ERC-R-94-082

Final Report

Automated Propulsion Data Screening Demonstration System

Contract No. NAS8-39782
SBA 0474-93-2-00007

Submitted to:
National Aeronautics and Space Administration
George C. Marshall Space Flight Center
Huntsville, AL 35812

for the period:
December 17, 1992–December 16, 1994

By W. Andes Hoyt and Timothy D. Choate; ERC, Incorporated
and
Bruce A. Whitehead, The University of Tennessee Space Institute

May 8, 1995

ERC
INCORPORATED

1940 Elk River Dam Rd.
Post Office Box 417
Tullahoma, TN 37388
615-455-9915 Fax 615-454-2042

Preface

The work reported herein is a continuation of the effort begun as NASA Contract NAS8-39184 at Marshall Space Flight Center. We are grateful to Michael Whitley for valuable suggestions and for facilitating the work in many ways. We thank Darrell Gaddy and Luis Trevino for expert guidance in selecting and interpreting the engine data used for this contract, and Jeff Cornelius for assistance in software installation.

Table of Contents

1. Introduction

A fully-instrumented firing of a propulsion system typically generates a very large quantity of data. In the case of the Space Shuttle Main Engine (SSME), data analysis from ground tests and flights is currently a labor-intensive process. Human experts spend a great deal of time examining the large volume of sensor data generated by each engine firing. These experts look for any anomalies in the data which might indicate engine conditions warranting further investigation. The contract effort was to develop a "first-cut" screening system for application to SSME engine firings that would identify the relatively small volume of data which is unusual or anomalous in some way. With such a system, limited and expensive human resources could focus on this small volume of unusual data for thorough analysis.

The overall project objective was to develop a fully operational Automated Propulsion Data Screening (APDS) system with the capability of detecting significant trends and anomalies in transient and steady-state data. However, the effort limited screening of transient data to ground test data for throttle-down cases typical of the 3-g acceleration, and for engine throttling required to reach the maximum dynamic pressure limits imposed on the Space Shuttle. This APDS is based on neural networks designed to detect anomalies in propulsion system data that are not part of the data used for neural network training.

The delivered system allows engineers to build their own screening sets for application to completed or planned firings of the SSME. ERC developers also built some generic screening sets that NASA engineers could apply immediately to their data analysis efforts.

2. Hypothesis

A liquid-fueled rocket engine, such as the SSME, can be viewed as a system with a set of physical and informational interfaces to other systems. A relatively clean interface is defined if the boundary is drawn around the engine controller and the system it controls (valves, turbopumps, etc.). The external influences which affect the state of the system include the informational interface of commands given to the controller, as well as various physical interfaces such as the fuel inlet, the oxidizer inlet, and the venting and repressurization interfaces to both the fuel and oxidizer tanks.

Under nominal steady-state operating conditions the behavior of the engine is, at least in principle, determined by what transpires at these interfaces. By adding time-series data to the engine system interface, engine operation can be determined for transient operation as well. Given the measurements of what happens at the system interfaces, it would be possible over time to predict the values of all parameters measured within the engine. In other words, to the extent that the SSME is a deterministic system, there would exist some function (Figure 1) which predicts the nominal value of any desired engine parameter. The function f would predict this value from measurements at all interfaces crossing the system boundary and time-history data when needed for transient predictions. The function f might be approximated as either a "white-box" model based on the underlying physics of the SSME, or as a "black-box" model which attempts to approximate the function using empirically-derived relationships between inputs and outputs.

The utility of approximating this function f depends on four factors: (1) whether the behavior of the SSME is indeed sufficiently determined by these external influences, (2) whether adequate measurements of these influences are available, (3) whether the function f can be approximated accurately enough and (4) whether time-series data can be used to predict behavior during transient operation. If these conditions are met, then the approximation of f can serve as a tool for detecting anomalies in the behavior of the SSME. The argument for ERC's approach is straightforward. If an anomaly within the SSME is detectable, it is presumably detectable because a measured parameter within the engine differs from the nominal value of that parameter for the given operating conditions.

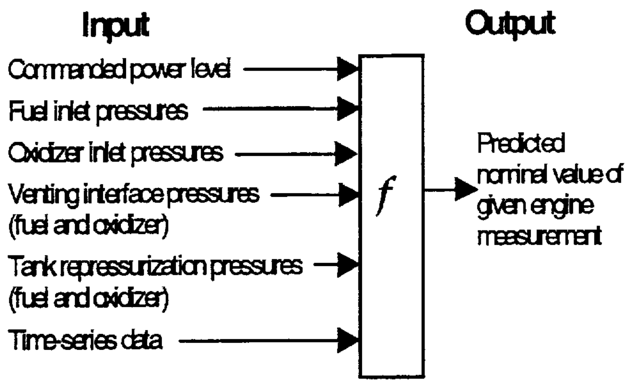

Input
Output
Commanded power level
Fuel inlet pressures
Oxidizer inlet pressures
Venting interface pressures (fuel and oxidizer)
Tank repressurization pressures (fuel and oxidizer)
Time-series data
f
Predicted nominal value of given engine measurement

Figure 1. Function Approximation Hypothesis

Anomaly detection is then based on the idea of making and continuously updating predictions of the expected nominal values of all internal engine parameters which might conceivably indicate an anomaly. Each prediction would be the nominal value of that parameter based on the available measurements of interface conditions affecting the engine. Each predicted nominal value for an engine parameter can then be compared with the actual measured value of the parameter. If the measured value differs significantly from the predicted value, then a potential anomaly is indicated. This process can be repeated for each data point in the time series of measurements to be screened. An automated system could conceivably screen large quantities of a engine data in this manner. The system could flag each potential anomaly for further study by a human analyst. The analyst would then determine whether an anomaly is (1) in the engine itself, (2) in the sensor or data channel, or (3) in the white-box or black-box model of *f*.

Unfortunately, a prediction which is possible in principle might not be accurate in practice. This might happen because of several sources of error:

- The knowledge of the interfaces affecting the SSME may be incomplete.
- The available measurements of conditions at these interfaces may not be complete enough or accurate enough.
- The state of the SSME may be subject to internal fluctuations which cannot be predicted from the external interfaces.
- The approximation of the function may not be good enough.
- The function *f* itself may vary from one engine to another, or among different configurations of the same engine.

Due to any combination of these factors, actual parameter measurements might fluctuate a certain amount from the predicted nominal values even under nominal operating conditions. If the fluctuation can be treated statistically, all is not necessarily lost. In this case, the predictions made by the ERC model of *f* would need to be surrounded by some nominal confidence interval. The utility of predictions made by the model would then depend upon the extent to which fluctuations caused by genuine anomalies are statistically significant in comparison to nominal fluctuations in the measurement. (To the extent that the assumptions of signal detection theory are met, this leads to a well-understood trade-off between "misses," in which genuine anomalies are missed because their fluctuations fall within the confidence interval, and "false alarms," in which nominal fluctuations fall outside the confidence interval and are erroneously flagged as anomalous.) For the purposes of data screening, any measurement deviating from the nominal prediction by more than a set confidence interval would be flagged for further study by a human analyst. The analyst would still have the task of distinguishing nominal from anomalous fluctuations among this reduced set of data falling outside the nominal confidence interval.

The hypothesis of ERC's study is that, despite the many sources of error enumerated above, the function f *can be sufficiently approximated to yield predictions which have practical utility for automated data screening.*

The developers criterion for practical utility was the ability to detect fairly subtle anomalies in SSME data without incurring an inordinate number of false alarms. The type of data screening system described in the Introduction section is not useful unless it significantly reduced the volume of engine data requiring human analysis. If under nominal engine conditions, the deviations of actual measurements from their predicted values is modeled as Gaussian noise, then a nominal confidence interval of three standard deviations above and below the predicted nominal value would perhaps yield an acceptable false alarm rate (on the order of 0.003). In fact, the nominal fluctuations observed in the present study do not fit the Gaussian assumption well enough to yield this low a false alarm rate with confidence intervals of three standard deviations. To compensate for this departure from Gaussian noise, the user can specify the nominal confidence interval, in standard deviations, around the predicted values. That is, only a measurement deviating more than the user-set standard deviations from its predicted nominal value would be flagged as anomalous.

Since neural networks are tools for approximating arbitrary nonlinear functions, the ability of candidate neural network architectures were studied to approximate the function f described herein[1-6]. A neural network does not assume any particular model or functional form for the system it is modeling. Instead, it converges to an approximation of a function based only on the actual data points observed for that function. A neural network is also particularly suitable for fine-tuning the approximation of f to each particular engine configuration. Since a neural network constructs its approximation by being trained on data points, it is conceivable that a neural network can be fine-tuned to adjust its predictions for each of several engine configurations, provided that engine data is available for each configuration.

3. Background

Many investigators have demonstrated that neural networks can learn to discriminate between nominal sensor data and various known classes of faults.[7-14] Since neural networks learn from examples, they avoid the costs of explicitly building and maintaining large, complex knowledge bases. Furthermore, such an example-based training method is likely to be easier to adapt to new SSME designs (or to engines other than the SSME) than a knowledge base which has been built for a specific engine.

Since neural networks are trained by example, however, their reliability depends upon the availability of representative training data for the discriminations to be learned by the network. If the discrimination to be learned is that of nominal versus anomalous data, conventional neural network training procedures require representative data on both sides of the discrimination, (i.e., representative nominal data and representative anomalous data).

In SSME testing (and in propulsion testing in general) it is possible to collect a representative sample of nominal data by systematically varying test conditions over the range of conditions for which engine behavior is of interest. Collecting a representative sample of anomalous data is, however, problematic. In testing any complex system, data is collected for only a very small fraction of all possible anomalies. Other possible anomalies might be simulated, but it is still not possible to anticipate and simulate every possible type of anomaly which might occur. Unfortunately, if a neural network has been trained on an incomplete or unrepresentative set of anomalies, there is no guarantee that it will reliably identify other types of anomalies which might occur.

A neural network, therefore, must be trained to reliably discriminate between two categories (i.e., nominal versus anomalous) when representative training data is available for only one of these categories (i.e., nominal).[15]

Our approach to identifying anomalous behavior, as indicated in the previous section (Hypothesis), was to train the neural network not to classify anomalies, but to predict nominal values of engine parameters. The neural network was to predict, as accurately as possible, the nominal steady-state value of each engine parameter *under the given interface conditions*. Anomalies were identified by comparing predicted nominal data to actual data. In the terminology of our hypothesis: The function f is modeled under nominal conditions. Only nominal training data was needed to achieve this purpose. The neural network was never trained on anomalous data.

