An Instance-Based Method for Remaining Useful Life Estimation for Aircraft Engines

Article *in* Journal of Failure Analysis and Prevention · January 2008

DOI: 10.1007/s11668-008-9118-9

J Fail. Anal. and Preven.
DOI 10.1007/s11668-008-9118-9

TECHNICAL ARTICLE—PEER-REVIEWED

An Instance-Based Method for Remaining Useful Life Estimation for Aircraft Engines

Feng Xue · Piero Bonissone · Anil Varma · Weizhong Yan · Neil Eklund · Kai Goebel

Abstract Under customer service agreements (CSA), engine operational data are collected and stored for monitoring and analysis. Other data sources provide damage assessments that are either provided post-maintenance or analytically assessed. This paper takes advantage of these data and investigates local fuzzy models to determine the remaining useful life (RUL) of an engine or engine component. Local fuzzy models are related to both kernel regressions and locally weighted learning. The particular local models described in this paper are not based on individual models that consider the track history of a specific engine nor are they based on a global average model that would consider the collective track history of all the engines. Instead, for a given engine or component, this local fuzzy model defines a cluster of peers in which each of these peers is a similar instance to this given engine with comparable operational characteristics; the RUL prediction for this given engine is obtained by a fuzzy aggregation of its peers' RUL. We combine the fuzzy instance-based approach with an evolutionary framework for model tuning and maintenance. This evolutionary tuning process is repeated periodically to automatically update and improve the fuzzy models such that they can be updated to date with the latest collection of data. This fuzzy instance-based approach is applied to predicting the RUL of a commercial engine validated with post-maintenance assessment.

Keywords Evolutionary algorithm · Fuzzy · Instance-based · Life-cycle · Prognosis · Remaining useful life (RUL)

Reprinted with permission from *Integration of Machinery Failure Prevention Technologies into Systems Health Management, Proceedings of the 61st Meeting of the Society for Machinery Failure Prevention Technology*, Society for Machinery Failure Prevention Technology, 2007, on CD-ROM.
This work was done while the author was with GE Global Research.

F. Xue (✉) · P. Bonissone · A. Varma · W. Yan · N. Eklund
GE Global Research One Research Circle, Niskayuna, NY 12309, USA
e-mail: xue@research.ge.com

P. Bonissone
e-mail: bonissone@research.ge.com

A. Varma
e-mail: varma@research.ge.com

W. Yan
e-mail: yan@research.ge.com

N. Eklund
e-mail: eklund@research.ge.com

K. Goebel
RIACS, NASA Ames Research Center, Moffett Field, CA 94043, USA
e-mail: goebel@email.arc.nasa.gov

Introduction

Performance Based Logistics (PBL) and Customer Service Agreements (CSA) are the main growth drivers for the manufacturers of military and commercials platforms and propulsion systems. A common element for their success is the ability of efficiently performing Conditioned-based Maintenance (CBM). CBM enables the repair/replacement of key components as a function of their condition, i.e., Remaining Useful Life (RUL) rather than their calendar life. This capability is enabled by Prognostics (P) and optimized by Health Management (HM), in what we commonly refer to as PHM. Specifically, PHM plays a critical role in PBL performance by linking real-time diagnostic and prognostics information to PBL metrics; it provides increased mission readiness and supportability assessment; it enables fleet-wide asset HM; and represents the foundation for logistics horizontal integration.

The goal of integrated HM of equipment is to maintain their availability for normal utilization with proper maintenance strategies. A key component in realizing such goal is to determine when to perform proper maintenance. Prognostics is defined as the estimation of RUL, measured in units of time. The time estimate typically has an associated uncertainty that is described as a probability density centered on the actual estimate. Operators can choose a confidence that allows them to incorporate a risk level into their decision-making. RUL estimates can be used to improve aircraft engines' operational and maintenance decisions. RUL estimates allow operators to change operational characteristics (such as load), which in turn may prolong the life of the component. It also allows planners to account for upcoming maintenance and start a logistics process that supports a smooth transition from faulted to fully functioning equipment. By improving the accuracy of RUL estimates, we can improve the quality of these decisions. With a good RUL estimation, one can schedule the proper maintenance while minimizing or avoiding the interruption of normal operations. Such decisions based on RUL estimation could increase equipment utilization and improve the operation safety of the equipment.

