Chapter 7

Algorithms and their Impact on Integrated Vehicle Health Management

Kai Goebel, National Aeronautics and Space Administration

George Vachtsevanos, School of Electrical and Computer Engineering, Georgia Institute of Technology

"Omnes scientiae sunt connexae..." — R. Bacon

7.1 Introduction

This chapter will discuss some of the algorithmic techniques commonly used in Integrated Vehicle Health Management (IVHM) once sensor validation, pre-processing, and feature extraction have been completed (see Chapter 6). Health management observes signals from the system and then reasons (using automated thinking or rules application) over the signals to determine the state of health, possible causes for faults, remaining life, and suitable mitigation strategies. Input into such a health management system is information about the configuration of the system under observation, stimuli from the system, usage history, and anticipated loads. Configuration and past usage information can come from a database. Anticipated usage loads may also come from a manual or automated input / output interface. Stimuli from the system are usually data that have been obtained from sensor measurements. Raw sensor measurements typically do not provide the fidelity of information needed to make health management determinations. Depending on the underlying physics of the sensor used, the sensor data will require some level of treatment to determine key useful information. Before interrogating the data for health information, some pre-processing is usually necessary to remove noise and to

make the signals crisper, and to extract features that can be used by a reasoner, as discussed in more detail in Chapter 6.

In a next step, the features are combined to determine whether the system is functioning normally or whether there are abnormal conditions. Techniques used here combine signal processing and classifiers. The former are used to transform the data into a readily used domain, while the latter are meant to make a yes-no determination about system normality. If the system is in an abnormal state, it is said to exhibit a fault; multi-class classifiers can be used to isolate such a fault. Under certain conditions one can determine how long it will take until the system reaches its end-of-life threshold. This step involves understanding of the damage propagation mode and uses extrapolation techniques. Critical to all steps is an understanding and handling of the inherent uncertainties in the data. An assortment of statistical techniques is available to characterize the quality and potential limits for the available data. Finally, to mitigate the suspected fault–and therefore actively manage the health of the system–one needs to convert the health information into actionable decisions. This can be done by applying the proper optimization method that trades off various objectives such as system safety, minimizing operational cost, and maximizing system performance.

It should be noted that from an operator perspective, making the distinction into the different elements outlined here may not always seem important because the fundamental objective of IVHM is to perform the most efficient process that will support operator safety, mission goals, and affordable operation / availability. Indeed, during the history of vehicle health management, the term “diagnostics,” for example, was sometimes used less stringently to also embrace a number of other activities related to health management, such as trending, abnormal condition detection, or perhaps even fault mitigation. However, from an algorithm development perspective, it is important to be able to distinguish between these elements because the developer

faces different issues and — based on the findings — he/she would possibly choose different algorithmic solutions.

Additionally, some of the algorithmic elements discussed in the following paragraphs are optional. Referring back to Figure 6.2 in the previous chapter, it is possible to skip some of the elements if they are not needed to accomplish the particular health management objective (e.g., maybe it is not important to know the remaining life) or if the cost-benefit for deployment is not favorable.

7.2 Algorithmic Tools and Techniques Used

The following sections describe in some detail the individual elements mentioned in the introduction, in the order shown in Figure 6.2.

7.2.1 Abnormal Condition Detection

The first line of defense in IVHM is often a check whether the system under observation behaves normally or whether there is any abnormal condition. An abnormal condition could be one where the normal operating conditions are exceeded in one or several monitored parameters. Therefore, one has to have an idea what constitutes "normal" and where to set the thresholds defining "abnormal."

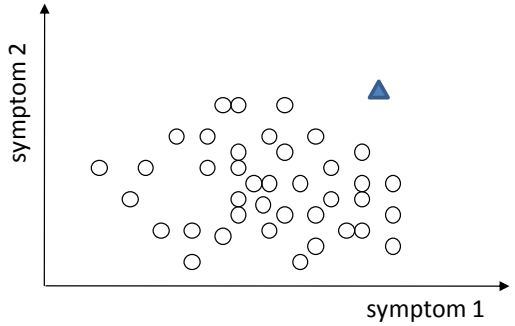

Figure 7.1 Measured symptoms.