The advantage of our approach was that it did not depend on our ability to characterize or gather data for all possible anomalous conditions. A potential anomaly was simply recognized as a greater-than-chance deviation from nominal. In practice, of course, this idea would work only if the neural network's predictions were sufficiently and consistently accurate enough to yield tight confidence intervals. Finding out whether this was the case was the purpose of the reported investigation of neural network architecture for the propulsion system data screening application.

Neural network architecture for multivariate function approximation have been evaluated in the neural network literature and shown to be comparable to classical techniques in the quality of the approximation expected [1-6, 16-17]. In screening large volumes of propulsion sensor data, however, the type of example-based training used in a neural network was likely to be more practical to implement than a classical function-approximation technique. Example-based training also had the advantage of avoiding any particular assumptions about the form of the function to be approximated.

4. Approximation Techniques

We investigated four function-approximation techniques during Phase I, off-contract and current work. They were: (1) a novel neural network architecture based on Gaussian bar basis functions[18, 19] a (2) standard back-propagation neural network [20], (3) a quick propagation neural network [21] and (4) linear regression. Back-propagation is the neural network technique most commonly used in real applications, and thus serves as a benchmark for evaluating other neural network architecture. The Gaussian bar basis function architecture was investigated because of two difficulties in the application of back-propagation to real-world problems: the slowness of the gradient descent optimization when multiple layers are involved, and the possibility of getting stuck on a local minimum. The quick propagation system was used to alleviate sensitivity problems found with the Gaussian Bar architecture as discussed in the Results and Discussion Section below. The quick-propagation ("quickprop") technique is a modified back-propagation system and can also get stuck on a local minimum as with the back-propagation ("backprop"). However, quickprop is less likely to get "stuck" because of its learning algorithm which can "jump ahead." Brief discussions about the architecture of the backprop and Gaussian bar basis functions follow this section.

The backprop and linear regression approximations were conducted off-contract by Dr. Bruce Whitehead as part of his professorial duties at The University of Tennessee Space Institute (our subcontractor for both contractual efforts). The results of his work were presented in a NASA-approved, American Institute of Aeronautics and Astronautics (AIAA) paper [22].

During this contract only the Gaussian bar basis function and quickprop neural network architectures were used as function approximation techniques. Although we experimented with the two other methods mentioned above, we focused on architectures that promised to provide the best choices for a *functional* system to be delivered to NASA that did not require extensive knowledge of neural networks to operate. The quickprop algorithm is the one delivered with the system. The back propagation took too long to train and could get stuck on a local minimum. The linear regression technique was used merely as a benchmark to test the neural network techniques. Results of comparisons of the backprop and Gaussian bar basis functions with the linear regression are contained in the AIAA paper [22].

This report uses the terms *input* and *predicted variables* as they relate to inputs to, and outputs from the function approximation techniques. Each variable is classified as one of the three following types: (1) a measured parameter from an SSME firing, (2) a scheduled input variable defined by the user to model physical events that are not present in the NASA data streams, (e.g., repressurization event), or (3) an input variable that represents the presence of a particular hardware component, (e.g., test stand or controller).

Each of the function-approximation techniques can be used to model a real-valued function of N real variables, which we term *input variables*. The cross product of these N input variables is termed the *input space*. This input space is the domain of the function f to be approximated. The number of input variables is the *dimension* of the input space. The value of the function f to be approximated is termed a *predicted variable*. (The term "output variable" is avoided, since in our application the variables to be predicted are typically measurements of the internal state of the SSME, not outputs per se.) If more than one variable is to be predicted, each is considered as the range of a separate real-valued function f_i.

The function-approximation techniques all assume the availability of a set of sample points (points in the input space) at which the value of the function f is observed. This set of sample points with corresponding observed function values is termed the *training set*, and fitting the function approximation to the training set is called *training*. In our anomaly detection application, the training set consists of nominal data only. The input variables (Figure 1) consist primarily of measurements of conditions at the interfaces we defined for the SSME. These interfaces are the lines supplying fuel and oxidizer to the engine, venting and repressurization interfaces which affect pressure in the fuel and oxidizer tanks, and the commanded main combustion chamber pressure and mixture ratio which the engine controller works to maintain.

In our application, the predicted variable is a measurement of some engine parameter which is known to be nominal in the training data, but which might be either nominal or anomalous in new data to be screened. Even under nominal conditions, measurements of the predicted variable are assumed to be subject to the sources of error and variability described in the Hypothesis (Section 2). All function-approximation techniques studied are based on the idea of minimizing the root-mean-squared (RMS) error of the function approximation over the sample points in the nominal training set.

4.1 Gaussian Bar Basis Function Network

Basis function networks are a family of neural network architecture useful for multivariate function approximation. Such networks attempt to approximate an arbitrary nonlinear function as a linear combination of a set of basis functions [2]. Since the basis functions themselves are nonlinear, this family of architecture is capable of approximating nonlinear functions. Neural network architecture in this family normally has three layers. Each node in the input layer receives input from one variable in the domain of the function to be approximated. The number of input nodes is thus the dimension of the input space. Each node in the middle layer computes the value of one of the chosen basis functions. The number of middle nodes is thus the number of basis functions used for the function approximation.

Various architecture in this family of basis function networks are differentiated primarily by the set of basis functions to be computed by the middle layer of the network. Two such basis functions have been used to build this type of neural network: radial basis functions and Gaussian bar basis. The number of basis functions for the Radial basis functions grows exponentially for input spaces more than a few dimensions. For this reason, a radial basis function architecture was not considered for the present study in which the dimension of the input space greater than 15. To adapt the basis-function technique to input spaces of high dimensionality, a set of semi-local Gaussian bar basis functions has been proposed, but there is only one Gaussian bar for each dimension of the input space. [18,19] Thus there is no particular reason to expect this architecture to be able to approximate an arbitrary nonlinear function and having only one basis function for each dimension appears to form too poor a basis set to give good performance in our application.

We therefore devised a set of basis functions that consists of M, rather than one, Gaussian bars along each dimension of the input data. We got good results with this system during the Phase I contract, but for reasons discussed in the Results and Discussion section below we did not include it as part of the delivered system.

4.2 Quick propagation Neural Network

We adapted the quickprop neural network technique for delivery with the APDS system. [21] This model is a version of backprop which uses a heuristic based on second-order derivative information to speed up the gradient descent training. We employed quickprop because of its speed and robustness.

Quickprop stores a copy of the error derivative, $\partial E/\partial w(t\text{-}1)$, for each weight from the previous training cycle; the difference in the error derivatives between the current and previous training cycle. Then using this stored data and the current error derivative, $\partial E/\partial w(t)$; the current weight is updated according to:

$$\Delta w(t) = \frac{S(t)}{S(t-1) - S(t)} \Delta w(t-1) \qquad (1)$$

where *S(t)* and *S(t-1)* are the current and previous error derivatives. The author makes two assumptions for updating the quickprop weight. First, he assumes "that the error vs. weight curve for each weight can be approximated by a parabola whose arms open upward; and, second, that the change in the slope of the error curve, as seen by each weight, is not affected by all the other weights that are changing at the same time."

This system works well except when the current slope of the error parabola is in the same direction, and the same size or larger than the previous slope. The author then employs a "maximum growth factor" for the weight adjustment. As he suggested we used a value of 1.75 for this maximum growth factor.

The quickprop algorithm also employs a method to update weights that previously had a step size of zero due to a zero slope in the update. However, due to changing other weights the slope size may not now be zero for a weight where it was zero previously. Therefore the algorithm adds the product of a learning rate, ε, and the current slope to the Δw value calculated by the weight update formula, unless the current slope is opposite in sign from the previous slope. If so, then no addition is made.

To avoid getting stuck on a local minimum, quickprop adds a bias of 0.1 to the non-linear function applied at each node. In this case, it is a sigmoid function.

5. Procedure

The following sections discuss the methods used to analyze the function approximation techniques, and to build and test screening sets. One trained neural network, that will approximate a specific PID, and its associated setup information (e.g., engine firings, hardware components, plotting parameters, etc.) constitutes a trained "Screening Set." Unless noted otherwise, the steps below were followed identically for each of the two function approximation techniques described in the Approximation Techniques Section above.

5.1 Selection of SSME Data Sets and Predicted Variables

As explained in Approximation Techniques Section above, our function approximation approach requires nominal training data which adequately represents the engine conditions for which data screening is desired. After training is complete, both nominal and anomalous data are required to test the performance of each technique. We therefore required one or more series of data from SSME firings meeting the following criteria:

1. The engine configurations within each series of data were similar but not identical.
2. Each series contained known nominal data over a range of SSME power levels.
3. Each series also contained one known anomaly which occurred during steady-state or a desired transient category operation.
4. Each series contained an adequate sample of nominal data at the same power or transient level as the data containing the anomaly.
5. The indications of the anomaly in the data stream were subtle enough to be detectable by a trained NASA engineer, but not necessarily obvious to an untrained observer.

Any anomaly in which a given engine measurement goes outside its normal operating range would be easily detectable by many different techniques, and was not considered difficult enough to serve as an adequate test of our technique. We therefore considered only anomalies in which all engine parameters remained within their normal operating ranges, and which appeared anomalous to trained engineers only on the basis of a much more precise understanding of what is deemed nominal *for a given set of operating conditions*.

We needed data from both ground tests (for screening steady-state and throttle transients operations) and flights (for screening steady-state operations only) to test the techniques.

Engine Firing Data Selection

Based on our criteria, NASA engineers selected data from the engine firings, shown in Table 1.

Predicted Variable Selection

The variables predicted were simply those engine measurements to be screened and classified by the system as nominal or anomalous. NASA engineers indicated that the screening data listed in Table 2 would give engineers a good first-cut at test data when utilizing our system. We attempted to build steady-state and transient screening sets for each of these engine parameters (PIDs). In addition to these predicted variables NASA engineers can build their own screening sets for these and additional PIDs using the APDS system.

Since we wanted to build screening sets for the parameters listed above, we chose engine firings so that we could test our screening sets for each desired predicted variable. The sets of data used to build and test screening sets for the parameters of interest to NASA engineers appears in Table 3.

