

Knowledge and Time: Selected Case Studies in Prognostics and Health Management (PHM)

Piero P. Bonissone

General Electric Global Research
Schenectady, NY 12308, USA
bonissone@research.ge.com

Kai Goebel

General Electric Global Research
Schenectady, NY 12308, USA
goebelk@research.ge.com

Naresh Iyer

General Electric Global Research
Schenectady, NY 12308, USA
iyerna@research.ge.com

Abstract

In a companion paper we proposed a framework based on *time* and *knowledge*, to analyze various PHM functions resolved by the application of Soft Computing. In this paper we will analyze representative cases studies, discussing applications in anomaly detection, prognostics, and logistics decision support.

Keywords: Soft Computing Applications, Prognostics and Health Management, PHM, Anomaly Detection, Classification, Prognostics, Logistics, Decision Support Systems.

1 Knowledge and Time: A PHM Framework

In a companion paper [1], we analyzed Prognostics and Health Maintenance (PHM) functions and proposed a decision framework for such functions based on the product space of *time horizon* and *domain knowledge*. A goal of PHM is the sustainment of assets. This includes meeting performance requirements and lifecycle cost targets as well as maximizing utilization and/or readiness. To accomplish this goal we need a close interaction among monitoring, diagnostics, prognostics, optimal decision-making, and maintenance. These are the constituents of the PHM framework outlined in [1]. In this paper we will sample this framework, by discussing selected PHM applications and their associated Soft Computing based solutions.

1.1 PHM Functions: Anomaly Detection, Prognostics, Logistics

We will discuss applications in Anomaly Detection, Prognostics, and Logistics Decision Support.

The goal of anomaly detection is to extract underlying structural information from the data, define normal structures and identify departures from such *normal* structures. Usually this detection is performed using unsupervised learning techniques.

Prognostics aims to produce estimates of Remaining Useful Life (RUL). This can be achieved by estimating damage incurred by the asset or, inversely, by estimating an asset health index. Initially, such index reflects expected deterioration under normal operating conditions. Upon the occurrence of an anomaly or failure, the index exhibits accelerating deterioration, reflecting faster RUL reductions.

Off-board maintenance/repair actions cover more complex offline decisions. They require a decision support system (DSS) performing multi-objective optimizations, exploring Pareto frontiers of corrective actions, and combining them with preference aggregations to generate the best decision tradeoffs.

Figure 1 (adapted from [1]) illustrates the location of these PHM functions in the *Domain Knowledge* and *Time* framework.

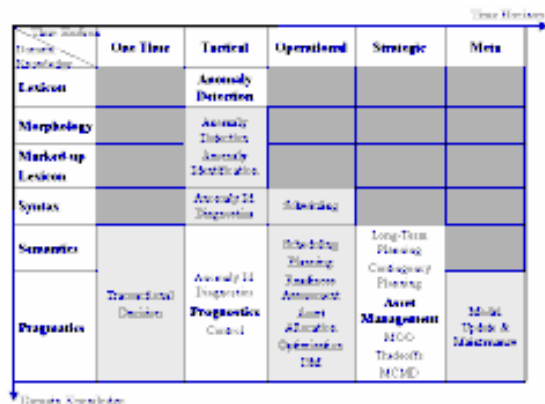


Figure 1: Framework: *Domain Knowledge* x *Time Horizon*

2 Anomaly Detection

2.1 Problem Formulation

The goal is to provide an early warning for anomalous behavior among units in a fleet, operating in dynamic environment. Such early detection will allow us to allocate additional monitoring resources, schedule preventive maintenance, limit downtime, and in general reduce the cost of unexpected maintenance events. When monitoring these units, we want to represent their evolving behaviors, and track them over time. We are interested in detecting the formation of clusters of units' behaviors, and their temporal changes. We want to identify regions of operational normality and detect units that are departing from such regions, or that have become complete anomalies.

While this goal has been attempted by clustering techniques defined over feature spaces [2-3], in our approach we want to do it at the lexical level - by comparing the presence/absence of event messages that characterize each unit's behavior - without resorting to semantics - by extracting features based on domain knowledge.

2.2 SC solution

Our method requires the following steps:

- 1) Represent each flight by a string.
- 2) Create a dissimilarity matrix, containing pairwise distances between strings (flights).
- 3) Project the dissimilarity matrix onto a lower dimensional space, minimizing projection distortion.
- 4) Cluster the projected points, identifying the clusters representing normal operations and the ones representing anomalies.

