singular valued activation function *f* as shown. Figure 2(b) illustrates a feedforward neural network with two layers, three inputs, two hidden nodes and two outputs. Here the weighted inputs are processed through function *f*. This is the first layer of the network. The outputs from this layer are in turn processed through the output layer function *F*, giving us the network outputs, •. This can be mathematically represented as

$$\hat{y}_i(w,W) = F_i\left(\sum_{j=1}^{q} W_{ij} f_j\left(\sum_{l=1}^{m} w_{jl}u_l + w_{j0}\right) + W_{i0}\right) \tag{1}$$

Common activation functions used for *f* and *F* are the sigmoid or hyperbolic tangent (*tanh*) function, linear transfer function, and the step transfer function.



(a)



(b)

Figure 2: Neural Network Schematics (a) Neuron and (b) Example Network

Training the network is the process of finding the optimum weights *w* and *W* that will give the best match with the known target outputs from the data. This is accomplished through methods such as backpropagation and its variations, and quasi-Newton methods such as Levenberg-Marquard all of which differ in speed, memory requirements, etc. The neural network described above, also known as multilayer perceptron network, is directly applicable to static systems. In the next section, we describe the extension to developing models of dynamic systems.

DYNAMIC NEURAL NETWORKS

System Identification background

In general, given a system's input and output data, the science of deriving a dynamic model from the data is called system identification. There are many different methods and model structures in system identification, as applied to linear systems. Dynamic neural networks [2] complements the suite of system identification techniques by extending to non-linear systems. A general input-output linear model for a single-output system with input *u* and output *y* can be written as

$$A(q^{-1})y(t) = \frac{B(q^{-1})}{F(q^{-1})}u(t-nk) + \frac{C(q^{-1})}{D(q^{-1})}e(t) \tag{2}$$

where *u* denotes the input; *e(t)* is the disturbance signal; *A, B, C, D* and *F* are polynomials in terms of the time shift operator *q*; and *nk* denotes the time delay.

$$\begin{aligned} A(q^{-1}) &= 1 + a_1q^{-1} + \ldots\ldots + a_nq^{-n} \\ B(q^{-1}) &= b_0 + b_1q^{-1} + \ldots\ldots + b_nq^{-n} \\ C(q^{-1}) &= 1 + c_1q^{-1} + \ldots\ldots + c_nq^{-n} \\ D(q^{-1}) &= 1 + d_1q^{-1} + \ldots\ldots + d_nq^{-n} \end{aligned} \tag{3}$$

The time shift operator is a notation that works on a function thus:

$$q^{-d}x(t) = x(t-d) \tag{4}$$

Some linear model structures are ARX (Auto-Regressive, External input), OE (Output Error), ARMA, ARMAX, etc. We describe the ARX and OE linear model structures below, since they are going to be used in the model structure for the dynamic neural network. For a linear ARX model, the above equation simplifies to

$$A(q^{-1})y(t) = B(q^{-1})u(t-nk) + e(t) \tag{5}$$

Ignoring the noise term, the above equation can be written as

$$\begin{aligned} &y(t) + a_1y(t-1) + \ldots + a_{na}y(t-na) = \\ &b_1u(t-nk) + \ldots + b_{nb}u(t-nk-nb+1) \end{aligned} \tag{6}$$

Once the parameters a_1 through a_n and b_1 through b_n are determined, the output *y* can be predicted in terms of past inputs and past outputs. The number of past inputs and outputs to be used is given by the order of the system*[na nb nk]*,where *na* is the number of past outputs, *nb*, the number of past inputs and *nk*, the time delay. For multi-input systems, *nb* and *nk* are vectors with as many entries as the number of input channels. For multiple outputs, *na* is a vector as many entries as the number of outputs.

For a linear OE model, the above equation simplifies to

$$y(t) = [B(q^{-1}) / F(q^{-1})]u(t-nk) + e(t) \qquad (7)$$

The parameters of the OE model structure are estimated using a prediction error method. In this case, the prediction • can be computed in terms of past inputs and past output predictions.