Table 1. SSME Firings Used as Data Sources for Training and Testing

Ground Tests #	Flight #
901-671	STS 46 Atlantis
901-672	STS 47 Endeavor
901-673	STS 50 Columbia
902-444	STS 55 Columbia
902-519	STS 56 Discovery
902-529	STS 57 Endeavor
902-538	
902-540	
902-548	
902-549	
902-550	
902-567	
902-568	
902-571	
902-574	

Table 2. Screening Sets Built for Predicted Variables (PIDs)

PID No.	PID Name
42	FPOV Position
40	OPOV Position
260, 261	HPFP Speed
52	HPFP Discharge Pressure
90	HPOP Discharge Pressure
231, 232	HPFT Discharge Temperatures
59	PBP Discharge Pressure
100	Average Fuel Flow Rate

We further tested, the ability of the two function approximation techniques to detect anomalies without incurring false alarms. Thus, we sought the best technique to build the screening sets. To determine the best technique we focused on the same PID we used to conduct initial system testing in our Phase I contract to predict nominal values of PID 42 for Tests 901-671, 901-672, and 901-673.

Experimentation primarily centered on predicting the time varying nominal values of PID 100 (Average Fuel Flow in GPM) during the 3g throttle-down transients and the maximum dynamic pressure throttle-down transients. The focus occurred since tests with anomalies in the transients that we received from NASA appear in PID 100, among others.

5.2 Selection of Input Variables

Steady-state Input Data Selection

Our choice of input variables was governed by the boundary defined around the SSME. As explained in the Hypothesis section, the external interfaces which cross this boundary include the informational interface of commands given by the controller, and the physical interfaces at the fuel inlet; the oxidizer inlet, and the venting and repressurization interfaces to both the fuel and oxidizer tanks. These types of inputs are depicted in Figure 1.

Table 3. Combinations of SSME Firings and Predicted Variables used to Build Screening Sets

	SSME	Firing No	Time of Onset	Predicted Variable
Screening Set Name	**Input Set**	**Validation Case***	**(sec. from start)**	**PID # and Name**
PID_40		Not Available		40: OPOV Position
PID_42	901-671 901-672* 901-673	901-672	96	42: FPOV Position Steady-state
PID_52		Not Available		52: HPFP Discharge Pressure
PID_59		Not Available		59: PBP Discharge Pressure
PID_90		Not Available		90: HPOP Discharge Pressure
PID_100	902-511 902-519 902-527 902-540	902-521	275	100: Average Fuel Flow Rate 3-g throttle transient
PID_221	902-548 902-549 902-550	902-541	120	221: POGO Pressure Change Steady-state
PID_231_232		Not Available		231, 232: HPFT Discharge Temps.
PID_260_261		Not Available		260, 261: HPFP Speed

* Due to funding limitations we were not able to fully validate the PIDs with which we were supplied anomalies. The PIDs where the validation case is marked not available means that due to time constraints we were not able to get anomalous cases with sufficient nominal training data from NASA to build the function approximations needed for screening the data.

Input variables were selected to capture information about physical conditions at these interfaces which might affect nominal operating conditions within the engine. Based on the advice of NASA engineers who analyze SSME test data on a routine basis, the input variables shown in Table 4 gave the best results for screening. Table 4 also contains the corresponding MSIDs used for flight data screenings.

The first of these variables, Commanded Main Combustion Chamber Pressure, is the desired chamber pressure sought by the engine controller. The remaining variables are pressure measurements at various physical interfaces to the fuel and oxidizer tanks as indicated in Table 4. We extracted data for these engine measurements from existing NASA databases in which measurements are sampled at 25 Hz in the case of the commanded chamber pressure, and at 50 Hz for all other parameters shown in Table 4.

The function f to be approximated might conceivably vary among different configurations of the same engine. To allow for this, the input variables also included discrete variables to represent the different combinations of hardware components of each engine configuration fired. These configuration variables were provided as inputs to allow each function approximation technique to bias its prediction on the basis of the specific components installed during each SSME test. We also allow the user to build input up to two additional data streams to model facility repressurization events (see the Scheduled Input Variables section below).

Selection of Inputs Used for Temporal Indicators

Function approximations for data during transient operation due to power level changes of the engines were also built for ground test data. Table 5 shows the categories of transients screened. We defined these categories for power-level changes based on the magnitude, direction, and slope of the change. To approximate engine parameters during transient operations we built temporal indicators using digital filters applied to time series of the same inputs as in the steady-state function approximations. In other words, to predict the nominal behavior of an engine parameter at time x, we used inputs from times x, $x-\Delta_1$, $x-\Delta_2$, . . . $x-\Delta_n$, where Δ is the spacing interval and n is the number of time steps back that defines the size of the filter. The Digital Filters and Derived Temporal Indicators for Transient Data Section below discusses the construction of the derived temporal indicators.

Table 4. Input Variables

SSME PID*	STS MSID*	Measurement Description	Units
8		Commanded Mixture Ratio	n/a
287		Commanded Main Combustion Chamber Pressure	PSI
819		Engine Fuel Inlet Pressure #2	PSI
821		Engine Fuel Inlet Pressure #1	PSI
827		Engine Fuel Inlet Pressure #3	PSI
830		Fuel Bleed Interface Pressure	PSI
835		Fuel Pressurization Interface Pressure	PSI
836		Fuel Pressurization Venturi Inlet Pressure	PSI
837		Fuel Pressurization Venturi Delta Pressure	PSI
858		Engine Oxidizer Inlet Pressure #2	PSI
859		Engine Oxidizer Inlet Pressure #1	PSI
860		Engine Oxidizer Inlet Pressure #3	PSI
878		Heat Exchanger Interface Pressure	PSI
881		Heat Exchanger Venturi Inlet Pressure	PSI
883		Heat Exchanger Venturi Delta Pressure	PSI

* Parameter Identification Number, Main Propulsion System Identification Number

Table 5. Categories of Transient Engine Operation Included in the APDS

Transient Categories Filter Names	Description
kernel.trans.dn.0_15.slw.#.vshort*	4 data points for bottom of max-q throttle down*
kernel.trans.dn.0_15.slw.#.short	16 data points for bottom of max-q throttle down
kernel.trans.dn.0_15.slw.#.med	64 data points for bottom of max-q throttle down
kernel.trans.dn.0_15.slw.#.long	256 data points for bottom of max-q throttle down
kernel.trans.dn.25_45.slw.#.vshort*	4 data points for 3-g throttle down*
kernel.trans.dn.25_45.slw.#.short	16 data points for 3-g-q throttle down
kernel.trans.dn.25_45.slw.#.long	64 data points for 3-g-q throttle down
kernel.trans.dn.25_45.slw.#.med	256 data points for 3-g-q throttle down

* Filters built, but not used in current system.

5.3 Scheduled Input Variables

The APDS system allows input of "scheduled variables" for facility repressurization events during engine operation that act as independent variables (inputs) and need inclusion in the input data set to properly model predicted variable behavior. These type of variables do not exist as PIDs in the data files. The scheduled variables use a lookup table to get their value according to a manually entered schedule. Allowing users to build scheduled variables into the input data set allows information that is not in the PID data to be included in the input data set.

5.4 Time-averaging of Engine Measurements

For steady-state predictions, all data from steady-state engine operations were time-averaged over a sliding window of 0.2 seconds. We averaged data to improve the signal-to-noise ratio of the data. Admittedly, this time-averaging would reduce the ability of the system to detect subtle anomalies of very short duration.

To improve the signal-to-noise ratio of steady-state predicted variable data "flat" digital filters are used to average data over a sliding window to be consistent with the software design that handles transient data. Several sizes of sliding window are available (1, 5, 10, 25, 50, 100 data points).Steady state data smoothing used for the input variable is fixed at 0.2 seconds.

5.5 Digital Filters and Derived Temporal Indicators for Transient Data

To screen data during the transient operation of the engines we employed digital filters to derive temporal indicators from raw time-series data to reflect, not just the instantaneous state measured by a sensor, but its recent history over the time scale associated with a transient. The filters extract features (the temporal indicators) from the transient times series data. The temporal indicators are used as input and predicted variables for training with nominal transient data. To generate indicators with the most predictive value, the digital filter for each PID is optimized to predict future values of that PID with a given predictive lead time in front of a sliding window of data being convolved with the filter.

The APDS system applies digital filters to time series of PID values to produce more useful temporal indicators for input to neural network training and screening processes. For each transient category (Table 5) present in the training data, extraction and categorization of transients occurs across all engine firing data. A custom neural network is then trained for each transient category. During screening, the appropriate networks are automatically selected by the software to screen transient segments. This approach works well for defined transients.

For each independent PID whose nominal state is being predicted, three different temporal indicators were derived to yield predictions on three different time scales. To detect anomalies on a relatively long time scale, the neural network predicts nominal values for a temporal indicator based on a 10.4-second history of the PID being predicted. To detect anomalies on a medium time scale, a separate temporal indicator based on a 2.56-second history is derived. Finally, to detect anomalies of very short duration, the third temporal indicator is based on a 0.64-second history of the PID in question. These temporal indicators employ digital filters of length 256, 64, and 16 data points respectively (see Table 5 and the section on Digital Filter Size below).

The values of these derived temporal indicators predicted by the neural network, compared to the measured values, are then analyzed to determine if a specified parameter stays within a calculated error band. If a parameter's derived indicator falls outside the nominal prediction band then a possible anomaly may exist. In some cases the derived temporal indicators are actual predictions of PID values as seen in our Phase I steady-state predictions of engine valve positions. In other cases the derived temporal indicators are transformed to more clearly show the temporal behavior of a given PID value. The APDS method of detecting anomalies is based upon first predicting the nominal value of the derived temporal indicators for each PID as a function of the time since onset of the transient.

The APDS system supports the extraction and grouping of transient sensor values for use in training the digital filters. The system selects and applies these filters during the creation of neural network training data. Unlike steady state screening sets, users cannot construct additional filters, they can only use those delivered with the APDS system.

A complete set of derived temporal indicators was trained for 3g throttle-down transients in all PIDs used as input variables as shown in Table 4, for SSME ground tests: 444, 519, 527, 529, 540, and 574.

Digital Filter Methodology

The application of a digital filter to a time series of data is basically a convolution. The two things being convolved are a long time series (the raw data) and a short time series. The short time series is the "kernel" of the filter consisting of floating point numbers. The filtering software represents this sequence of floating point numbers as a vector. The convolution can be thought of in terms of sliding a window (the kernel) over the raw data.