Consider the data represented by the circles and the triangle in Figure 7.1. These data can be seen as some measured quantity in the system that varies during normal operation. The fundamental question to answer in abnormal condition detection is whether the triangle should be considered an indication for an abnormal condition. The two symptoms in this figure could represent the exhaust gas temperature and the core speed in a jet engine, for example.

Abnormal condition detection is an important first step in system health management. Abnormal conditions are the first signs of a potential equipment failure at some future time. Detecting abnormal conditions implies, at the very least, detecting change through observations from one or more sensors, ideally in a minimum number of samples after the change has occurred. Having detected that there has been a change, one might then desire to determine the precise nature of the change. This is accomplished using diagnostics (described in the next section). Numerous algorithms can be used for change detection on time series, including statistical approaches (cumulative **Error! Reference source not found.**], sequential probability ratio test **Error! Reference source not found.**], and generalized likelihood ratio test **Error! Reference source not found.**], etc.), signal processing techniques (wavelet **Error! Reference source not found.**]), regression (autoregressive process **Error! Reference source not found.**]), and computational intelligence techniques (neural networks **Error! Reference source not found.**] and fuzzy logic **Error! Reference source not found.Error! Reference source not found.**], etc.). To achieve good performance, relatively large amounts of accurately labeled data are necessary to train many algorithms (particularly computational intelligence approaches, e.g., neural networks). Often this requirement is particularly difficult for real-world problems–data are costly to collect, fault data are sparse, and the labeling is uncertain.

One example of a change detection algorithm is the rank permutation test. This technique transforms features from the raw feature space to a so-called “rank permutation probability

space." In this method, "n" sequential data points are assessed against another "n" points drawn randomly from all the raw data. For example, in Figure 7.2, the ranked order of seven raw data points (the red stars) will be tested against sets of seven ranked points, each drawn randomly from the full population of blue points.

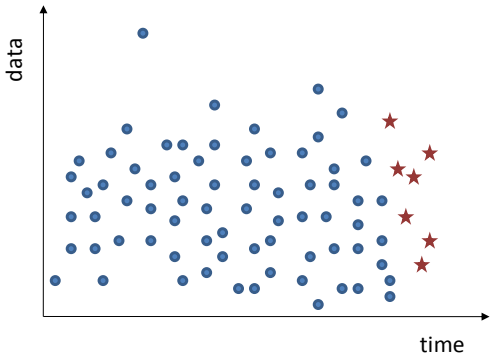

Figure 7.2 Example time series for rank permutation test.

The statistical assessment (here, the sum of ranks) is computed for each comparison as new sets of seven blue dots are randomly redrawn from the set of all points. After the sampling tests have been repeated many times, the results are assembled as a distribution and the test hypothesis is either accepted or rejected. Based on this test for the data in Figure 7.2, there is a, say, 91.7% chance that the red stars should be considered abnormal. The rank permutation method has the advantage that it boosts the classification rate by making events that are statistically improbable more pronounced. It also helps to diminish the effect of noise and outliers.

Other techniques used in abnormal condition detection include 1-class or 2-class classifiers. These are covered in more detail in the section on diagnostics.

7.2.2 Diagnostics

Generally, diagnostics is the process of reasoning over manifestations of effects to determine a possible cause. In the context of IVHM (and noting the statements made in the introduction of this chapter), this becomes the process of determining (or "isolating") fault modes from symptoms.