Dynamic Neural Networks

For dealing with non-linear system identification, one approach is to use the feed forward network, but use input

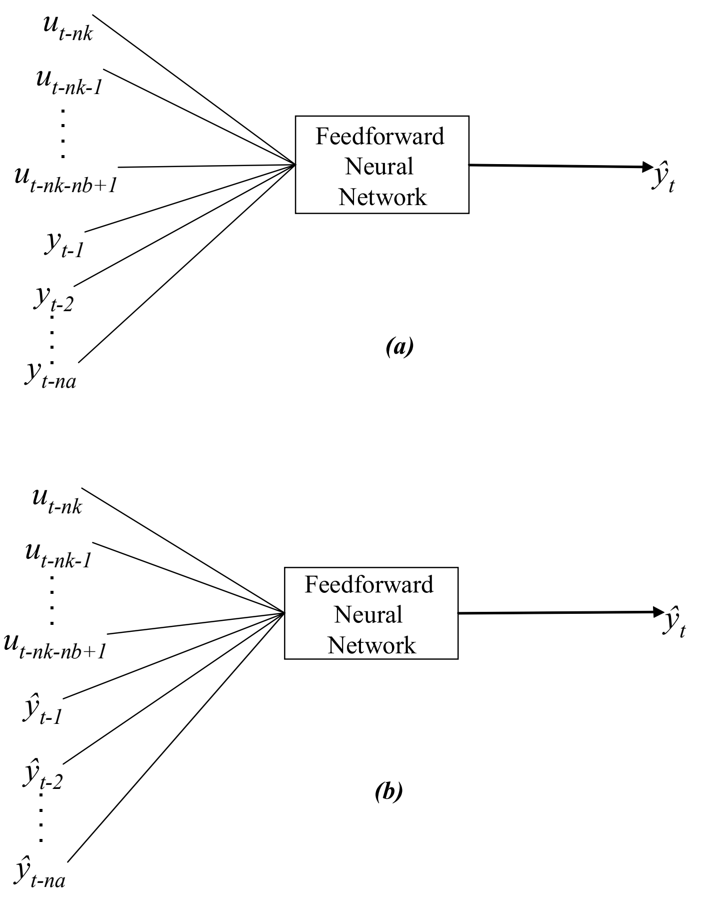


Figure 3: (a) NNARX model structure and (b) NNOE model structure

structures similar to the linear system identification model structures such as ARX, OE, ARMAX, etc. Norgaard [13] has implemented a toolbox for just such an approach and the same approach has been followed in this work. The neural network implementation of ARX model structure is called NNARX (for Neural Network ARX), and that for OE model structure is called NNOE, and so on. Non-linear counterparts to the linear model structures are obtained by

$$\hat{y}(t) = g[. \;,. \;] \qquad (8)$$

where g is the function realized by the neural network, . is the regression vector, . is the vector containing the weights.

For the above representation, the regressors for the NNARX model structure is given by

$$\begin{aligned} .\ (t) = [&y(t-1),....y(t-na), \\ &u(t-nk),....u(t-nk-nb+1)] \end{aligned} \qquad (9)$$

Similarly, the regressors for the NNOE model structure is given by

$$\begin{aligned} .\ (t) = [&\hat{y}(t-1),....\hat{y}(t-na), \\ &u(t-nk),....u(t-nk-nb+1)] \end{aligned} \qquad (10)$$

As we can see, the NNOE regressors have the past predictions of the output rather than actual past outputs. These representations are depicted schematically in Figure 3.

DYNAMIC NEURAL NETWORKS DEVELOPMENT AND IMPLEMENTATION

Problem domain definition and Selection of input(s)

Before the dynamic neural network model is built, the domain and inputs need to be defined well. Figure 4 shows different domain boundaries that can be considered for this problem, As in the previous work on impulse response model, we choose domain boundary #3 [1]. Selecting boundary #3 implies that various performance model outputs such as engine station parameters (gas temperature, pressure, flow) and engine speed would be used as inputs to the dynamic neural network model.