For steady-state data, the system uses a flat window. Convolving with a flat window (e.g., all components of the kernel are the same) is equivalent to doing a running average.

These kernels are stored on disk in the form of persistent vectors. Numerical Recipe routines which perform singular value decomposition and back substitution were used to construct the. least-squares optimal linear predictive filter for each sensor.

Filter Size

Temporal indicators were constructed by taking shifted differences of the kernels obtained from predictive digital filters of different lengths. We standardized on a series of filter sizes of integral powers of four in order to insure that we developed a general method that is not tuned too closely to one particular type of anomaly.

Digital filters of large order (e.g., 250) are desirable because they yield good predictive performance. Unfortunately, 250 inputs produced too many degrees of freedom to be fit with the amount of nominal transient training data made available to us. With this many degrees of freedom, there was a risk that the filter would "overfit" the training data and not generalize well to other comparable data sets. The output of the filters trained did, in fact, appear to overfit the data.

Singular-value decomposition was used to reduce the number of degrees of freedom of the filter without reducing its order. For both the input variables and the predicted variable, the digital filters were constructed using a threshold value of 0.0001 for singular values. This resulting temporal indicators fit the trend in a 3g throttle-down transient data well, without overfitting the noise in the data. This was done to eliminate linear combinations of independent variables which are of small incremental predictive value. Each linear combination thus eliminated from the digital filter reduces by one the number of degrees of freedom of the filter to be trained, and consequently can be expected to counteract any tendency of the filter to overfit the training data.

For the predicted variable whose nominal state is modeled, the system derives three different temporal indicators to yield predictions on three different time scales. To detect anomalies on a relatively long time scale, the neural network predicts nominal values for a temporal indicator based on a 10.4-second history of the predicted variable. To detect anomalies on a medium time scale, a separate temporal indicator based on a 2.56-second history is derived. Finally, to detect anomalies of very short duration, the third temporal indicator is based on a 0.64-second history. These temporal indicators employ digital filters of the lengths 256, 64, and 16, respectively.

Construction of Derived Temporal Indicators

The derived temporal indicators for input variables used to make the prediction employ digital filters of a length of 256 data points (fit from a sliding window of 10.4 seconds of data over the transient) in order to base nominal predictions on 10.4 seconds of parameter history.

Temporal indicators based on a long (e.g. 10.4 second) history emphasize trends in the data occurring at this or longer time scales. Temporal indicators based on a short (e.g. 0.64-second) history emphasize events in the data occurring on this time scale. However, since the filter is trained to predict the value of the PID on an absolute scale (i.e., in engineering units), short-term events are visualized by the filter superimposed on the long-term trend necessary to bring the filter's prediction up to the actual measured value. (The converse is not true since short-term events are smoothed out by long-duration filters and so are not important factors in nominal predictions based on a longer history of data.)

We found that to isolate events occurring on a short time scale, long-term trend information must be subtracted out of the output of a short-term filter. In principle, this could be done by subtracting the time-series output of a longer-duration filter from the corresponding output of a shorter-duration filter, then plotting the difference between these two outputs to visualize short-term events with long-term trend information removed. However, the difference between the output of the two convolutions would be mathematically the same as a single convolution with the difference of the two filter kernels.

The desired result was achieved by the following method: (i) Align the two filters on their leading edges. (ii) Pad the shorter filter with zeroes on the lagging edge to make the two kernels the same length. (iii) Take the pointwise difference between the two kernels. (iv) Use the resulting kernel to digitally filter the data.

Neural network predictions of derived temporal indicators which employ the refined filter appear in Figure 2. It is important to note that we did not simply create a narrow bandpass filter at a frequency specific to this anomaly, which would yield great results but could not be expected to generalize to other types of anomalies. Instead, the results used the system's standard powers-of-four filter lengths of 16, 64, and 256 (at a sample rate of 25 Hz). We standardized the filter lengths used by the APDS system to powers of four, long before the ERC project team had any knowledge of the characteristics of the predicted variable PID used for testing (PID 100). The general technique under test therefore used these standard filter lengths but implemented the shorter two as difference kernels, so that for each PID of interest in transient data the neural network predicts three derived temporal indicators:

1. The output of convolution with the filter kernel of length 256, to visualize long-term trends.
2. The output of convolution with the difference between the kernels of length 64 and 256, to visualize medium-time-scale events with long-term trends de-emphasized.
3. The output convolution with the difference between kernels of length 16 and 64, to visualize short-time-scale events with both medium-time-scale events and long-term trends de-emphasized.

To yield a fair test we did not adjust the filter length to the observed frequency of any specific anomaly. In essence, the information about transient behavior of a PID is captured by three temporal indicators (listed as 1, 2, and 3 above) which predict the trend in the PID and medium-term and short-term variations from this trend, respectively. The appropriate engineering units for predictions 2 and 3 are those of band-limited power. Hence, the predicted and observed mean-squared outputs of the filter are the relevant quantities for constructing nominal confidence intervals.

Subtracting one kernel from another is roughly the equivalent to band-limiting the output of the shorter kernel. It is important to note, however, that the resulting digital filter, while band-limited, is not simply a flat bandpass filter (i.e., a filter whose frequency response is a rectangle with rounded edges) such as would result from a standard Fourier transform method. The least-mean-squared fitting of the original filters to the data using singular value decomposition selectively emphasizes those frequencies which are of the greatest predictive value, and thus can be expected to produce a digital filter with a frequency response as varied as necessary to optimally fit the data. Such a filter could of course be hand-crafted for each predicted variable. We used singular value decomposition to produce digital filters useful for prediction, individually tailored to each predicted variable, with less human intervention than digital filters produced by manually specifying the variable frequency response desired for each predicted variable.

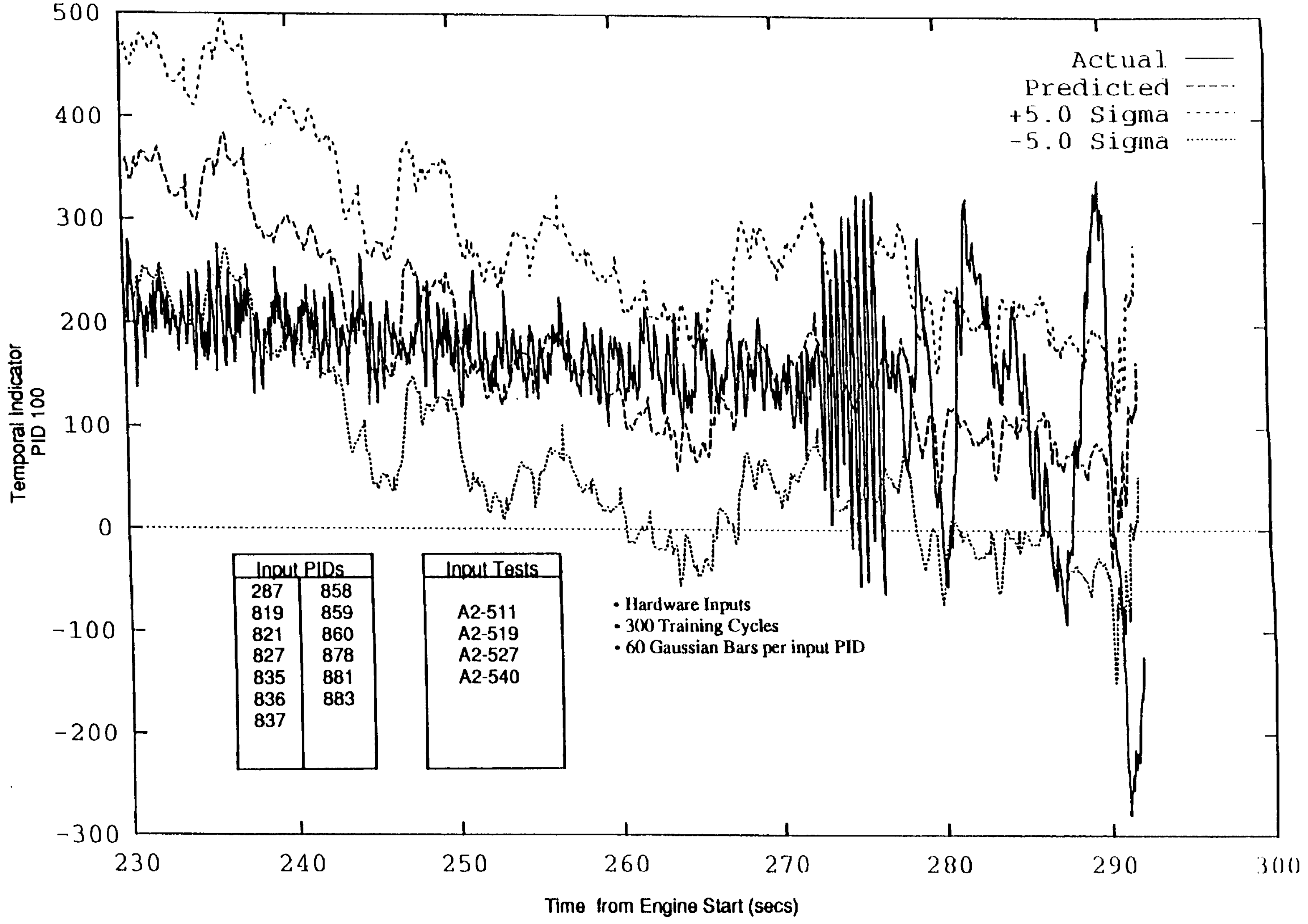


Figure 2. Screening Results of Nominal Data for Engine Firing 902-519,during a 3-g Throttle Down Transient.

Digital filters were constructed to minimize the mean squared prediction error, using the method of singular value decomposition. A major motivation for using singular value decomposition was that linear combinations of independent variables which are of small incremental predictive value can be detected and eliminated. This was done in the filters we constructed. We experimented with singular threshold values of .1, .01, .001, .0001, .00001, and .000001.

The filter obtained using a threshold of .001 yielded more temporal trend information than the value of .01. In the case of PID 827, no further refinement was obtained for lower threshold values, but for the other PIDs, the result from a threshold value of .0001 yielded a better fit to the trends without overfitting noise.

Based on this study, we selected a singular value threshold of .0001 to construct the complete set of filter kernels delivered with the system.

Transient Types

The system screens the data for engine operations during transients as shown in Table 6.

Table 6. Transient Engine Operation Screened by the APDS.