In the field of prognostics, we focus on developing approaches to estimating the RUL of equipment or its component to help the decision making process to come up with such maintenance strategies. In this paper, we discuss our approach in the context of an aircraft engine or engine components. In the general domain of equipment life prediction, the task of predicting RUL is not straightforward. Ordinarily, remaining life is conditional on future usage conditions such as load and speed, among others. In some cases, the usage profile of the equipment does not change much from one time to another, as in the case of commercial aircraft engines. Every cycle of operation includes take-off, climbing, cruise, and landing. For each of these phases, load variations over cycles are relatively small. Thus, an average load can be used over the span of the RUL prediction. On the other hand, other equipments (e.g., military engines) do not follow an average load profile, making RUL estimation also a function of load variation (e.g. mission profile), while building the prognostic model.

In this paper, we study the RUL prediction at the module level of an aircraft engine that allows the reasoning over the entire module without needing to assess the particular damage propagation mechanics at the part level. We introduce a RUL estimator (fuzzy instance model, or FIM) for a faulted aircraft engine module, based on aggregating RUL from its selected peers (similar instances). The tunable parameters are used for the selection of peers based on system operational data. The fuzzy instance model aggregates peers' outputs to estimate the RUL of an aircraft engine module. The rest of the paper is organized as follows: the second section is the background and related work; the fuzzy instance model is described in the third section; a case study and its results are described in the fourth section; and section five contains the conclusion.

Background

Diagnostics

Diagnostics involves making fault diagnosis based on incipient fault signatures before the interruptive failure occurs. This fault detection and diagnosis provides a trigger point for the prognostic algorithms. In the absence of abnormal conditions (or fault conditions) the best estimates for remaining life are often fleet wide statistics expressed by Weibull curves or other suitable mechanisms. Condition-based systems depend on reliable fault diagnostics to initiate the prognostic algorithms. It is therefore important to optimize the detection and diagnosis capability to attain optimal prognostics.

Prognostics Approaches

In the context of RUL prediction within the field of prognostics, we particularly prefer to employ prognostics in the presence of an indication of abnormal wear, faults, or other non-normal situation. Such prognostics is closely linked with incipient fault detection and diagnostics. Indeed, it is critical to provide accurate and quick detection and diagnostics to allow prognostics to operate afterwards since diagnostic (such as faulted module identification) and detection information (such as accurate fault initiation time) flow down to the prognostic model. In the absence of any evidence of damage or faulted condition, prognostics reverts to statistical estimation of average fleet-wide life rather than boiling down to individual cases.

Common approaches to prognostics either employ a model incorporating detailed materials-based knowledge or using a data-driven pattern recognition approach. The former approach is very costly and can only be utilized on a few parts with particular failure modes. The latter assumes the availability of complete run-to-failure data and run-time sensor measurements in order to deduct the system damage state for RUL estimation [1, 2].

Obviously, developing a prognostic model (even data driven) for each individual component is very costly and not realistic except for a few very critical components of the aircraft engine. There are various resource limitations and benefits tradeoffs in making such a decision on the development of prognostics models with different levels of details. For instance, a detailed data-driven prognostics model for a particular component such as a bearing requires the availability of particular online sensors for collection of operational measurements.

Another possible approach is to take advantage of time series data where equipment behavior has been tracked via sensor measurements from normal operation to equipment failure. When a reasonably sized set of these observations exists, pattern recognition algorithms can be employed to recognize these trends and predict remaining life [3, 4]. These approaches use either functional approximations or decision trees based on run-to failure data. When there are multiple prognostics models (physics-based or data-driven), effective fusion strategies can be employed to further enhance the accuracy of the RUL estimation as shown in [4].

However, often times, run-to failure data are not available because, when the observed system is complex and expensive and safety is critical, faults are caught before they lead to system failure. This deprives the data driven approach its critical information. On the other hand, there are many possible sources on the preliminary RUL estimations. For instance, we may have: field observations leading to a distribution of RUL; laboratory experiments measuring the wear of the equipment; inspections of the equipment before major overhauls; statistical models, parametric or nonparametric, for instance Weibull, neural networks, providing preliminary estimates. In this case, a fusion of multiple estimates could provide more reliable preliminary RUL estimates. This paper assumes the availability of some preliminary RUL estimates.

In previous work [5–7], we applied a peer-based approach to forecast the reliability of an asset within a fleet for the purpose of asset selection to improve mission reliability. This method focused on the overall platform (locomotive, aircraft) without providing any insight about the RUL of its components or sub-components. Furthermore, this method assumed the availability of operational, maintenance, and environmental data for each platform. However, this information may not always be available. In this paper, we only use operational data in our analysis.