To accomplish the goals of diagnostics, it is required that the operation of the system be observed using appropriate instrumentation that senses thermal, electrical, mechanical, and fluid characteristics such as temperature, pressure, voltage, displacement, strain, and vibration. Selecting the best instrumentation suite for the systems is described in Chapter 8 (Design). One of the most basic methods is to correlate the magnitude of a sensed sensor signal with a fault condition. If a threshold is surpassed, a fault condition can be declared. Unfortunately, in most situations, such a straightforward assessment does not work because the sensor signal is poorly correlated with the fault, or it may be drowned in transients or operational and environmental conditions. More advanced methods include neural networks, rule-based expert systems, case-based reasoning systems, model-based reasoning systems, learning systems, and probabilistic reasoning systems.

Neural net diagnosis algorithms are almost considered the baseline technique because of their ease of use. Fundamentally, a set of labeled training data is used to "learn" the model of the behavior of the system. A neural net is composed of a parametric interconnected representation of nonlinear functions that allows adaptation of its parameters ("training") such that the desired result can be retrieved (within bounds) based on a particular input stimulus. The stimulus is the observation from the system. The desired result is the diagnostic finding.

In rule-based systems, one would encapsulate quantitative and qualitative expert knowledge about the component or systems. Through logical inferences and constraint analysis, one can arrive at a

set of potential failure candidates. In qualitative model-based methods, one relies on dependency tracking, constraint analysis, and qualitative simulations of the dynamics of system behavior. In model-based methods, abstracted forms of observed behavior are compared to behaviors generated by the quantitative models, and differences are traced.

Case-based reasoning is another data-driven method that is fairly easy to deploy. What is required is a number of validated "cases" that provide solutions to different problems. The problems need to be characterized by a set of measurable observations (such as sensor measurements). A new problem would then be evaluated with regard to its proximity to the different solutions. Ultimately, the correctness of the answer will be confirmed and the observation-solution pair can be added to the database of "cases."

It has to be recognized that there is no one method that performs optimally for all possible application domains (also called the "no-free-lunch theorem"). Challenges for applying diagnostic reasoning technology include determining the best combination of methods for a given system under the constraints of computational resources available, providing information with enough time to act in time-critical situations, the cost of developing the automated system, and the costs of maintaining the automated system over the lifetime of the application.

7.2.3 Prognostics

Prognostics is the science of determining the remaining useful life of a component or subsystem given the current degree of wear or damage, the component's load history, and anticipated load and environmental conditions. A quantification of the degree of a component's wear or damage and the estimate of end-of-life gives decision makers additional knowledge about the health of a system. It provides critical information for risk reduction in go / no-go decisions, cost reduction through the scheduling of maintenance as needed, and improved asset availability. Prognostics

employs technologies that are often based on a detailed analysis of fault modes and modeling of the physics of both the component at hand as well as the mechanisms underlying the fault. For the latter, the idea is to model the damage progression and its dependency on certain accelerators or stressors. Next, algorithms that estimate the remaining life use these physics-based models as well as measurements from the system as input. They then use estimation techniques that propagate the anticipated degradation into the future and provide, as output, the point where the component no longer meets its desired functionality. The anticipated degradation is a function of future load conditions and environmental conditions. For many systems, future load and environmental conditions are very similar to the load and environmental variations seen in the past, but in applications where the future loads and environmental conditions may vary significantly from those of the past, it is desirable to formulate an anticipated load profile, based on knowledge of how conditions may vary, to improve prediction accuracy and precision.

The task of tracking a state variable and predicting future values is often cast as a filtering problem. Generally, quantification and management of different sources of uncertainty — stemming from state assessment, model, measurements, future load, and environmental conditions–provides critical information necessary for users to assess the risk of failure of a component and determine when action needs to be taken. Therefore, algorithms need to be able to receive, process, and output information about uncertainty in the system. Where detailed modeling of the component's physics is not feasible (or to augment the less-important fault modes that are not modeled with detailed physics-based models), the remaining life estimation can also be accomplished through an evaluation of run-to-failure data using machine learning techniques.