To illustrate this concept, we will consider a fleet of commercial aircrafts, flying according to their operational schedule. Each aircraft has an *event-log*, in which the aircraft computer systems record time-stamped event messages. Each event could be a routine event (e.g. take-off) or an abnormal event (e.g., a mismatch between two sensor readings). At the end of the flight, the event log has a chronological list of all the events that occurred during the flight. This *event-log* characterizes the behavior of that aircraft during that flight. We can represent each aircraft flight as an object in an event space in which the aircraft's behavior is compared

with the behavior of the same aircraft during previous flights.

Step 1. We create a matrix M , of dimension $[f+1, n]$, where f is the number of different messages recorded over all flights, and n is the total number of flights. Each column represents the count of different messages recorded during that flight. At the end of each column we also record the flight duration. Table 1 shows an example of Matrix M .

FAULT CODES	Flight # 1	Flight # 2	Flight # 3	...	Flight # n
Fault code 1	c(1,1)	c(1,2)	c(1,3)		c(1,n)
Fault code 2					
Fault code f	c(f,1)	c(f,2)	c(f,3)		c(f,n)
Duration	d(1)	d(2)	d(3)		d(n)

Table 1. Matrix M : fault codes over flights

The entry $c(i,j)$ represents the number of occurrences of fault code “ i ”, during flight “ j ”. We normalize the counts by the flight duration, obtaining a ratio of count per hour flight. This will allow us to compare message logs collected over flights of different durations. We will refer to this normalized count as:

$$\hat{c}(i, j) = \frac{c(i, j)}{d(j)}$$

Finally we change these counts into frequencies, so that we have a density for each flight. Thus, the normalized entry $freq(i,j)$ is computed as:

$$freq(i, j) = \frac{\hat{c}(i, j)}{\sum_{i=1}^f \hat{c}(i, j)}$$

At this point we can now represent each flight “ j ” by a string x_j containing f frequencies:

$$x_j = [freq(1,j), freq(2,j), \dots, freq(f,j)]$$

Step 2. For each pair of flights (x_i, x_j) we compute the Normalized Compression Distance $NCD(x_i, x_j)$, which is a surrogate for the Normalized Information Distance based on

$$NCD(x_i, x_j) = \frac{C(x_i, x_j) - \min\{C(x_i), C(x_j)\}}{\max\{C(x_i), C(x_j)\}}$$

Kolmogorov complexity:

where $C(x)$ denotes the length of the compressed string “ x ”, using a standard compressor. Suppose that $C(x) < C(y)$. Then, the metric $NCD(x,y)$ captures the improvement due to compressing string “ y ” using string “ x ” as the previously compressed database (numerator), with compressing string “ y ” from scratch (denominator). This distance has been used to create static classification, affinity groups in

music (showing musical similarities/differences of various composers), linguistic taxonomies (showing the hierarchical grouping of many natural languages), and biological taxonomies (showing the hierarchical grouping of animals based on DNA similarities) [4-6]. In our application we use it to track the behavior of a fleet of aircrafts, whose state changes over time, and to update their similarity-based clustering.

The dissimilarity matrix $D(x_i, x_j) = [NCD(x_i, x_j)]$ is a n by n matrix. Each entry is a normalized distance value in the interval $[0,1]$. When $i=j$, $NCD(x_i, x_i) = 0$. Also, $NCD(x_i, x_j) = NCD(x_j, x_i)$, so matrix D is symmetric, with 0 diagonal.

Step 3. We can visualize the content of the matrix D by projecting it onto a 2-dimensional space. There are many ways to implement this projection, for instance by using a Kruskal's Stress-1 projection that minimizes the overall distortion caused by the projection – this is also referred to as Multi Dimensional Scaling (MDS) - or by using Self-Organizing Maps (SOM's) that map it to a pre-specified granularity. Figure 3 shows the use of Kruskal's Stress-1 approach to perform this projection.

2.3 Results

Step 4. From the figure 2, it is possible to see that the region of normality is centered around the origin, and that larger distances from the origin represents flights that are quite different from normal operations, and therefore are possible anomalies.



Figure 2. 2D projection of NCD distances

By continuously monitoring and tracking new flights, computing the NCD with previous flights, and projecting them onto this 2-dimensional map, we can classify those flights that are potential anomalies. A (normalized)

distance from the origin can be used as the *degree of anomaly* of the flight. This information can then be fused with similar metrics from other classifiers to increase the decision's robustness.