Transient Category	Screening Module
Repressurization Events	Steady State
Facility Venting	Steady State
3g Throttle Transients	Transient
Maximum Dynamic Pressure Throttle Transients	Transient

Repressurization Events

The steady-state training algorithms adequately handle and build screening sets for repressurization events. The system builds flags that model the repress events as shown in Table 7. The user only needs to input the time of the repress event if it occurs.

Table 7. Repress Event Flags Used in the APDS System.

Repress. Event Type	Flag Value
Max->Min Fuel Repress.	1
No Fuel Repress.	0
Min.->Max Fuel Repress.	-1
Max->Min GOX Repress.	1
No GOX Repress.	0
Min.->Max GOX Repress.	-1

Facility Venting

Transients due to fuel and oxidizer tank facility venting are represented by data that are acquired from fuel and oxidizer fuel inlet pressures and are not excluded from steady-state screening.

3-g Throttle Transients

We successfully modeled PID 42 during 3g throttle transients under nominal conditions. The neural network's ability to predict 3g throttle-down transient values of PID 42 from PIDs 819, 821, 827, 835, and 858 was investigated by training a neural network on SSME ground tests 444, 519, 527, 529, 540, and 574. This completed the set of temporal indicators which could be meaningfully trained from the data available.

The predictions were observed to be much better near the middle of the transients than at the endpoints. The algorithms which led to these results were analyzed. We observed that, when using the Gaussian bar, changing the placement algorithm could improve the prediction at the transient endpoints.

Maximum Dynamic Pressure Throttle Transients

We also analyzed the bistability anomaly present during engine operations for the maximum dynamic pressure throttle down. Derived indicators showed differences in the characteristics of the data due to the bistability problem (see the Results Section below).

5.6 Differentiation of Steady-State and Transient Engine Data

The current screening system builds separate function approximations, for the same predicted PID, for steady-state and transient operation. Transients due to SSME power-level changes, fuel and oxidizer tank facility venting, and fuel and oxidizer repressurization were present in the raw data. We automated the software to recognize transients due to SSME power-level changes by examining PID 287, commanded power level. An algorithm examines the commanded power-levels present in the data stream. All constant power levels are treated as steady-state data segments. Data for seven seconds after a transient completes (according to this threshold) is not treated as steady-state. Transients due to fuel and oxidizer tank facility venting are represented by data that is acquired from fuel and oxidizer fuel inlet pressures and are not excluded from steady-state screening. Transients due to fuel and oxidizer repressurization are not excluded. For these events the user inputs "scheduled variables" (see the Scheduled Input Variables Section). In the case of engine startup and cutoff our algorithm sees these events as transients and ignores this data for steady state screening and transient screening.

5.7 Automated Categorization and Extraction of Transient Data

The module discussed in the previous section automatically identifies transient data segments for the user. This module performs the following functions:

1. Identifies steady state and transient data segments within a test.
2. Categorizes power-level transients according to the magnitude, sign of the slope, and rate of change of the data.
3. Extracts data for the relevant PIDs during transients into appropriately named binary files. These binary files are used as inputs for neural network training.

5.8 Normalization Enhancements

The APDS software normalizes neural network training data to equalize the a priori influence of different variables measured on different scales, as well as SSME hardware represented on a binary scale. All training variables are normalized to have the same variance. This process assures that the squared weights assigned to each input variable by training will accurately represent the relative influence of that variable.

5.9 Random Sampling of Nominal Training Data

As explained in the Hypothesis section above, only nominal data was used for training ("fitting") each function approximation. All steady-state data segments identified by NASA engineers that contained anomalies (Table 3) were excluded from any training sets.

For steady-state predictions, the actual training data was created by randomly sampling the available data points. This was done for two reasons. First, training with the complete data sets would have been computationally prohibitive. The second reason was that, after training was completed, it would be possible to test the response of each function approximation technique on a different sample of data than was used in training. Since data points useful for transient prediction were relatively scare, all data points were used.

For steady-state data screening, the software allocates training data among different power levels to give each power level equal representation in the sample. The user inputs the number of data samples to train a screening set and the software equalizes the number of data points for each power level. This was necessary because the criterion to be optimized by both neural network techniques was mean-squared error over the training sample. Any significant inequality in the representation at different power levels would translate into unequal weighting for these power levels in the mean-squared error criterion, allowing a heavily-represented power level to be approximated closely at the expense of a lightly-represented power level.

5.10 Scaling of Input Variables

For the quickprop algorithm it is desirable to scale all input variables to the same dimensionless scale so that the weights resulting from training reflect the actual influence of each variable, rather than an artifact of the scale on which that variable is measured. Each input variable was therefore scaled to have a mean of zero and a standard deviation of 0.25 over the training set. (The same scale was subsequently used for data to be screened.) The data was scaled to a standard deviation of 0.25 so that all training data within four standard deviations of the mean would fall within the interval (-1, 1) which was desirable for good performance of the quickprop algorithm.

In the case of the Gaussian bar basis function technique, basis function centers were automatically spaced over the range of each input variable where data is meaningful (e.g., during steady-state operations power levels may only be at 100% and 104% for the entire engine firing) as described in the Function Approximation section above. However, the spacing is always distributed in one-half overlaps of the Gaussian bar basis functions. The units of measurement for these input variables are therefore irrelevant to the definition of the basis functions, and to the activation and training of the basis function network.

5.11 Scaling of Predicted Variables

The variables to be predicted were biased to have a mean of approximately zero over the training set. Since the approximation techniques contain adjustable bias terms, this constant bias was not strictly necessary but it served to improve the speed of convergence of iterative training.

In the case of quickprop, the sigmoidal output units are capable of producing outputs only within the range (-0.5, 0.5) and can only approach the endpoints of this range asymptotically. For quickprop training, therefore, each variable to be predicted was scaled to a range of (-0.25, 0.25) so it would fall well within the sigmoidal output range. All prediction errors computed by the quickprop algorithm were then scaled back to the original engineering units of the predicted variables for comparison with the prediction errors of the other two techniques.

5.12 Parameters and Training Procedures for the Approximation Techniques

The approximation techniques used sets of input variables, consisting of the 15 engine measurements shown in Table 4 and the discrete engine hardware component variables described in Selection of Inputs section.

The neural network technique was trained at the highest learning rate (rate of gradient descent) for which convergence to a global minimum was reliably achieved in both prediction tasks by all network configurations tested. The normal rate for the quickprop was set to 0.1 when it was not using the second order heuristic technique to change the weights. A maximum weight change of 1.75 was used.

5.13 Statistical Estimation of Nominal Approximation Error

As explained earlier, the detection of anomalies depends upon surrounding the predictions made by each function approximation technique with a nominal confidence interval. The user can input the confidence interval size (in standard deviations). To achieve an acceptably low false alarm rate, the confidence interval was usually set to three, four or five above and below the predicted value. To calculate the accuracy of the trained network and to build the confidence interval, the trained neural network was then used to predict the nominal values of the input data set using independent samples of nominal data. Then a nominal standard deviation between the actual and predicted nominal data was calculated as the RMS prediction error over this independent nominal data. This nominal standard deviation was then used in the screening procedure described in Screening Procedure and Visualization of Results Section below.

5.14 Sampling of Data to be Screened

To test the ability of each function approximation technique to distinguish nominal from anomalous data, we constructed data sets containing both nominal and anomalous data to be screened. We used all steady-state data from the engine firing when screening. Each screening data set contained the same input variables used in training.

5.15 Screening Procedure and Visualization of Results

Each trained neural network was applied to each screening data set to generate a time series of expected nominal values for the predicted variables shown in Table 3. The measured value at each time was compared to the expected nominal value predicted by the given neural network. Each prediction was surrounded by a confidence interval of five standard deviations above and below the nominal prediction, as calculated according to the Statistical Estimation of Nominal Approximation Error section above. Measurements falling within the user-input confidence interval are classified as nominal; those falling outside the confidence interval are classified as anomalous..

This screening is portrayed graphically in which the measured values are shown as a solid line and the prediction by a solid line. For clarity, the confidence interval band is shown on a separate plot that also contains the value of the error between the measured and predicted value of the PID.

5.16 Screening Types

The delivered demonstration system screens the following types of engine operation data:

- Steady State Screening
  - Ground Test
  - Main Propulsion System
- Transient Screening: Ground Test only
  - 3g-throttle down operation
  - Maximum dynamic pressure throttle down operation

The APDS software supports the definition, training, and execution of screening sets for Main Propulsion System (MPS) engine firing flight data. A single software interface supports both modes of operation and performs the necessary data extraction and preprocessing with minimal user input. Based upon user inputs, data is extracted and synchronized from up to five NASA databases, controller and Orbiter data, in order to create the neural network training data. Flight data screening is limited to steady-state engine operation, but in theory could be applied to transient engine operation during in flight as well.

6 Delivered System

The APDS System as delivered is an automated system capable of the following:

1. Interfacing to NASA engine firing databases
1. Steady state screening
2. Steady-state screening set construction
3. Transient screening for ground-test throttle-down transients
4. Automated extraction of data segments by steady-state or transient engine operation.
5. Plotting of screening results
6. On-line hypertext help

More detailed descriptions of the APDS system as delivered are contained in the "APDS User's Guide" [23] and the "APDS Software Documentation," [24] also written for this contract. The following section summarizes the system components.

6.1 Applications Programming Interface

We built the APDS system with an Application Programming Interface (API) so it could be easily integrated into current or future data analysis and health monitoring systems. This interface consists of high level C functions which an application program can use to invoke the functionality of the APDS without concern for the complexity of the underlying software. The delivered API is the mechanism by which the NASA Space shuttle Vehicle Health Monitoring system can utilize the APDS system. In addition, we built and delivered the User's Interface that allows stand-alone operation of the APDS and access to its modules.

The APDS system software handles all file accesses required to load/save a screening set definition. A screening set is defined as the collection of all information necessary to screen a specific sensor on a previously unknown data set from an engine firing. From the user's standpoint, the system now consists of the three functions as follows:

define: prompts the user for and validates information necessary to define a screening set. This information is then stored in an ASCII format file.

train: analyzes data from engine firings, extracts training data and trains neural networks. The result of this process is a screening set which can be used to screen data.

screen: invokes the screening set and related neural networks against a previously unknown engine firing. The output of this process are graphs which show actual sensor values compared to predicted values with confidence bands to identify anomalies.