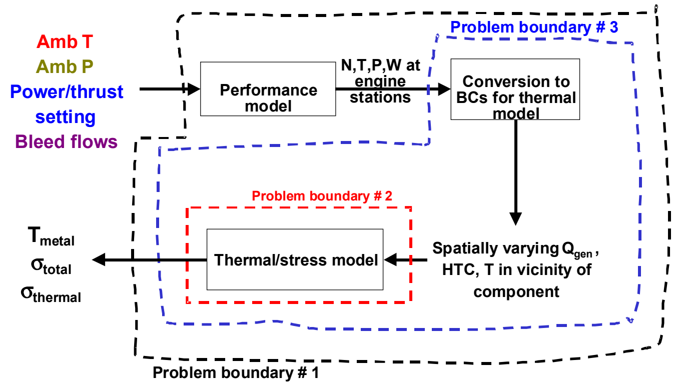


Figure 4: Problem domain and input selection

Model structure selection

The next step in the development is selection of the model structure. In our case, the problem is unique from the system identification point of view in that the outputs (critical location metal temperature or stress) are not likely to be measured or computed with the original models during operation. However, we do have the original model outputs from different flight profiles, for training purposes. Therefore, during training, the NNARX model structure is used to get the full benefit of the original model outputs. During validation, the NNOE model structure is used, so that past predictions are used as inputs in place of the past outputs. This ensures that during operation, when no past outputs are likely to be present, the past predictions can be used, and that we validate our approach to real conditions.

We use the 2-layer network architecture as shown in the example. The other phases of model structure selection is the selection of number of hidden nodes, selection of model order (number of past inputs and outputs) and time delay. Several experimental runs were conducted to arrive at good combinations of these parameters, with help from physical insights into the system responses. The hyperbolic tangent (*tanh*) and linear functions were used as the activation functions for the hidden and output layers. All implementation was done in Matlab. Results are described in the next section. Once the weights are found, the output predictions can be computed by equation (1).

Critical Node Temperature Predictions for Engine Component

The dynamic neural network method was applied to predict the temperature at several critical locations of a Honeywell turbomachinery component. The inputs used were rotational speed (N), related gas temperature (Tg) and gas flow (Wg) near the component. A 2-layer architecture of one hidden layer and one output layer was used. The usual procedure of tweaking the inputs, model structure, the network and the system order was followed to learn from the interim results and obtain the best model. Finalizing the number of past inputs to be used for each physical input (N, Tg and Wg) takes some trial and error and physical insight for that particular location. For example, for a location with a slow thermal response, the number of past inputs should be higher, In the following results, 8 hidden nodes, with 3 physical inputs were used. The order of the system used for these results were:
Number of past outputs (na): 1
Number of past inputs (nb): [4 8 2] (for inputs N, Tg and Wg)
Number of time delays (nk) : [1 1 1] (for inputs N, Tg and Wg)
The regressors in this case can be written as (see equation 9):

$$. \ (t) = [T(t-1), N(t-1), N(t-2), N(t-3), N(t-4)$$
$$T_g(t-1), T_g(t-2), T_g(t-3), T_g(t-4), T_g(t-5),$$
$$T_g(t-6), T_g(t-7), T_g(t-8), W_g(t-1), W_g(t-2)]$$

The above regressors serve as inputs (u_i) to the neural network representation in equation 1. The actual metal temperature predicted from the original model serves as • during training. We use the linear and hyperbolic tangent functions respectively for activation functions *F* and *f*.