Locally Weighted Learning

A relevant approach in the literature to the fuzzy instance model presented in this paper is nonparametric regression approaches such as Nadaraya–Watson regression and locally weighted learning. In both cases, there are no pre-specified model structures for the learning of some relations. The underlying principles for these approaches are similar data points in their input feature space would have similar results for their output.

The idea of avoiding pre-constructed models and creating a local model when needed can be traced back to memory-based approaches [8, 9] and lazy-learning [10]. The Nadaraya–Watson regression (also referred as kernel regressions) is equivalent to locally weighted regressions for data points located away from the boundaries within which they are distributed. Assuming we have a set of historical training data points: $\{(X_1, y_1), \ldots (X_N, y_N)\}$

An estimation of a probe point $(X_0, ?)$ would be the following:

$$\hat{f}(X_0) = \frac{\sum_{i=1}^{N} K_h(X_0, X_i) \cdot y_i}{\sum_{i=1}^{N} K_h(X_0, X_i)}$$

$K_h(X_0, X_i)$ is a kernel function defining the similarity between points X_0 and X_i; h is the smoothing parameter that is usually optimized by cross validations; and there are different forms of kernel functions. In the kernel regression, the aggregation is a weighted average method.

Locally weighted regression is a further development (generalization) of kernel regression. In the early development of kernel regression, the local aggregation model is zero order (e.g. weighted average as mentioned above); while locally weighted regression often uses higher order local models. Linear models are often used due to a limited number of local data points and for better generalization.

Fuzzy Instance Model

Fuzzy Instance Model

Instance-based reasoning (IBR) relies on a collection of previously experienced data that can be kept in their raw representation. Unlike Case-Based Reasoning (CBR), they do not need to be refined, abstracted and organized as cases. Like CBR, IBR is an analogical approach to reasoning, since it relies upon finding previous instances of *similar* problems and uses them to create an ensemble of local models. Hence the definition of similarity plays a critical role in the performance of IBR's. Typically, similarity will be a dynamic concept and will change over the use of the IBR. Therefore, it is important to apply learning methodologies to define and adapt it. Furthermore, the concept of similarity is not crisply defined, creating the need to allow for some degree of vagueness in its evaluation. We addressed this issue by evolving the design of a similarity function in conjunction with the design of the attribute space in which the similarity was evaluated. Specifically we used the following four steps (Fig. 1):

(1) *Retrieval* of similar instances from the Data Base (DB)

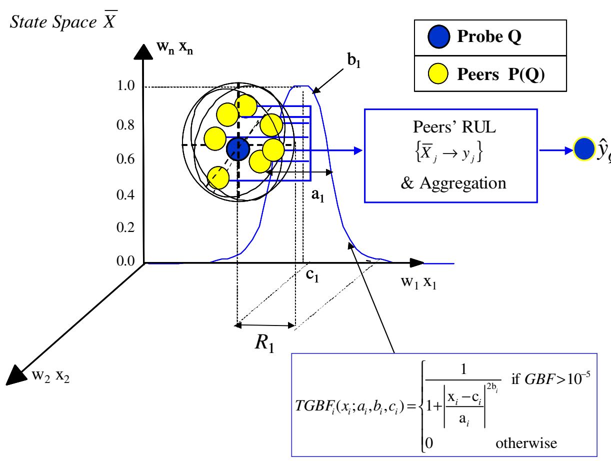


Fig. 1 Instance-based model

(2) *Evaluation of similarity measure* between the probe and the retrieved instances
(3) *Creation of local models* using the most similar instances (weighted by their similarity measures)
(4) *Aggregation of outputs* of local model to probe.

Retrieval

The retrieval step consists in finding all DB instances whose behavior is similar to the probe. These instances are the probe's potential peers and can be seen as points in an n-dimensional feature space. For instance, let us assume that a probe Q has an associated n-dimensional vector of values for each potential attribute: $\bar{X}_Q = [x_{1,Q}, x_{2,Q}, \ldots, x_{n,Q}]$. A similar n-dimensional vector characterizes each instance u_i in the fleet with an attribute vector: $u_j = [x_{1,j}, x_{2,j}, \ldots, x_{n,j}]$;

For each dimension i, we define a *Truncated Generalized Bell Function*, $TGBF_i(x_i;a_i,b_i,c_i)$, centered at the value of the probe c_i, which represents the degree of similarity along that dimension. Specifically:

$$TGBF(x_i;a_i,b_i,c_i) = \left\{\begin{array}{ll}\left[1+\left|\frac{x_i-c_i}{a_i}\right|^{2b_i}\right]^{-1} & \text{if } \left[1+\left|\frac{x_i-c_i}{a_i}\right|^{2b_i}\right]^{-1} > \varepsilon \\ 0 & \text{otherwise}\end{array}\right\} \quad (1)$$

where ε is the truncation parameter, e.g. $\varepsilon = 10^{-5}$.