Prognostics can be developed for almost any critical component as long as one has either some knowledge about the underlying physics or a sufficient amount of run-to-failure data exist. In the following, the basic elements that are required to perform remaining life estimation will be

explored in more detail. These are knowledge about the system behavior, damage threshold(s), damage propagation model, data, and a propagation algorithm.

Model

Models are meant to encode one's knowledge of the domain into a form that replicates the system's behavior under nominal conditions as well during degraded conditions. Typically, a model is composed of a *structure* and its *parameters*. Physics-based models capture the underlying physical properties of a system or component. An example of a physics-based model is one based on first principles, such as conservation of energy. Sometimes, these physics-based models require the use of simplifying assumptions to keep the problem tractable, e.g., linearization, hierarchy of local models, or the use of default values. Theoretically derived knowledge may then be inconsistent with the real system's behavior. Models can be supplemented with experiential knowledge.

Data-driven models, in contrast, attempt to derive models from any information available from (or usually buried in) historical data that may have been collected. Information is then represented by a collection of instances of relationships among the system variables, which ideally points to causality, but more often just highlights correlation. Purely data-driven methods have other drawbacks, because data tend to be high-dimensional, noisy, incomplete (e.g., databases with empty fields in their records), or wrong (e.g., outliers due to malfunctioning or failing sensors, transmission problems, or erroneous manual data entries). Some techniques, which attempt to address these problems, include feature extraction, filtering and validation gates, imputation models, and virtual sensors that model the recorded data as a function of other variables.

Models can be built to encapsulate the system behavior as well as the propagation of damage.

Damage Threshold

The need for a damage threshold seems straightforward. Of course, one needs to know what condition should terminate the end-of-life prediction. There are numerous cases where the end-of-life condition is difficult to establish or to determine. It would be convenient if the system (or subsystem or component) fails at the end-of-life condition. But a requirement for an end-of-life threshold also needs to be a measurable condition. This is not always the same condition as a catastrophic event. It is also possible that the system may continue to operate beyond the limits of the end-of-life conditions.

Algorithms

The task of the prognostic algorithm is to perform state assessment and to determine the remaining life with the aid of the models. The algorithms must take into consideration future load and environmental conditions, and express the fidelity of the solution using appropriate uncertainty representation. Algorithms that can take advantage of physics-based models include Kalman filters and particle filters. Data-driven algorithms, in contrast, retrieve information from an internal mapping of expected load and environmental conditions. They can also extrapolate on the current trend. Data-driven algorithms include various types of regression algorithms that can come from statistics or the machine-learning domain. They include auto-regressive algorithms, neural nets, relevance vector machines, and others.

7.2.4 Contingency Management

Contingency management is meant to close the loop of integrated vehicle health management by providing the appropriate mitigating action to resolve the issue stemming from the degraded state of health. Depending on the prognostic horizon, different technologies need to be employed for contingency management. This has to do with the inertia of the system and lead-time for certain mitigation actions (see Figure 4.2). If the time-to-failure is in the range of milliseconds, one

needs to generally react fast using adaptive control mechanisms at the machine controller level. If the prognostic horizon is in the second range, appropriate contingencies may involve control reallocation, i.e., the use of other components or subsystems. If the prognostic horizon is longer, one can consider mission re-planning. Finally, if the prognostic horizon is considerably longer, perhaps even extending beyond the duration of the mission at hand, one can integrate the logistics operations and consider various optimal maintenance actions. The latter would typically involve multi-objective optimization.

In the following, we give an example for controller adaptation, and the different individual contingency management actions will be discussed briefly.