Once the physical inputs, the number of past inputs and the number of hidden neurons have been identified, the model structure definition is complete. Norgaard's toolbox is used to train the network and obtain the weights using the training datasets. The toolbox uses the Levenberg-Marquard method for training the network. Once the weights *w* and *W* are found, the outputs are calculated by use of equation (1). Because of the availability of toolboxes, the bulk of the work is done in experimenting to find the optimum model structure, inputs and model order. Several realistic flight profiles were used for the training and validation of the neural networks model. Out of the 9 flight profiles available, 6 were used for training and the rest for validation.

Figure 5 through Figure 8 show the results comparing the temperatures obtained by the dynamic neural network model and the original finite element thermal model. Figure 5 displays the comparison for flight profiles (a), (b) and (c) for critical location 1. The data in these flight profiles were used for training and the maximum error between the original and reduced model is very small. Figure 6 shows the comparison for three flight profile data that the neural network model had not seen (validation data) for the same critical location. These flight profiles are a more rigorous test of the reduced model performance, and as can be seen from the plots, the reduced model results match very well with the original model results, with maximum error of 10%.

Figure 7 shows the plots of critical location 2 temperatures against mission time, for flight profiles used for training. While (a) and (b) represent a full flight profile, (c) represents minor fluctuations that could be part of a full flight. In all cases, the error rate is small, within 5%. Figure 8 shows the critical location 2 temperatures, for validation flight profiles that were not used for training. Other than in flight profile (a), the error between the original and reduced models show good agreement, within 10%.

Discussion

One of the sources for the errors shown is the difference in the model structure used for validation versus training. As discussed in the section Model Structure Selection, we used the NNARX model structure for training, and the NNOE model structure for validation. This is because actual past outputs (metal temperatures) will not be available during operation of the real system. As a result, the regressor *T(t-1)* is the actual metal temperature from the original model for training, and the neural network model predicted metal temperature during validation.

Another cause for some of the errors is the 'one-size fits all' approach for the implementation where the same model structure used for all critical locations. Structure in this context means the inputs, model order, delays and number of neurons. During the initial development of this approach, several locations in one component were being evaluated for temperature estimation. Since each location was physically situated differently with the gas path and its fluctuations, each had its own thermal response characteristic. It would have been ideal to fit each with a separate neural net model. However, at the demonstration phase of this work, the question to be resolved was 'would this approach be feasible to estimate metal temperature within a certain percentage?' Considering that we were evaluating multiple locations with multiple flight profiles, the additional permutations of varying the model structure (inputs, number of past inputs, delays, number of neurons) would have been very large. It was decided to fix the model inputs and order at an optimum for all locations. By developing a different model structure that fits the temperature response characteristics of each location better, these errors can be minimized. During implementation, different model inputs, order, delays and neurons *were* used for different locations and components, to meet the accuracy constraints of temperature prediction.

In neural network modeling, under and over-training can be sources of errors. With under-training, sufficient data is not available to capture all features of the system. Since a model is the source of training data, undertraining is not a serious concern here. Overtraining can happen if the amount of data is very large and the network may be capturing insignificant details (such as those with noise) at the expense of capturing the significant ones. In our case, noise is not a factor because the data is generated by a model. However, some parts of a flight profile such as a long steady state phase (see Figure 5a) could contribute to overtraining at the expense of capturing shorter transients. In such a case, the repetitive or redundant data can be removed during training, taking care to keep enough of that feature for training, and added back during testing. This was done for the flight profiles shown in Figure 5a and Figure 7a.

Another point to be noted is that the validation flight profiles were chosen to be different from the training flight profiles. The dynamics in each case could be different, although the static conditions enveloped the validation sets. The validation sets illustrate the scenario, wherein a dynamic behavior is being estimated with the model trained on different operating dynamics, and hence possibly different dynamic behavior.

A concern, or rather a challenge, of the neural network technique is finding the right amount of training data to cover all possible operating conditions and dynamics so that the model generalizes well to new inputs during operation. There may well be a case when the network encounters inputs or dynamics that are out of the data range that it was trained on. The network predictions could then have large errors. The remedy is to train the network on all known limits of operating conditions and dynamics. Periodic retraining and updates of network weights with real operating data, would also go a long way to remedy this limitation.