3 Prognostics

3.1 Problem Formulation

The goal of a prognostics reasoner is twofold: To provide an estimate of remaining life for a component or part and to manage the inherent uncertainty of the information sources. The latter is particularly important because the utility of the prognostic estimate depends on the associated uncertainty. This is a critical task because the resulting estimate needs to be within uncertainty bounds that allow for decision-making at a desired risk level. If the uncertainty bounds are very wide, the resulting time-of-failure estimate at the acceptable risk level may be too early to provide any benefit to the decisioning process. That is, there would be no advantage of prognostics compared to a reactionary diagnostics system alone. Uncertainty bounds ideally are tight but need to reflect true output variability.

Generally, there are two fundamentally different approaches can be employed to estimate future damage. One is to model from first principles the physics of the system as well as the fault propagation for given load and speed conditions. Such a model must include detailed knowledge of material properties, thermodynamic behavior, etc. Alternatively, an empirical experience-based model can be employed wherein data from experiments at known conditions and component damage level are used to build a model for fault propagation rate. Such a model relies heavily on a reasonably large set of experiments that sufficiently explores the load and speed space. It can be beneficial to fuse the output of both methods to produce a more accurate result with reduced uncertainty.

3.2 SC solution

Dempster Shafer regression (DSR) [7] provides a prediction of the output in form of a fuzzy belief assignment. This assignment is defined as a collection of fuzzy sets of values with associated masses of belief. The output is computed using a nonparametric, instance-based approach: evidence samples $e_i = (x_i, m_i)$ in the

neighborhood of the input vector x are sources of partial information on the response variable. The evidence samples can be represented by a fuzzy belief assignment $m_y[x, e_i]$. Relevance of the evidence with respect to y is assumed to be dependent on the dissimilarity to y . If x is “close” to x_i according to a given metric $\|\cdot\|$, y is expected to be close to y_i , which makes example e_i quite relevant to predict the value of y . On the contrary, if x and x_i are very dissimilar, example e_i provides only marginal information regarding the value of y . Therefore, neighborhood evidence input elements are discounted as a function of their distance to x . They are then pooled using Dempster’s rule of combination. While the method can cope with heterogeneous training data, the more important characteristics in this context is the formalism for modeling both unreliable and imprecise information provided by multi-sensor systems.

DSR determines the value of sensor measurement y at a given time by discounting the belief mass of each observation by:

$$\phi(|x - x_i|) = \gamma e^{-\frac{(x-x_i)^2}{\Theta^2}}$$

where:

- γ is a tuning parameter (usually ≥ 0.9)
- Θ is a scale parameter, commonly set using cross validation on training data

Next, the discounted belief masses are combined using appropriate version of DS combination.

To enable the prognostic model aggregation, each prognostic model output is assigned a quality assessment that can be interpreted as a subjective confidence. These confidences are computed based on *a priori* performance of the models. That is, the models may be known to have different performance characteristics within different regions of the load-speed mission space. For example, they may produce biases at different damage levels or at different damage rate levels. Moreover, the longer the prediction horizon, the smaller is the prediction correctness. Finally, consistent values should be believed more than values that fluctuate a lot.

3.3 Results

The prognostic reasoner was implemented and tested on a sequential set of experiments that model a simulated, cyclic mission profile. An indent was added to the outer race of a production bearing, which was then run under

those conditions. The bearing was examined several times during the course of the test, and actual spall length was recorded. The test ran to cage failure.

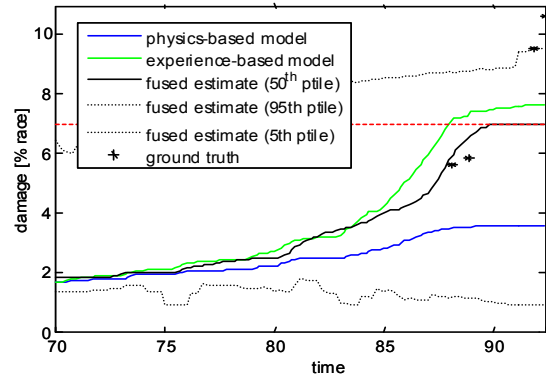


Figure 3 - Prognostic Output at $t = 70$ h [8]