Since the parameters c_i in each $TGBF_i$ are determined by the values of the probe, each $TGBF_i$ has only two free parameters, a_i and b_i, to control its spread and curvature. In a coarse retrieval step, we extract an instance in the DB if all of its features are within the *support* of the TGBF's. Now we can formalize the retrieval step. *P(Q)*, the set of potential peers of Q to be retrieved, is composed of all instances within a range from the value of Q: $P(Q) = \{u_j, j = 1, \ldots, m | u_j \in N(Q)\}$ and *N(Q)*, a neighborhood of Q, is defined by the constraint $|x_{i,Q} - x_{i,j}| < R_i$ for all potential attributes i for which their corresponding weight is non-zero. R_i is half of the support of the $TGBF_i$, centered on the probe's coordinate $x_{i,Q}$.

Similarity Evaluation

Each $TGBF_i$ is a membership function representing the degree of satisfaction of constraint $|x_{i,Q} - x_{i,j}| < R_i$. Thus, $TGBF_i$ measures the *closeness* of an instance around the probe value $x_{i,Q}$ along the *ith* attribute. For a potential peer P_j, we evaluate $S_{i,j} = TGBF(x_{i,j};\ a_i, b_i, x_{i,Q})$, its similarity with the probe Q along each attribute i. The values (a_i, b_i) are design choices initially chosen manually, and later determined by the evolutionary algorithms (EA). Since we want the most similar instances to be the closest to the probe along *all* n attributes, we use a similarity measure defined as the intersection (minimum) of the constraint-satisfaction values:

$$S_j = \mathrm{Min}_{i=1}^{n}\{S_{i,j}\} = \mathrm{Min}_{i=1}^{n}\{TGBF(x_i;a_i,b_i,x_{i,Q})\} \quad (2)$$

Equation (2) implies that each attribute is equally important in computing similarity. In our case, however, we consider each criterion to have a different relevance in that computation. Therefore, we attach a weight w_i to each attribute A_i and we extend the notion of similarity measure between P_j and the probe Q using a weighted minimum operator:

$$\begin{aligned} S_j &= \mathrm{Min}_{i=1}^{n}\{\mathrm{Max}[(1-w_i), S_{j,i}]\} \\ &= \mathrm{Min}_{i=1}^{n}\{\mathrm{Max}[(1-w_i), TGBF(x_i;a_i,b_i,x_{i,Q})]\} \end{aligned} \quad (3)$$

where $w_i \in [0,1]$. The set of values for the weights $\{w_i\}$ and of the parameters $\{(a_i, b_i)\}$ are critical design choices that impact the proper selection of peers. In this section we assume a manual setting of these values. In the next section, we will explain their derivation using evolutionary search.

Creation of Local Models

Let us assume that for a given probe Q we have retrieved m peers, $P_j(Q)$, $j = 1,\ldots,m$. Each peer $P_j(Q)$ has a similarity measure S_j with the probe. First, we need have RUL values for these peers. Then, we use an aggregation mechanism, based on the similarities of the peers, to determine the final output for the probe Q. The RUL values for those peers can be generated from difference sources as aforementioned in the introduction section.

Aggregation of Local Models' Outputs

We need to combine the individual RUL value from each similar instance y_j to generate the RUL estimate y_Q for the probe Q. We define this aggregation as the *similarity weighted average*, by computing the weighted average of the peers' individual predictions using their normalized similarity to the probe as a weight:

$$y_Q = \frac{\sum_{j=1}^{m} S_j \times y_j}{\sum_{j=1}^{m} S_j} \tag{4}$$

The aggregation can be formed in a different way in which a locally weighted linear model is first constructed based on the peers, and the prediction for the probe Q is estimated based on this local linear model. To construct the linear model, we can simply make a diagonal matrix based on similarities such as:

$$W_{jj} = S_j, j = 1, \ldots, m$$

The weighted instance vectors and their corresponding RUL values can be easily constructed as:

$$Z = WU \quad V = WY$$

where, $U_j = u_j$, and u_j is the attribute vector for the jth instance; and $Y_j = y_j$ is the corresponding RUL value. The RUL prediction for the probe Q can be estimated from the *locally weighted model* constructed from the peers in the following way:

$$\hat{y}(Q) = \overline{X}_Q (Z^T Z)^{-1} Z^T V \tag{5}$$

Given the critical role played by the weights $\{w_i\}$, and by the search parameters $\{(a_i, b_i)\}$ it was necessary to create a methodology that could generate the best values according to our metrics (RUL estimates accuracy). The process of estimating RUL for an engine or engine component is broken down into an off-line training process and an on-line monitoring process.