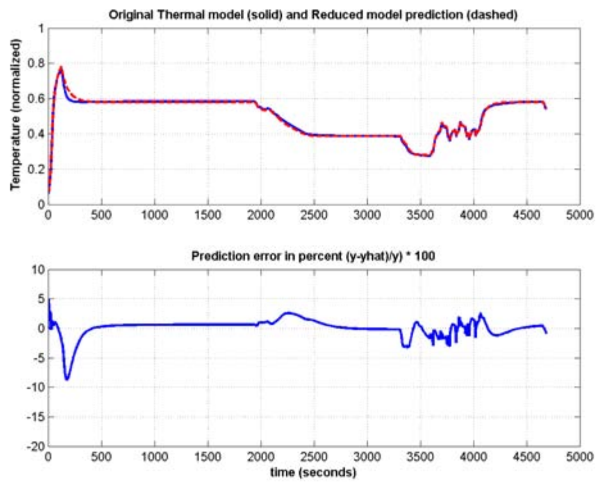
CONCLUSIONS

A model reduction technique for computing critical component parameters for remaining life prediction was demonstrated. A dynamic neural network model was developed for the computation of engine critical location metal temperatures. The dynamic neural network model reduces the original thermal model of a turbomachinery component, so that the temperatures can be computed on the fly if needed. Results for two critical locations of a component were presented. The results show that such techniques can be applied with minimal error for computing remaining life estimates, while taking into account the actual operating conditions of the turbomachinery.

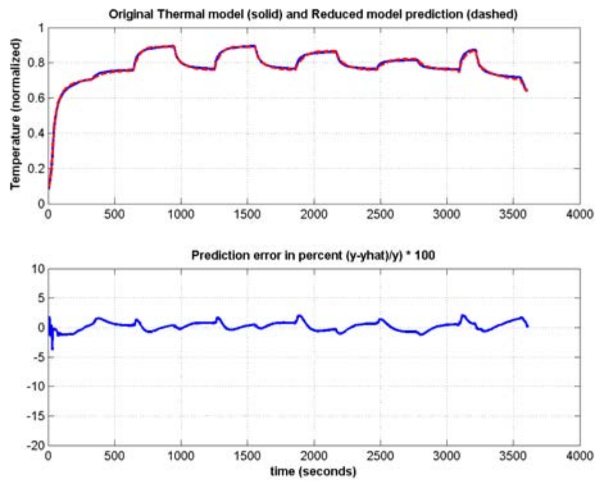
The same approach can also be used for the computation of thermal stress at the critical location. There are several other applications for the model reduction concept, where the potential for detailed models can be exploited for estimating quantities that cannot be measured.