Adaptive Controls

The optimization of the controller can be divided into several complementary subtasks. These subtasks include (1) optimization of the actuator gains, (2) optimization of the control modifiers (adjustables), and (3) design and optimization of the control schedules. This task decomposition is necessitated by the fact that local gain modifications often do not result in any significant variation at the global performance level. In addition, the potential for *crosstalk*, i.e., the difficulty to track correlations of several simultaneously manipulated variables on the overall controller, supports the strategy of dividing the optimization endeavor into smaller optimization tasks. Depending on the impact that the particular control variable under consideration has on the overall and local performance criteria, we maximize the observability from an optimization standpoint. This means that for some control variables, only local performance criteria (local tracking errors) are considered while other control variables are considered from a global level (critical margins, tracking of vital parameters, and global tracking error). Figure 7.3 gives an overview of this strategy.

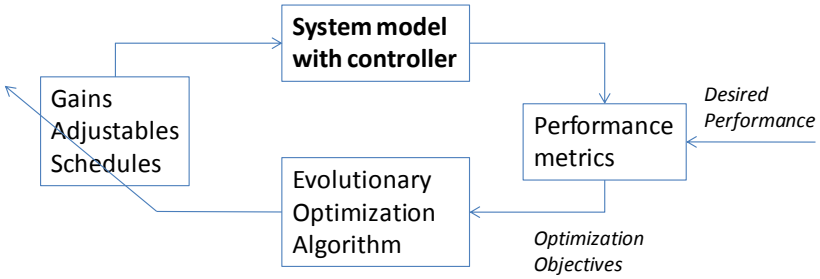

Figure 7.3 Architecture for engine controller reconfiguration

[Subbu, R., K. Goebel, and D. Frederick. 2005. "Evolutionary Design and Optimization of Aircraft Engine Controllers." In *IEEE Transactions on Systems, Man, and Cybernetics; Part C: Applications and Reviews*, IEEE].

The optimization can take advantage of simulators that encapsulate the dynamic behavior of a system (e.g., a jet engine) and its controller with a high degree of fidelity. The user may specify control settings and flight scenarios, and execute the simulator to obtain the engine response given a high-level pilot command, such as demanded fan speed, which is a good measure of thrust.

Optimization of parameters in engine controllers is reported by Chipperfield and Fleming [1996, 1998] and Fonseca and Fleming **Error! Reference source not found.**]. Gremling and Passino **Error! Reference source not found.**] report the design of an online adaptive state estimator for a jet engine compressor whose model is evolved by a genetic algorithm. Subbu et al. **Error! Reference source not found.**] use genetic algorithms for controller design.

Multi-Objective Optimization

When the prognostic horizon is sufficiently large, one can consider various long-term actions that might include considerations of the logistics chain. Challenges arise from the large amount of different information pieces upon which a decision-maker has to act. Consider, as an example, a decision support system (DSS) for use in operational decision-making in the context of running

missions and performing maintenance. The DSS enables the user to make optimal decisions based on his / her expression of rigorous trade-offs through guided evaluation of different optimal decision alternatives under operational boundary conditions using user-specific and interactive collaboration. An Evolutionary Multi-Objective Optimization (EMOO) can perform the search in this space and generates a set of optimal solutions (the "Pareto frontier"). This will result in the identification of alternative mission allocations and maintenance plans that are non-dominated (i.e., optimal) along IVHM-specific objectives (e.g., overall mission success, safety, and maintenance cost) **Error! Reference source not found.**].

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 **Error! Reference source not found.**] to select the operational plan that best meets the field requirements to iteratively select a small subset of alternatives. An interactive method would also allow users to employ what-if situations, permitting them to manually test the robustness of the solution.