Engine firing data is read directly from NASA compressed database using NASA software.

6.2 User's Interface

Forms

The user's interface employs a forms-type (fill-in-the-blank), character-based display. The implication of this is that the APDS system will run on any computer or terminal capable of ANSI or VT100 terminal emulation. In fact, we successfully ran the APDS with "shareware" communications programs using 2400 baud modems and telephone connections.

Hypertext Help

A hypertext facility exists as part of the user's interface. It allows on-line manual support for the APDS system. By following conceptual links through this manual, NASA personnel can simply navigate to APDS usage information contained in the on-line manual.

Plotting Routines

Graphical results of an engine data screening use GNUPLOT from the GNU public domain software foundation.

SUN Workstation

We developed the project on a SUN SPARC 2 workstation with 32 megabytes of memory, a CD-ROM unit, 1/4" tape unit, a 424 megabyte internal hard disk drive, and a 1.3 gigabyte external hard disk drive.

Other Procurement

We procured Numerical Recipes software for use in pre-processing transient data, and PV Wave visualization software for the contract.

The workstation ships pre-installed with an end user operating system configuration. The OS was configured for a software development environment including the installation of the C++ tools for use by this project.

6.3 Items Absent Due to Lack of Funding

Due to incomplete funding of the contract we did not complete the full statement of work. Items not completed include:

1. Validation of the system with all PIDs requested by NASA for 15 sets of engine firing data from flights and 12 sets of data from ground tests.
2. Scripts to perform visualization of screening results using PV Wave plotting software.
3. Screening results compared to NASA historical databases.

7 Results and Discussion

7.1 Training

We conducted training sessions with the quickprop neural network to determine which parameters for defining a screening set are most useful for day-to-day operational use of the system by NASA engineers. We tried to find optimal values of the definition parameters based on a compromise between accuracy and speed of training. If the training accuracy is not good enough then possible anomalies may not be flagged by the APDS System. If the training speed is too slow then the system may not meet the goal of increasing data analysis productivity.

Steady-State Engine Operation

Figure 3a shows the progress achieved by the quickprop neural network technique in learning to predict PID 42, fuel preburner oxidizer valve (FPOV) actuator position. The training set was composed of nominal steady-state data from SSME Tests 901-671, 901-672, and 901-673. The ordinate of the figure shows the absolute squared error over the training set. Because training reduces this error by more than two orders of magnitude a logarithmic scale is used is the first figure. Each plotted line in Figure 3a shows the reduction in this error as a function of the amount of training. The lines show the same information for quickprop networks with 1, 2, 5, 10, and 20 nodes in the middle layer. Among these quickprop configurations, best results were obtained with 20 middle nodes after 300 training cycles. Increasing the number of middle nodes slowed down training, as expected for the backprop algorithm.

Figure 3b shows the same information but on a different scale. This scale shows that 10 and 20 middle node networks have the best training accuracy. As a compromise between accuracy and training speed we choose 10 nodes trained for 300 cycles (4800 weight adjustments per input data point) for our production runs that produce screening sets.

Transient Engine Operation

To validate the training and screening of anomalies during transient engine operation we experimented with PID 100 as the predicted variable of the trained neural network. We tested by predicting the time varying nominal values of PID 100 (the average fuel flow measured in gallons per minute) during the 3g throttle-down transients. The focus occurred since tests with anomalies in the transients that we received from NASA appear in PID 100.

Using the methods described in Derived Temporal Indicators section above, we successfully predicted nominal values of PID 100 in 3-g throttle-down transients for SSME tests 511, 519, 527, and 540. The prediction for Test 902-519 is shown in Figure 2 (see Procedure Section above). In this figure, the solid line shows the values of the derived temporal indicator for PID 100 based on actual data measurements of PID 100, while the dashed line in the midst of the solid-line plot shows the nominal values for this temporal indicator as predicted by the neural network. (The horizontal dotted line at zero shows the x-axis for reference.) As can be seen from the figure, agreement between predicted and measured values is excellent for these four nominal tests.

Neural network predictions of derived temporal indicators appear in Figure 4. The anomaly that occurred at about 271 seconds was identified by NASA engineers as a bistability in the engine fuel flow meter (indicated by PID 100).

Flight Firings

We tested the APDS for building screening sets using data from flights of the STS. However, we did not validate any of these screening sets due to funding limitations. The results of the training are shown in Figure 5.

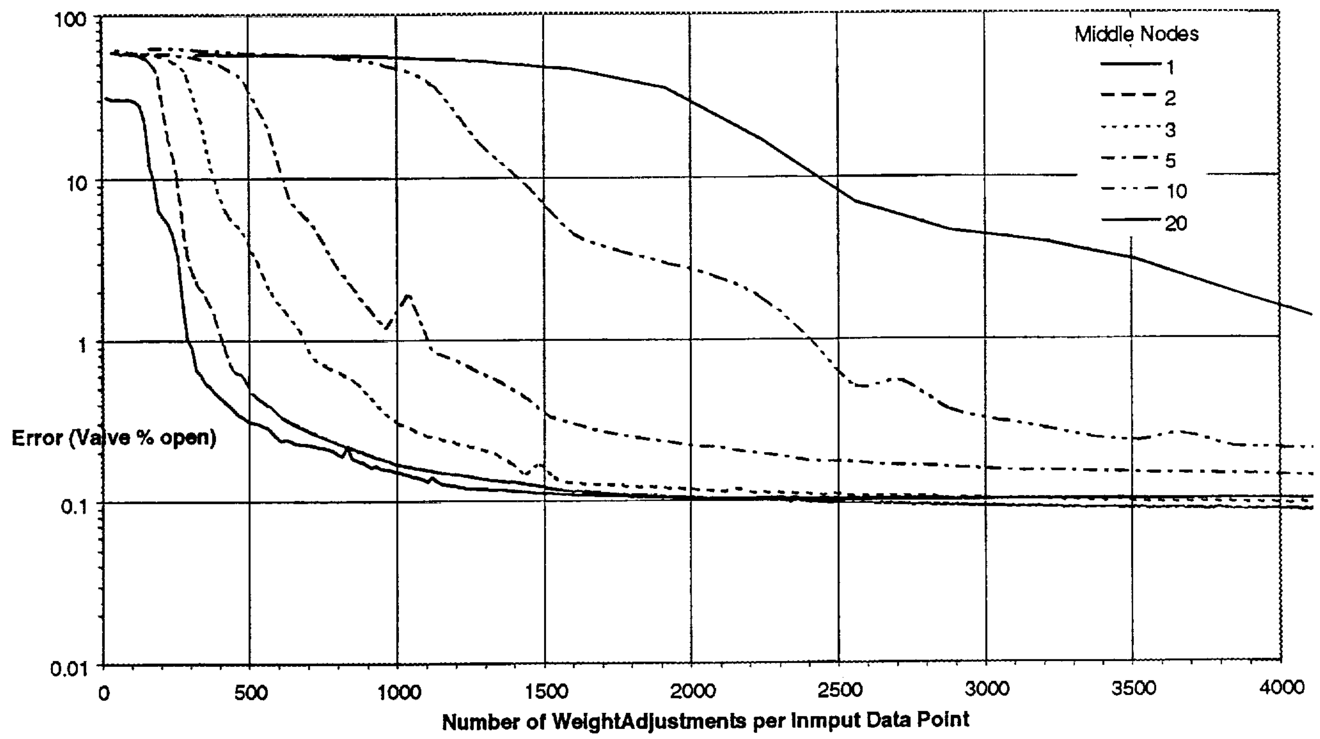


Figure 3a. Training Speed Comparison: Absolute error of the quickprop prediction decreases as training progresses. Lines indicate quickprop networks with 1, 2, 3, 5, 10, and 20 middle nodes.