The prognostic mode can be executed at any time after fault detection. Here, it is shown for $t=70$ hours as the starting point (the fault was detected at $t=43$ h) [8]. The fundamental characteristic of the information source confidences is that they drop as a function of time. In addition, there is an *a priori* bias assigned to the different confidences, which reflects the accuracy of the models as observed during testing. Figure 3 shows the output of the prognostic reasoner run forward to $t = 93$ hours where cage failure occurred. The green line reflects the output of the experience-based model in forward mode. The blue line is the output of the physics-based model in forward mode. The stars are the measurements taken during the experiment. Note that they would of course not be available during an actual forward mode. However, for purposes of demonstration, the bearing race was inspected several times and the actual damage is shown to illustrate agreement between real damage and damage estimate. The solid black line is the 50th percentile of the reasoner. The dashed black lines are the 5th and 95th percentiles of the reasoner, respectively. Both experience-based and the physics-based models show an increase of damage over time. At smaller time scales (not shown here), one can see how different operating conditions have varied impact on the forward model. The 95th percentile line crosses the critical damage level of 7% at about 75 hours. Depending on the risk tolerance, this can be used by operators to schedule maintenance or to alter the planned mission sequence - through

correction of the load/speed profile - to achieve longer time to failure.

4 Decision Support for Maintenance

4.1 Problem Formulation

We next describe a decision support system (DSS) for use in operational decision making with PHM-specific data. Challenges arise from the large amount of different information pieces upon which a decision maker has to act. The DSS will enable the user to make optimal decisions based on his expression of rigorous trade-offs through guided evaluation of different optimal decision alternatives under operational boundary conditions using user-specific and interactive collaboration.

4.2 SC solution

The need for a SC solution comes from the complexity of search associated with this decision-making problem. Given the multiple ways in which assets can be allocated to missions and repairs can be performed to assets, the size of the potential set of decisions that can be enacted at any given moment is fairly large, indeed exponential in nature with respect to the number of repair actions. Hence, we propose the use of Multi-Objective Evolutionary Multi-Objective Optimization (EMOO) to perform this search efficiently and generate a Pareto frontier formed by *non-dominated* alternatives.

The use of EMOO is expected to result in the identification of *alternative mission allocations* and *maintenance plans* that are *non-dominated* along PHM-specific objectives (e.g., *overall mission success, safety, maintenance cost*) [9].

Having generated the non-dominated alternatives, and depending upon present and future requirements, the end-user can potentially employ constraint-based approaches or *interactive* tools [10] to select the operational plan that best meets his field requirements to iteratively select a small subset of alternatives. We also foresee the user selecting a portion of a solution (for instance 4 out of 12 tail numbers assigned to a flight schedule) and asking to find alternative selections for the remaining portion of the problem. The EMOO could re-run a reduced version of the problem, with smaller degrees of freedom, to represent a solution that incorporates the user's handpicked selections. If there are feasible non-dominated solutions, the EMOO will generate the corresponding Pareto

Frontier and complete the assignments. This interactive method would also allow the user to employ *what-if* situations permitting him to manually test the robustness of the solution. In addition, an interactive module better addresses many decision-making elements in an operational environment, like interactions between user and system, state representation and awareness, preference elicitation, query answering, and uncertainty representation.

4.3 Results

For a 2-asset, 2-mission, 7 maximum parts per asset problem, with only one available unit per part on the inventory, we find that only 2,135 (of the exhaustive 32,768) allocations (each being a simultaneous assignment of repair actions resources to assets and assets to missions that is logically possible) are both feasible and optimal (4 objectives). Selection from this subset can be a daunting task given that each of the remaining 2,135 plans is feasible and optimal.

In one mode of interaction, the DM expresses desired levels of performance for each of the current missions, such as *maximum time within which to dispatch mission, minimum reliability required to fly mission*, and so on. The DSS looks through the feasible set of allocations to find the ones that satisfy the specified constraints and presents them to the DM. In the absence of any allocation that satisfies the constraints, the DSS additionally tries to find a currently infeasible allocation that could potentially satisfy the expressed constraints. Upon finding such an allocation, it indicates to the decision-maker the infeasible allocation as well as the requirements in terms of additional parts that would make this allocation feasible. Thus the DSS translates currently unattainable goals of the decision-maker into a set of actions for the decision-maker by virtue of which those goals could be attained. If none of the infeasible plans can satisfy the currently expressed constraints, then the DSS indicates that to the decision-maker and communicates the need to weaken some of the constraints.

In an alternate, interactive mode of decision-making, the DM is presented with all of the available feasible allocations in a fashion that allows him to identify the levels of performance that are attainable across multiple missions simultaneously and select from the available set.