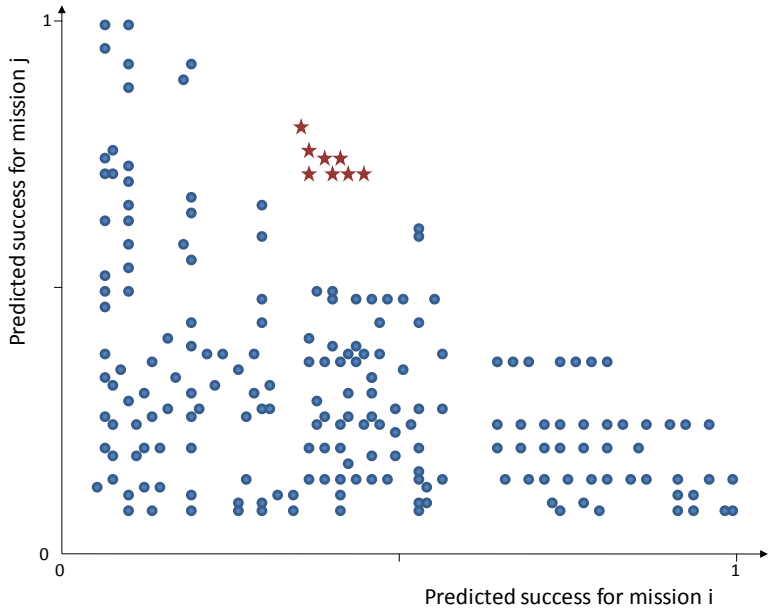

Figure 7.4 Interactive decision-making for selection of best repair and mission allocation [Adapted][Iyer, N., K. Goebel, and P. Bonissone. 2006. "Framework for Post-Prognostic Decision Support." IEEE Aerospace Conference, IEEE].

Figure 7.4 shows an instance of the decision problem, where the decision space is composed 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. 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, and manpower). Figure 7.4 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.

7.3 Closing Thoughts

This chapter discussed some of the algorithmic choices one encounters when designing an IVHM system. While it would be generally desirable to be able to pick a particular set of algorithms for a particular problem, the reality is a bit more complex. Depending on the budget, the performance requirements, the computational constraints, sensor availability, access to historical data, operational and environmental conditions, robustness to changing system configurations, algorithm maintenance needs, etc., no one algorithm will perform best in all situations. Indeed, it is necessary to evaluate these constraints during the algorithm design process and determine the best choice on a case-by-case analysis. The trade-offs between different choices are very real, and sometimes no solution can be found, which means that some of the constraints have to be relaxed. The simplest solution is generally preferred over a more complex one, but it is also important to consider that there is no free lunch. Finally, any health management solution also has to undergo verification and validation (V&V) and, in some cases, certification. Some of these issues are topics of other chapters in this book.

References

Arulampalam, S., S. Maskell, N. J. Gordon, and T. Clapp. 2002. "A Tutorial on Particle Filters for On-line Non-linear/Non-Gaussian Bayesian Tracking." *IEEE Trans. on Signal Processing,* 50(2): 174-188.

Boser, B. E., I. M. Guyon, and V. N. Vapnik. 1992. "A Training Algorithm for Optimal Margin Classifiers. Haussler, D., editor, 5th Annual ACM Workshop on COLT. Pittsburgh, PA: ACM Press, 144-152.

Box, G. E. P. and G. Jenkins. 1976. *Time Series Analysis: Forecasting and Control*. San Francisco, CA. Holden Day.

Chipperfield, A. and P. Fleming. 1996. "Multiobjective Gas Turbine Engine Controller Design Using Genetic Algorithms." *IEEE Transactions on Industrial Electronics*, Vol. 43, No. 5.

Chipperfield, A. J. and P. J. Fleming. 1998. "Evolutionary Design of Gas Turbine Aero-Engine Controllers." In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*.

Drucker, H., C. J. C. Burges, L. Kaufman, A. J. Smola, and V. Vapnik. 1997. "Support Vector Regression Machines." Mozer, M., M. Jordan, and T. Petsche, editors. *Advances in Neural Information Processing Systems.* Cambridge, Mass. MIT Press, 9:155-161.

Eklund, N. and K. Goebel. 2005. "Using Neural Networks and the Rank Permutation Transformation to Detect Abnormal Conditions in Aircraft Engines." *Proceedings of the 2005 IEEE Mid-Summer Workshop on Soft Computing in Industrial Applications*, SMCia/05, pp. 1-5.