Off-Line Tuning

The fuzzy instance model has a number of parameters that require tuning to identify an optimal combination of values. We employ an EA for this purpose (Fig. 2). The EA is composed of a population of individuals ("chromosomes"), each of which contains a vector of elements that represent distinct tunable parameters within the FIM configuration. Examples of tunable parameters include the range of each parameter used to retrieve neighbor instances and the relative parameter weights used for similarity calculation. Each chromosome specifies a vector of weights $[w_1, w_2, \ldots w_D]$ and defines an instance of the attribute space used by its associated classifier. If $w_i \in \{0, 1\}$, we perform *attribute selection*, i.e., we select a crisp subset of the universe of potential attributes. If $w_i \in [0, 1]$ we perform *attribute weighting*, i.e., we define a fuzzy subset of the universe of potential attributes.

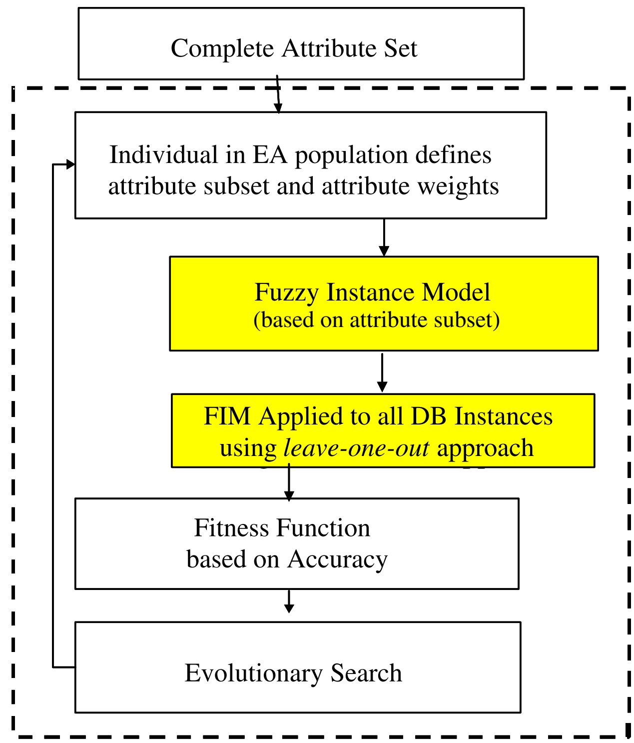


Fig. 2 Evolutionary search for best model

$$[w_1 \; w_2 \; \ldots \; w_n] \; [(a_1, b_1), \; (a_2, b_2), \; \ldots, \; (a_n, b_n)] \tag{6}$$

where $w_i \in [0, 1]$ *for* attribute *weighing* or
$w_i \in \{0, 1\}$ *for* attribute *selection*
n = Cardinalit y of universe of features U, $n = |U|$
$d = \sum_i^n w_i$ (fuzzy) cardinality of selected features
(a_i, b_i) = Parameters for GBF_i

In summary, the first part of the chromosome, containing the weights vector $[w_1, w_2, \ldots w_n]$, defines the attribute space (e.g. the FIM structure) and the relevance of each attribute in evaluating similarity. The second part of the chromosome, containing the vector of pairs $[(a_1,b_1),\ldots(a_i,b_i,)\ldots(a_n,b_n)]$ defines the parameter for the retrieval and similarity evaluation.

The fitness function can be based in overall precision in diagnostics (e.g., classification problems), or in prediction accuracy in prognostics.

On-Line Process

In the on-line process, the model is executed when the faults are detected. An n-dimensional vector of values is derived to represent the attribute for this instance such as

$Q = [x_{1,Q}, x_{2,Q}, \ldots, x_{n,Q}]$

Specific diagnostic information, such as failure modes, can be incorporated in this attribute representation to improve the estimation performance. The process as described in the *fuzzy instance model* section is to provide RUL estimate for the new probe Q based on instances in the DB. This n-dimensional vector of values is constantly updated with new operational data gathered from the aircraft engine as the engine is still serving on wing. RUL estimates are also updated with new information gathered from the engine.