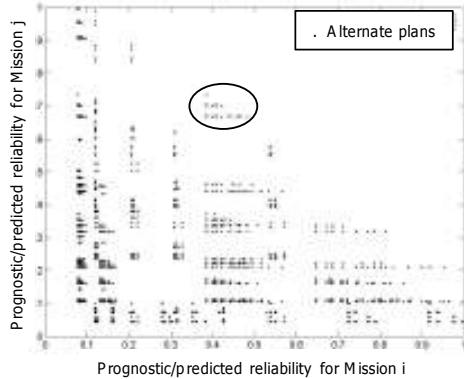


Figure 4 – Interactive decision-making for selection of best repair and mission allocations.

Figure 4 shows an instance of the decision problem, where the decision space is comprised of actions or allocations. In the figure, each point represents a potential *plan* that prescribes the repair actions for an asset in the repair-shop as well as the asset to be allocated to a mission. Each such plan is feasible only if the repair actions are achievable, given the part availability in the shop (for example, if only one unit of part P_b is available in the shop, a plan that allocates P_b to more than one asset is infeasible).

The plot shows the intrinsic trade-offs present in the real-world when trying to satisfy multiple missions (Mission i and Mission j in this case) which compete for the same resources (parts, time, man-power). Figure 4 shows that repair-plans with very high values of *predicted reliability* for a mission i are also plans that result in low *predicted reliability* values for competing mission j in the deck of missions to be satisfied (and vice-versa). Presenting actors in the logistics platforms with such plots confronts them with the need to understand the competing/conflicting nature of the metrics they are trying to simultaneously maximize, and thereby presents them also with the opportunity to locate feasible plans that can potentially optimize along all such metrics simultaneously. For example, the set of plans labeled inside the ellipse represent a subset of plans within which each plan has reasonable values for *predicted reliability* in terms of both competing missions. Such an interaction gives the decision-maker an opportunity to identify what is globally attainable in the space of feasible allocations/plans. By using interactive features like cross-linked plots as in [10], heat-maps and other interactive tools, the user can further explore allocations of interest in the set of

visible plots and use this visual exploration to drive his choice of the best plan to implement.

5 Conclusions

In this paper, we proposed a framework based on time and knowledge to analyze various PHM-related tasks. We sampled this framework with representative tasks (anomaly detection, prognostics, and logistics decision support) and presented cases studies discussing the application of SC techniques to their solution, along with preliminary results.

References

- [1] P. Bonissone (2006). Knowledge and Time: A Framework for Soft Computing Applications in Prognostics and Health Management (PHM), *Proc. IPMU 2006*, Paris, France.
- [2] M. Markou and S. Singh (2003). “Novelty detection: A review - Part 1: Statistical approaches,” *Signal Processing*, vol. 83, no. 12, pp. 2481-2497.
- [3] M. Markou and S. Singh (2003). “Novelty detection: A review - Part 2: Neural network based approaches,” *Signal Processing*, vol. 83, no. 12, pp. 2499-2521.
- [4] R. Cilibrasi, and P. Vitany (2005). “Clustering by Compression”, *IEEE Tran. on Information Theory*, Vol. 51(4), pp. 1523-1545.
- [5] M. Li, X. Chen, B. Ma, P. Vitany (2004). “The Similarity Metric”, *IEEE Tran. on Information Theory*, Vol. 50(12).
- [6] R. Cilibrasi, P. Vitany and R. de Wolf (2004). “Algorithmic Clustering of Music Based on String Compression”, *Computer Music Journal*, 28:4, pp. 49–67.
- [7] S. Petit-Renaud and T. Denoeux (2004). “Nonparametric Regression Analysis of Uncertain and Imprecise Data Using Belief Functions”, *Int'l Jour. Approximate Reasoning*, Vol. 35(1), pp. 1-28, 2004.
- [8] K. Goebel, N. Eklund, and P. Bonanni (2006). “Fusing Competing Prediction Algorithms for Prognostics”, *IEEE Aerospace Conf.*, 11.1004.
- [9] N. Iyer, K. Goebel, and P. Bonissone (2006) “Framework for Post-Prognostic Decision Support”, *IEEE Aerospace Conf.* 11.0903.
- [10] J. R. Josephson, B. Chandrasekaran, M. Carroll, N. Iyer, B. Wasacz, G. Rizzoni, Q. Li, D. A. Erb, “An Architecture for Exploring Large Design Spaces,” *Proc. of the 4th Natl. Conf. of the AAI*, Madison, Wisconsin, pp. 143-150, 1998.