REFERENCES

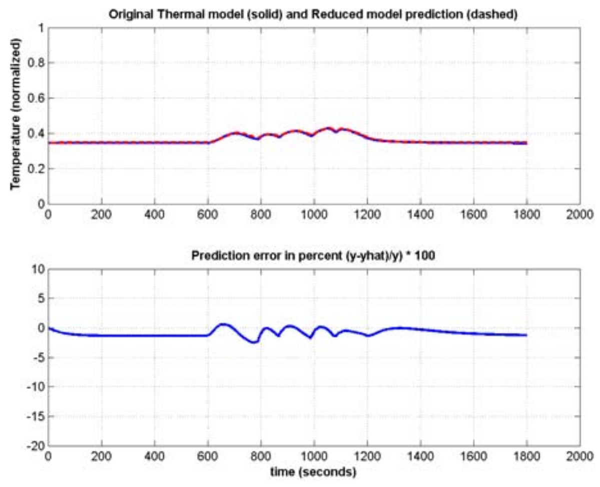
1. Parthasarathy, G. and Allumallu, S., 2004, ‘Reduced Models for Rotating Component Lifing’, Proceeding of ASME Turbo Expo 2004, June 14-17, 2003, Vienna, Austria.
2. Larsen, J. M., Russ, S. M., Rosenburger, A. H., John, R., Fecke, T. and Rasmussen, B., 2002, ‘Achieving the Potential of Material Prognosis for Turbine Engines’, Presented at DARPA Bidders Conference on Materials Prognosis, Washington, D.C.
3. Fujimoto, W. T., 2002, ‘A Critical Assessment of Fatigue Prediction Methodologies for Rotorcraft Members’, presented at 6th joint FAA/DoD/NASA Conference on Aging Aircraft, San Francisco, CA.
4. Forth, S. C., Everett, R. A. and Newman, J. A., 2002, ‘A Novel Approach to Rotorcraft Damage Tolerance’, presented at 6th joint FAA/DoD/NASA Conference on Aging Aircraft, San Francisco, CA.
5. Kwon, Y. S., Fawaz, S. A., and Ingram , J. E., 2002, ‘3D Finite Element Modeling of MSD-Cracked Structural Joints’, 6th joint FAA/DoD/NASA Conference on Aging Aircraft, San Francisco, CA.
6. Cope, D. A., Lacy, T. E. and Luzar, J. J., 2002, ‘Application of Advanced Fracture Mechanics Methods for KC-135 Fuselage Structural Integrity Evaluation’, 6th joint FAA/DoD/NASA Conference on Aging Aircraft, San Francisco, CA.
7. Cardona, A. and Idelsohn, S., 1986, ‘Solution of Non-linear Thermal Transient Problems by a Reduction Method’, International Journal for Numerical Methods in Engineering, **23**, pp. 1023-1042.
8. Guyan, R. J., 1965, ‘Reduction of Stiffness and Mass Matrices’, AIAA Journal, **3(2)**, p.380.
9. Bushard, L. B., 1981, ‘On the Value of Guyan Reduction in Dynamic Thermal Problems’, Computers and Structures, **13**, pp. 525-531.
10. Botto, D., Zucca, S., Gola, M. M. and Salvano, S., 2002, ‘A Method for On-line Temperature Calculation of Aircraft Engine Turbine Discs’, Proceedings of ASME Turbo Expo 2002, Amsterdam, The Netherlands.
11. Haykin, S., 1999, ‘Neural Networks, A Comprehensive Foundation’, 2nd Ed., Prentice Hall, New Jersey.
12. Norgaard, M., Ravn, O., Poulsen, N. K. and Hansen, L. K., 2000, ‘Neural Networks for Modeling and Control of Dynamic Systems’, Springer-Verlag, London, UK.
13. Norgaard, M, 2000, ‘Neural Network Based System Identification Toolbox, Version 2’, Technical University of Denmark.
14. Ogunnaike, B. A. and Ray, W. H., 1994, ‘Process Dynamics, Modeling, and Control’, Oxford University Press, New York.
15. Korb, R., Skorjanz, P., and Jorgl, H.P., 1997, ‘Black box modeling of an industrial combustion plant’, SYSID'97, 11th IFAC Symposium on System Identification, Kitakyushu, Japan.
16. Bittani , S. and Lovera, M., 1997 , ‘Identification of linear models for a hovering helicopter rotor’, 11th IFAC Symposium on System Identification, Kitakyushu, Japan.
17. L. Ljung, 1987, ‘System Identification: Theory for the User’, P.T.R. Prentice Hall, New Jersey.