Preliminary Results

We built a prognostic reasoner based on the FIM described in the previous section and we tested it in experiments on aircraft engines. We created a pool of fault cases over the period of 2002–2005. The operational data measured for each flight are used to train the fuzzy instance model. In particular, all these cases are HPT (High Pressure Turbine) shroud burn related faults. The combination of data analysis and domain knowledge helps us focus our analysis on three sensor measures: EGT (Exhausted Gas Temperature), WF (Fuel Flow), N2 (Core Speed). An aircraft can be operated in different flight conditions, i.e., flight envelopes. Variation of flight envelope affects the operational parameters even though the internal conditions of the aircraft engine remain the same. Therefore, we work on the operational parameters after flight envelope corrections, for example DEGT (Delta EGT) is the delta EGT after flight envelope correction.

In Fig. 3 we use a sample of eight fault cases and we show the cross plots of (N2, DEGT), and (WF, DEGT) for each case. The green dots indicate their normal operation region, while the magenta line shows the trajectory starting from fault initiation point (the blue asterisk) to the end of the operation (the red asterisk). From the visualization, it is shown that the DEGT increases as the operational trajectory moves from fault initiation to the end of operation. Generally, WF increases and N2 decreases for most of the trajectories.

As we can observe from these figures, the absolute values of these operational parameters are different from one case to another. If we look at the red asterisks across all these cases, they are in different positions from one to another, while all their RUL's are equal to zero.

Features Extraction

To overcome this problem, we performed another transformation of the operational parameters. In particular, we focused on the *deltas* of operation parameters *after fault initiation* over their mean values before fault initiation. In addition, we also calculated the integral of these deltas over the time span after fault initiation to current time as the features for the prediction of RUL. This feature captures the compound factor of fault magnitude and duration, and plays an important role in the RUL prediction. In Fig. 4, the delta and integral features are plotted against RUL for two examples.