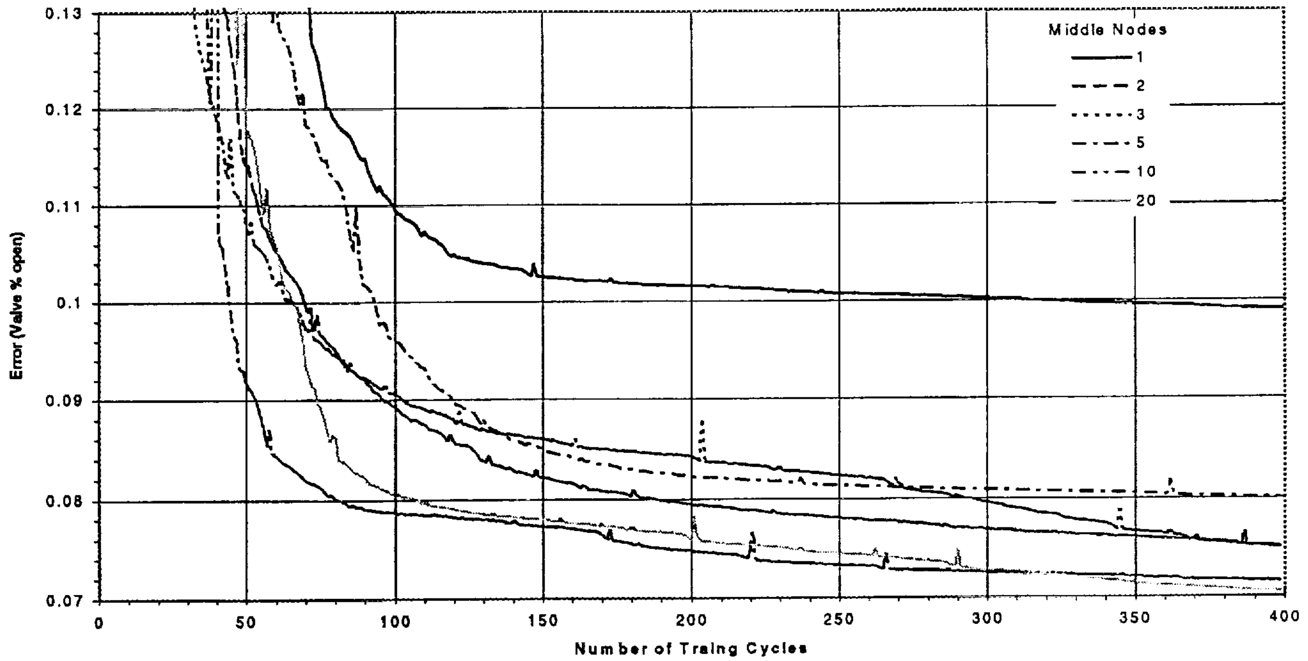
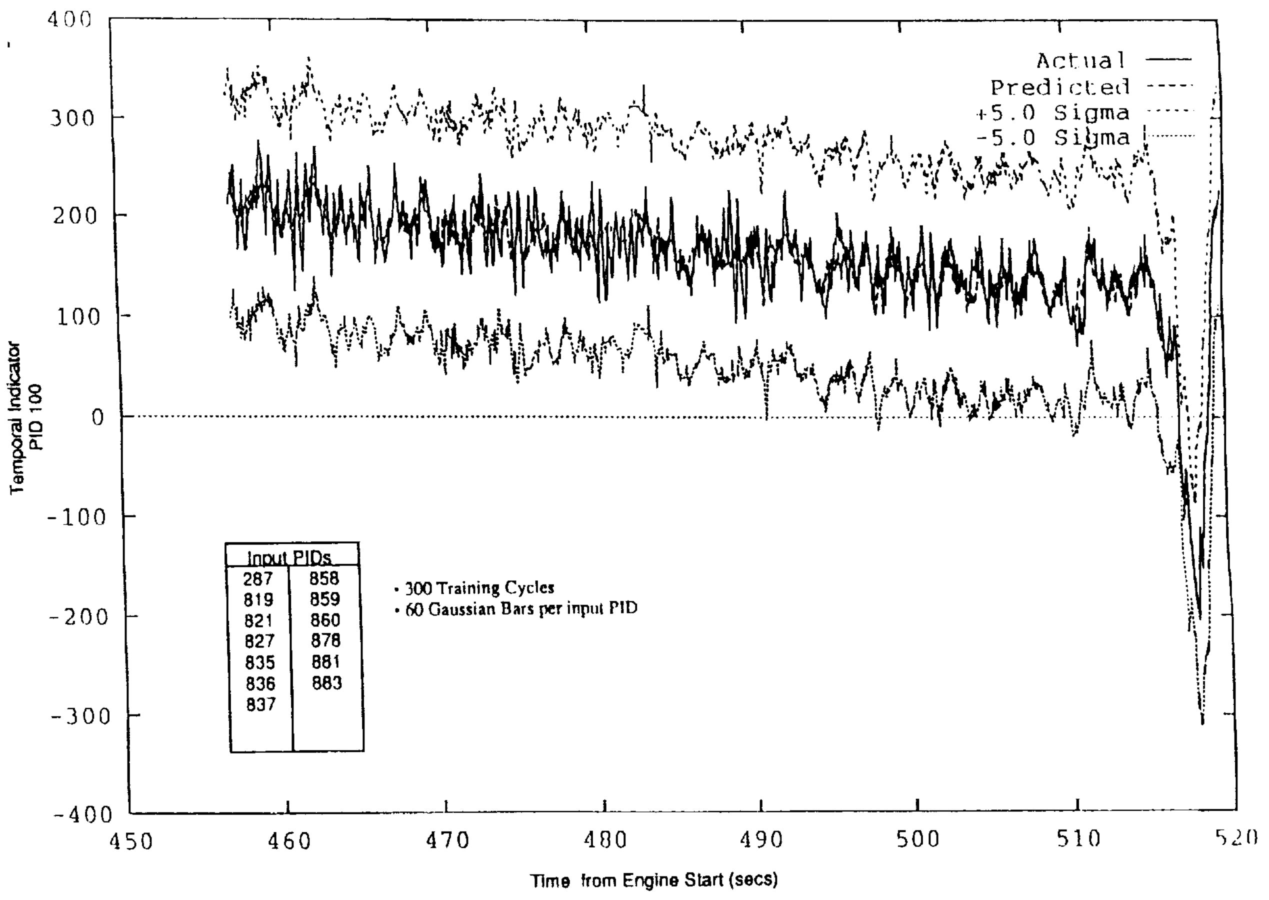


Figure 3b. Training Accuracy Comparison: Absolute error of the quickprop prediction decreases as training progresses. Lines indicate quickprop networks with 1, 2, 3, 5, 10, , and 20 middle nodes. The networks with less middle nodes train faster but do not have as good accuracy after many weight adjustments.



Input PIDs	
287	858
819	859
821	860
827	878
835	881
836	883
837	

Figure 4. Training Results of Nominal Data for Engine Firing 902-519, during a 3-g Throttle Down Transient.

Figure 4. Training Results of Nominal Data for Engine Firing 902-519, during a 3-g Throttle Down Transient.

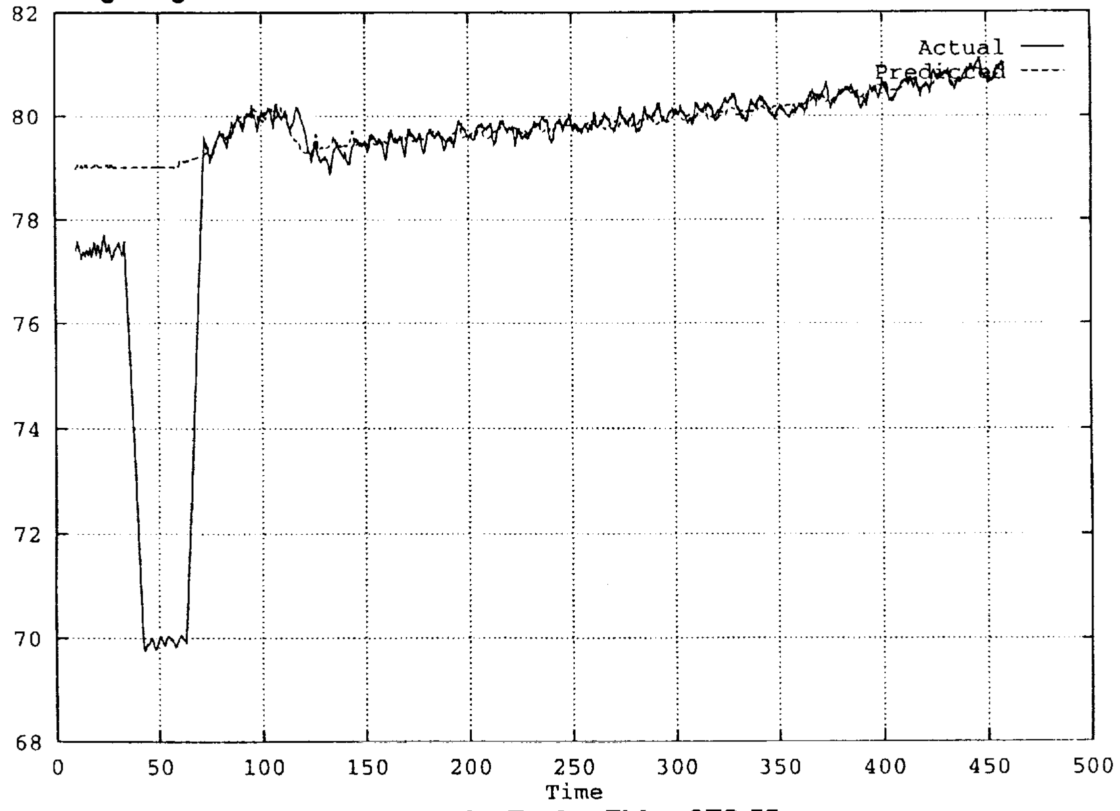


Figure 5. Training Results of Nominal Data for Engine Firing STS-55, during Steady-state Operation for PID 42.

Similarities Required

Hardware

Baseline and Engine-sensitive Screening Sets

The original system design provides two types of screening sets (1) engine-sensitive screening for firings whose major components were represented in the training set, and (2) baseline screening for firings containing new components for which the neural networks have not been trained. Both of these types of screening are currently in operation. As anticipated from the Phase I results, baseline training results in much wider confidence bands (less sensitivity and thus less chance of finding small anomalies) than engine-sensitive training.

We investigated whether the engine-sensitive training method can be refined to provide better performance than the baseline method on engine firings with new components. Experimental software to investigate this was developed. The results of this work show that:

1. Engine firings with one or two component changes to untrained hardware (i.e., that have never been used for training) can be successfully modeled using the new software. Figure 6 shows the results of screening ground test A2-549 with a baseline model (Figure 6a) and our new engine-sensitive model (Figure 6b). The new method gives much better results than the baseline. Ground test A2-549 has two components, the controller and HPOP, that are new to the screening set.
2. Many component changes to untrained hardware result in no better, if not worse, results than screening with a baseline screening set. Figure 7 shows the results of screening ground test A2-511 with a baseline model (Figure 7a) and the new engine-sensitive model (Figure 7b). Note that the results of both models are similar and only give information about a PID signature (trend) as opposed to predicting the actual values.
3. To find the type of anomalies given to us as examples the APDS should be employed with engine sensitive models (screening sets trained with hardware).

Engineers should pick nominal engine firings to train (build) screening sets that have as much similar hardware to the new firing to be screened as possible (try at least only one different component).

As expected, and verified under this work and the Phase I work, hardware differences play a major role in training and screening data using the APDS. We found that more than two changes in hardware from the engine firing(s) used for training versus the data screened affected the system's ability to screen properly. Figure 1 shows the results of screening a nominal firing that had several hardware differences from the data used for training. In general the trend of the prediction matches the trend of the actual data. However, the values of the prediction exhibit what may be called a bias across the whole set of data from engine firings. This bias results from the fact that the screening set used data for training that did not contain all of the same hardware components in the data that we screened.