(a)

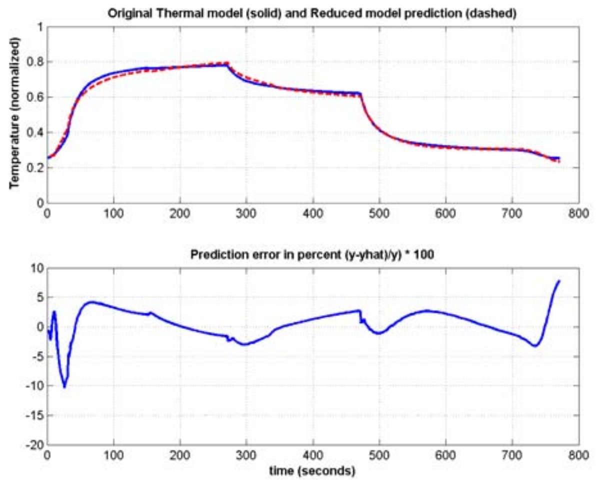


(b)

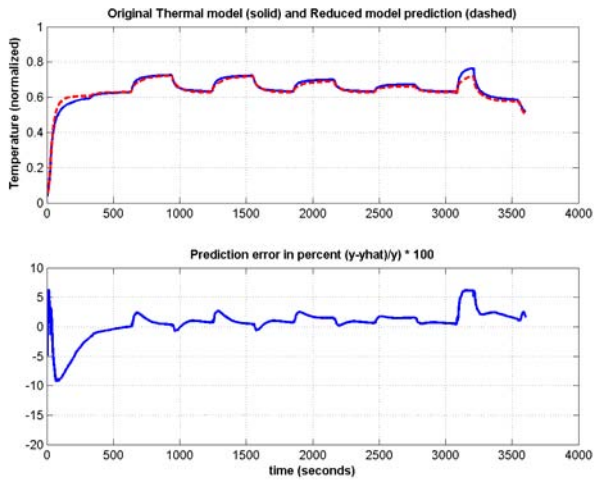


(c)

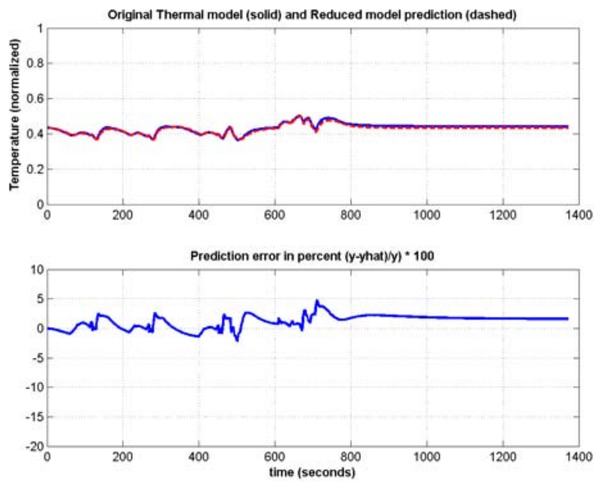
Figure 5: Critical Location 1 Temperatures for Different Flight Profiles (Training data)



(a)

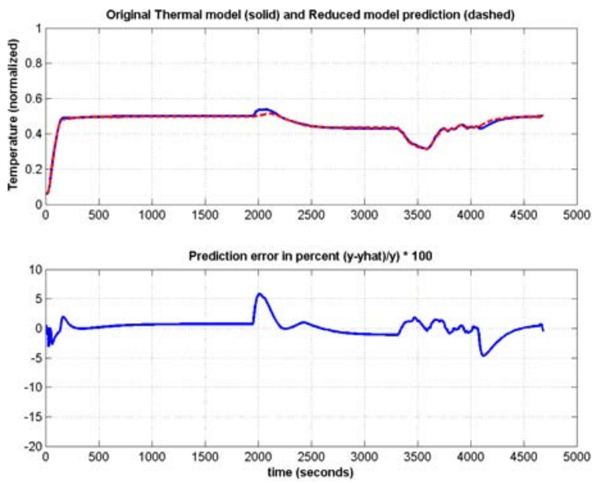


(b)