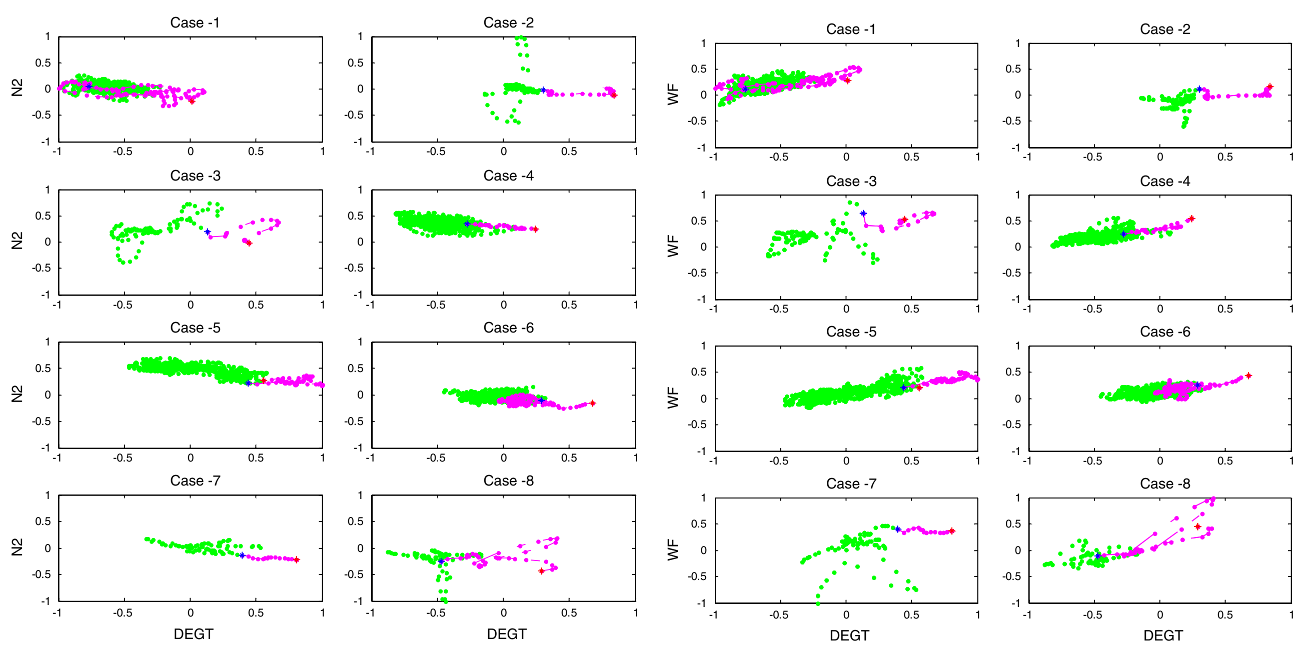


Fig. 3 Cross plots of operational parameters for eight fault cases

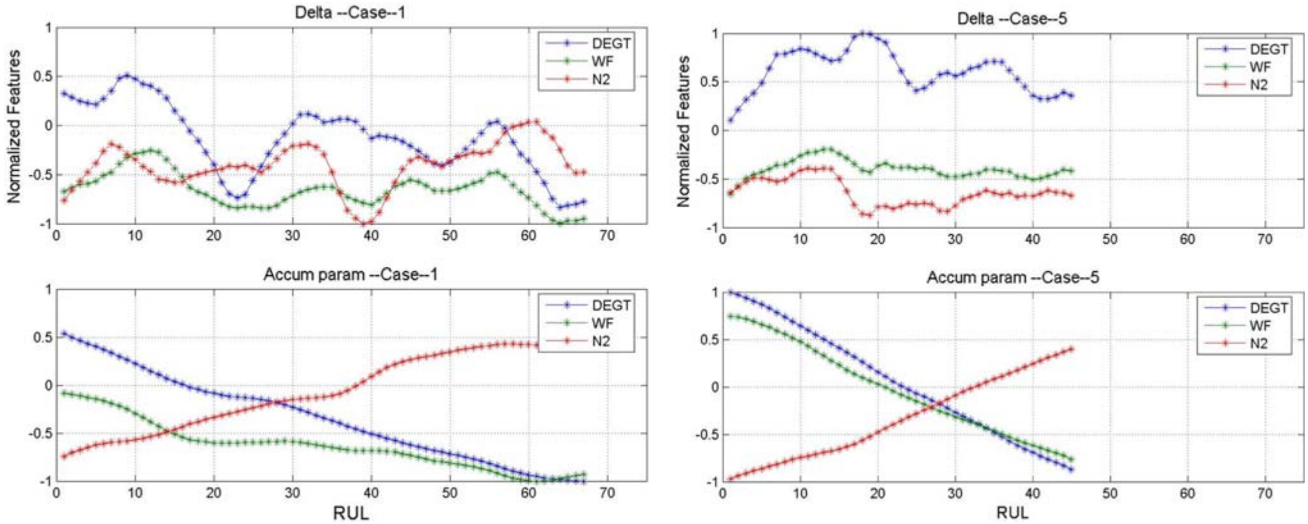
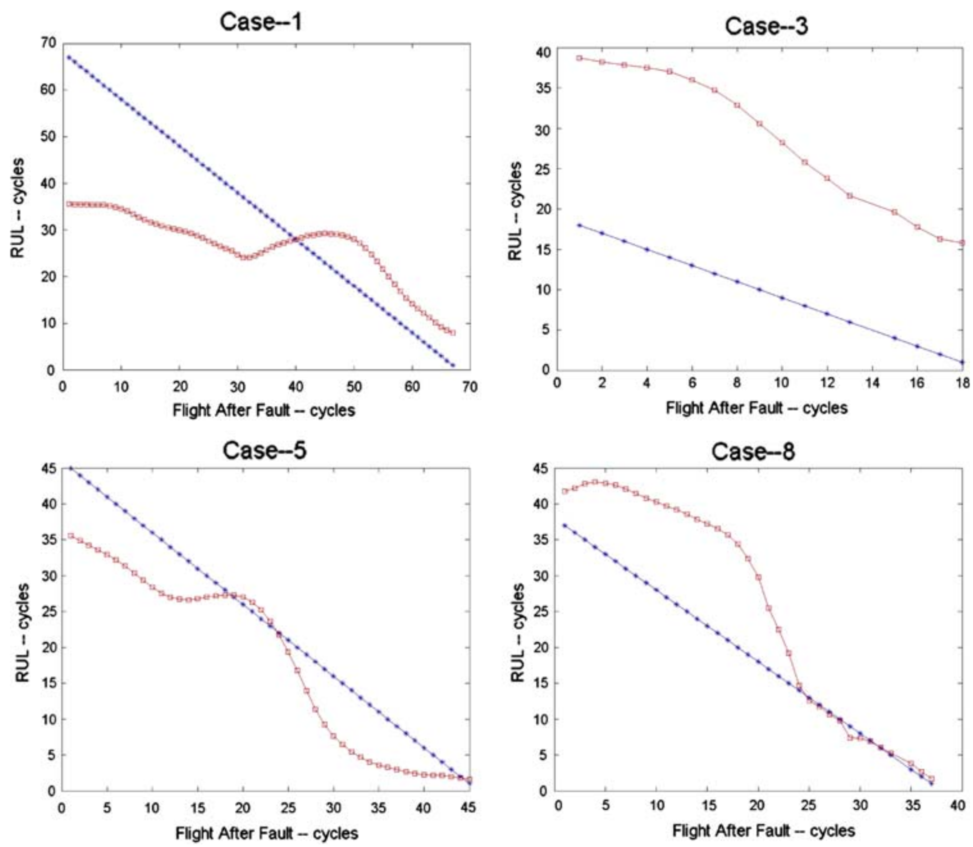