Fonseca, C. and P. J. Fleming. 1998. "Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms—Part II: Application Example." *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, Vol. 28, No. 1.

Gordon, N. J., D. J. Salmond, and A. F. Smith. 1993. "Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation." *Radar and Signal Processing, IEE Proceedings F* 140(2):107-113.

Gremling, J. R. and K. M. Passino. 1997. "Genetic Adaptive State Estimation for a Jet Engine Compressor." In *Proceedings of the 12th IEEE International Symposium on Intelligent Control*.

Iyer, N., K. Goebel, and P. Bonissone. 2006. "Framework for Post-Prognostic Decision Support." IEEE Aerospace Conference, 11.0903.

Jazwinski, A. H. 1970. *Stochastic Processes and Filtering Theory*. New York, Academic Press.

Josephson, J. R., B. Chandrasekaran, M. Carroll, N. Iyer, B. Wasacz, G. Rizzoni, Q. Li, and D. A. Erb. 1998. "An Architecture for Exploring Large Design Spaces." *Proc. of the 4th Natl. Conf. of the AAAI*, Madison, Wisconsin, pp. 143-150.

Kozma, R., M. Kitamura, M. Sakuma, and Y. Yokoyam. 1994. "Anomaly Detection by Neural Networks Models and Statistical Timeseries Analysis." *Proceedings of the International Joint Conference on Neural Networks*, Vol. 5, pp. 3207-3210, Orlando, FL.

Kumar, K. and B. Wu. 2001. "Detection of Change Points in time series analysis with fuzzy statistics." International Journal of System Science, Vol. 32(9), pp. 1185-1192, Taylor & Francis, September 2001.

Malladi, D. P. 1999. "A Generalized Shiryayev Sequential Probability Ratio Test for Change Detection and Isolation." IEEE Trans. Automat. Control, Vol. 44, pp. 1522-1534.

Morgenstern, V. M., B. R. Upadhyaya, and M. Benedetti. 1988. "Signal Anomaly Detection Using Modified Cusum Method." *Proceedings of the 27th IEEE Conference on Decision and Control*, Vol. 3, pp. 2340-2341.

Platt, J. C. 1999. "Fast training of support vector machines using sequential minimal optimization." *Advances in Kernel Methods - Support Vector Learning*. (eds.) B. Scholkopf, C. Burges, and A. J. Smola. MIT Press, Cambridge, Massachusetts, chapter 12, pp. 185-208.

Ramirez-Beltran, N. D. and J. A. Montes. 1997. "Neural Networks for On-line Parameter Change Detection in Time Series Model." *Computer & Industrial Engineering*, Vol. 33, pp. 337-340.

Severs, G. C. and R. A. Fliess. 1899. "Cost/Ton Mile for Horses and for Electric Vehicles." *Scientific American*, Vol. 81, No. 4, p. 50.

Sharifzadeh, M., F. Azmoodeh, and C. Shahabi. 2005. "Change Detection in Time Series Data Using Wavelet Footprints." *Advances in Spatial and Temporal Databases*, Vol. 3633, pp. 127-144.

Subbu, R., K. Goebel, and D. Frederick. 2005. "Evolutionary Design and Optimization of Aircraft Engine Controllers." In *IEEE Transactions on Systems, Man, and Cybernetics; Part C: Applications and Reviews*, Vol. 35, No. 4, Nov. 2005, pp. 554-565,
Prognostics.

Tsay, R. S. 1988. "Outliers, Level Shifts, and Variance Changes in Time Series," *J. Forecasting* 7, pp. 1-20.

Vapnik, V. N. 1995. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

Willsky, A. S. and H. L. Jones. 1976. "A Generalized Likelihood Ratio Approach to Detection and Estimation of Jumps in Linear Systems." *IEEE Trans. Automat. Control*, Vol. 21(1), pp. 108-112.