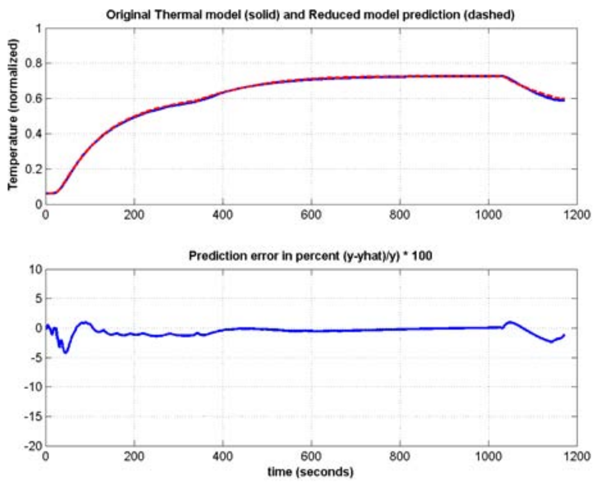


(c)

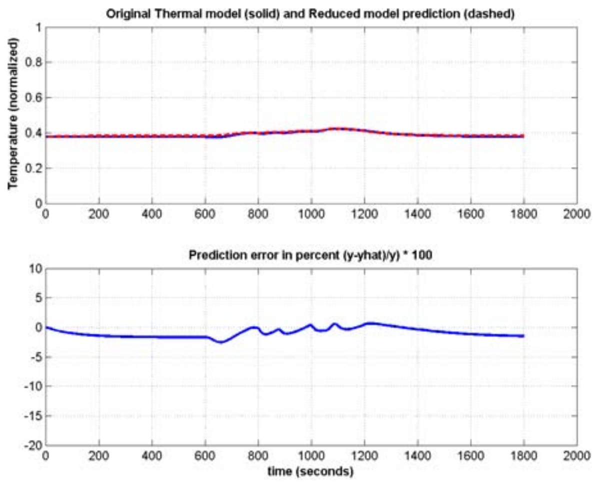
Figure 6: Critical Location 1 temperatures for Different Flight Profiles (Validation data)



(a)

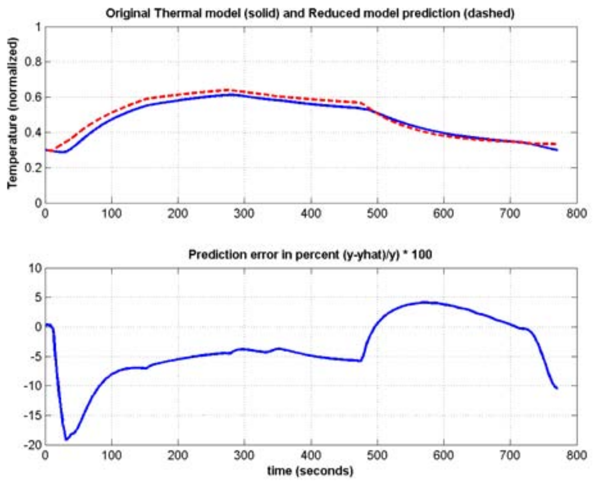


(b)

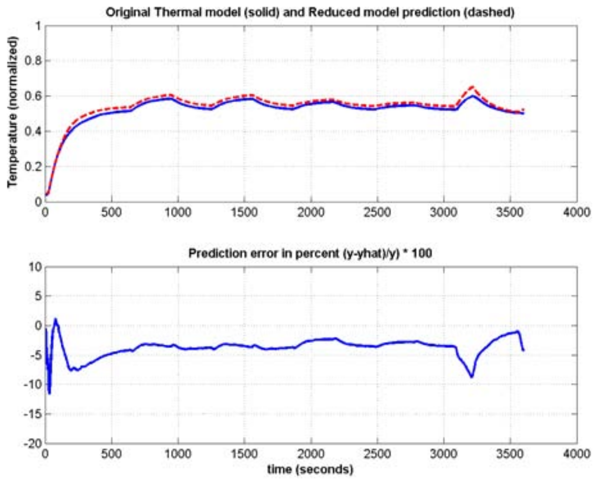


(c)

Figure 7: Critical Location 2 Temperatures for Different Flight Profiles (Training data)



(a)



(b)