Fig. 4 Extracted features for FIM

Fig. 5 Example cases of RUL estimates with FIM



Once we have these potential features and the corresponding RUL's for these cases. We can use the EAs to train the FIM and automatically select the best fuzzy membership functions for each of these features. Every set of operational parameter features from one flight and the corresponding RUL define an instance.

However, during the training process, one instance can only select peers from other aircraft engine operational trajectories. This mostly replicates the run-time prediction mode, in which we have an instance base and operational parameter trajectory up to current time for a particular aircraft engine and we want to make the RUL estimation for a particular faulted module of this unit.

We ran the EA with a population size of 50 for 30 generations. The details of this particular differential EA can be found in [11] and [12]. In the construction of FIM, we use the weighted average method for aggregation. Figure 5 shows some example cases of the RUL prediction using FIM. The blue stars are the true RUL, while the red circles are the predicted values using FIM. We can see that the prediction tends to improve towards to the end of operation.

Summary and Conclusions

This paper describes an instance-based model for RUL estimates of an aircraft engine module. We proposed the use of local fuzzy models based on clusters of peers-similar instances with comparable operational characteristics and performance. The proposed fuzzy instance-based approach produces very encouraging results in such a high-noise environment. An evolutionary framework is used for tuning and maintening the model. RUL estimates are generated when a fault has been detected based on the instance model. The estimate is updated with operational data from each additional flight, which makes the estimates increasingly updated and more accurate. As noted above, prognostics is dependent on the inputs of diagnostics. As we get accurate inputs from the diagnostics, the peer selection can be improved if the operational data is segmented by failure modes.

References

1. Bonissone, P., Goebel, K.: When will it break? A hybrid soft computing model to predict time-to break margins in paper machines. Proceedings of SPIE 47th Annual Meeting, International Symposium on Optical Science and Technology, vol. #4787, pp. 53–64 (2002)
2. Goebel, K.F., Bonissone, P.P.: Prognostic information fusion for constant load systems. Proceedings of the 7th Annual Conference on Information Fusion (2005)
3. Goebel, K.F., Bonanni, P., Eklund, N.E.: Towards an integrated reasoner for bearings prognostics. Proceedings of 2005 IEEE Aerospace Conference, pp. 1–11 (2005)
4. Goebel, K.F., Eklund, N.E., Bonanni, P.: Fusing competing prediction algorithms for prognostics. Proceedings of 2006 IEEE Aerospace Conference, 11.1004 (2006)
5. Bonissone, P.P., Varma, A.: Predicting the best units within a fleet: prognostic capabilities enabled by peer learning, fuzzy similarity, and evolutionary design process. Proceedings of the FUZZ-IEEE 2005, Reno NV, USA, May 22–25 (2005)
6. Bonissone, P., Varma, A., Aggour, K., Xue, F.: Design of local fuzzy models using evolutionary algorithms. Comput. Stat. Data Anal. **51**, 398–416 (2006)
7. Bonissone, P.: The life cycle of a fuzzy knowledge-based classifier. Proceedings of 2003 North American Fuzzy Information Processing Society. Chicago, IL (2003)
8. Atkeson, C.G.: Memory-based approaches to approximating continuous functions, In: Casdagli, M., Eubank, S. (eds.), Nonlinear Modeling and Forecasting. Addison Wesley, Harlow, UK, pp. 503–521 (1992)
9. Atkeson, C.G., Moore, A., Schaal, S.: Locally weighted learning. Artif. Intell. Rev. **11**(1–5), 11–73 (1997)
10. Bersini, B., Bontempi, G., Birattari, M.: Is readability compatible with accuracy? From neuro-fuzzy to lazy learning. In: Freksa, C. (ed.) Proceedings in Artificial Intelligence 7, Infix/Aka, Berlin, Germany, pp. 10–25 (1998)
11. Stom, R., Price, K.: Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces. Technical Report TR-95–012, International Computer Science Institute, Berkeley (1995)
12. Xue, F., Sanderson, A.C., Graves, R.J.: Pareto-based multi-objective differential evolution. Proceedings of IEEE Congress on Evolutionary Computation, pp. 862–869 (2003)