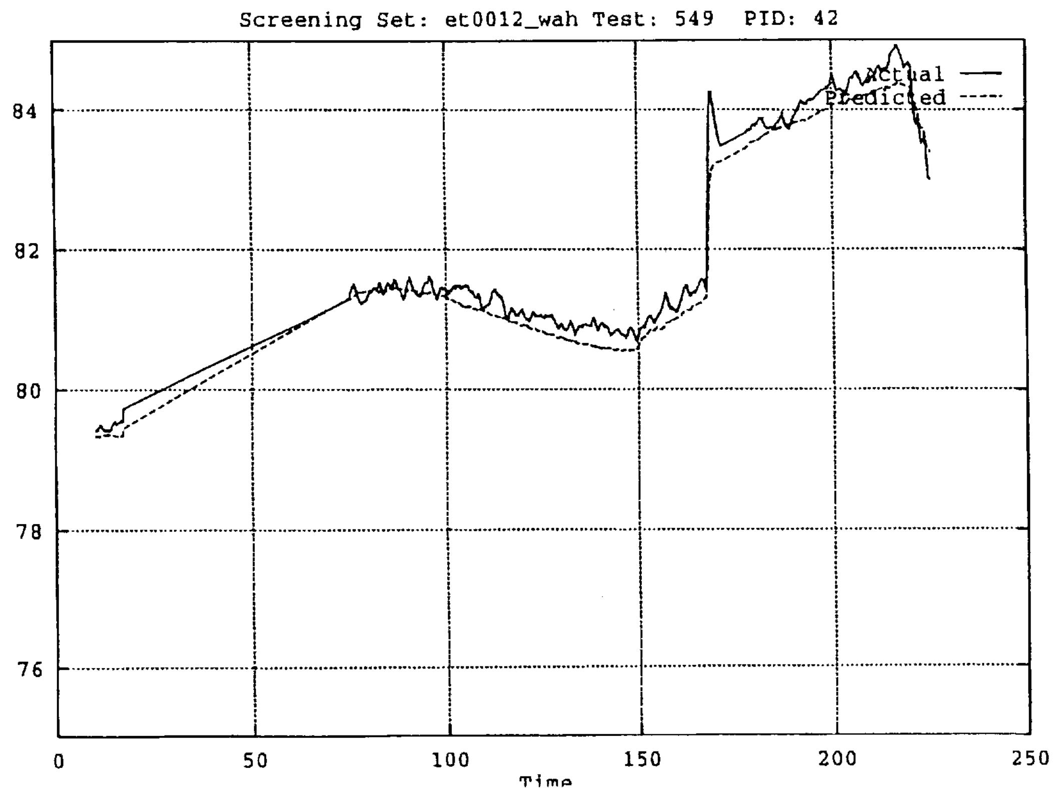


Figure 6. Training Results WITH All Hardware Represented for an Engine Sensitive Screening Set

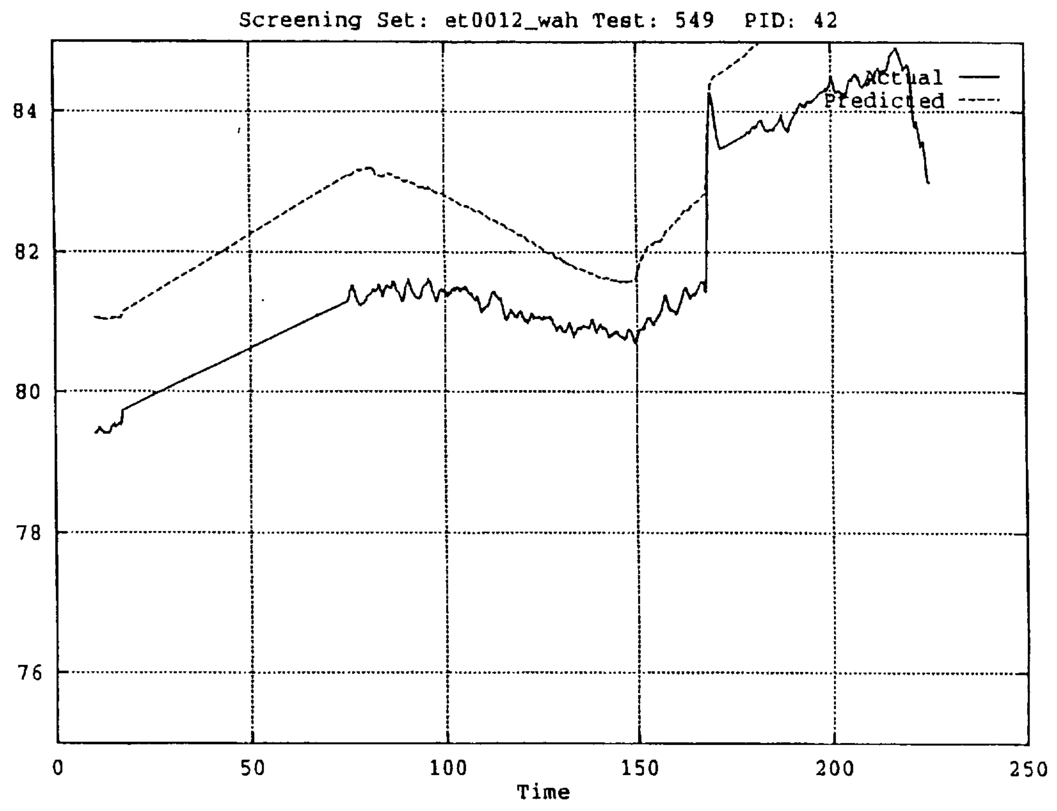


Figure 7. Training Results WITHOUT All Hardware Represented for a Baseline Screening Set

Power Level

Like hardware screening sets must be built with all power levels that will be screened. A good example of this occurred with ground test A1-672. When we did not train with the data at the 100% power level we did not match the actual data. According to NASA engineers this part of the data was nominal. Figures 8a and 8b show the screening with and without using data from the 100% power level.

7.2 Screening

Figure 9a and 9b show screening results of ground test A1-672 for PID 42 and A2-549 for PID 221, respectively. Since we did not get enough tests (see the Validation section below) for other anomalies we did not test screening sets for any other PIDs against anomalies. These results are the same as presented in our Phase I report.

However, this demonstration contract includes the ability to screen transients during ground tests and flight data. Much of the effort was spend on building these two new screening modules and thus we did not build any new screening sets for PIDs other than 42 and 221 for steady state operation.

7.3 Validation

By validated system, we mean a system that might actually work "out of the box" on new untried data sets. We believe a minimum of nine months would be needed to fully validate the system.

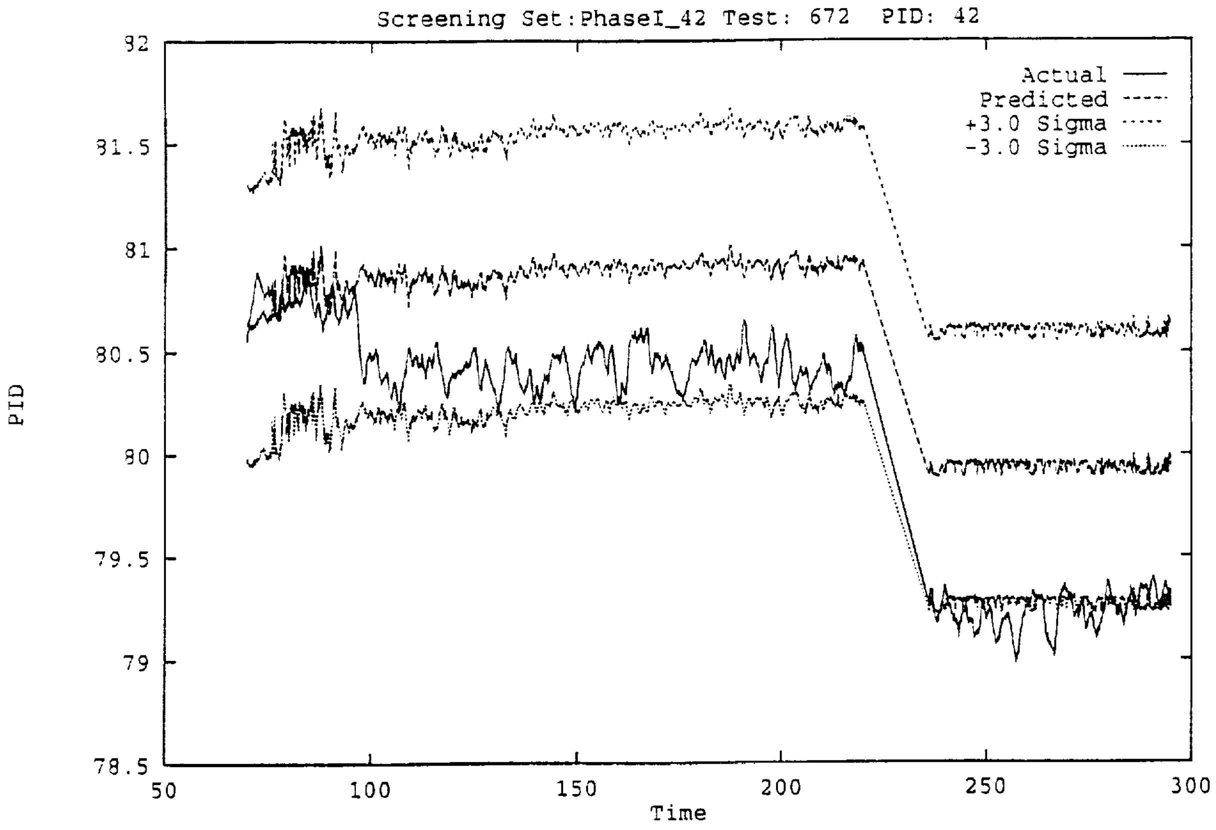
We constructed temporal indicators which filer raw time-series input data to reflect not just the instantaneous state measured by a sensor, but its recent history over the time scale associated with a transient. These temporal indicators were built by combining kernels of digital filters of different sizes to capture trends at distinct scales. The component digital filters were constructed using singular value decomposition so that linear combinations of independent variables of small incremental predictive value could be detected and eliminated form the temporal indicators in the delivered system.

For the predicted variable whose nominal state is being predicted, temporal indicators are derived to yield predictions on different time scales as discussed in the Derived Temporal Indicators Section above. These indicators are obtained by taking shifted differences of the kernels obtained from component digital filters of different lengths.

Using these canonical filters, our neural network method has detected the transient anomalies for which ERC has data. We do not, however, possess enough examples of transient anomalies to validate these temporal indicators–i.e. to provide confidence that the indicators will be robust for detecting anomalies different than those for which we have experimented. The data is available to us but we need NASA engineers' expertise to pick more anomalies and to understand the behavior of these anomalies.

We have defined a few different categories of power-level changes (excluding startup and shutdown transients) based on the magnitude, direction, and slope of the change. We would like to deliver trained and validated temporal indicators for as many of these categories as the availability of validating data permits.

To validate a set of temporal indicators for a given category of power-level change, both nominal and anomalous data for transients in that category are required. For each given category (of magnitude, slope, and direction) of power-level transient to be validated, nominal data from at least two different ground tests, and two different anomalies from at least two different engine firings, would be the minimum required to give us confidence in the robustness of the derived temporal indicator trained on that category.



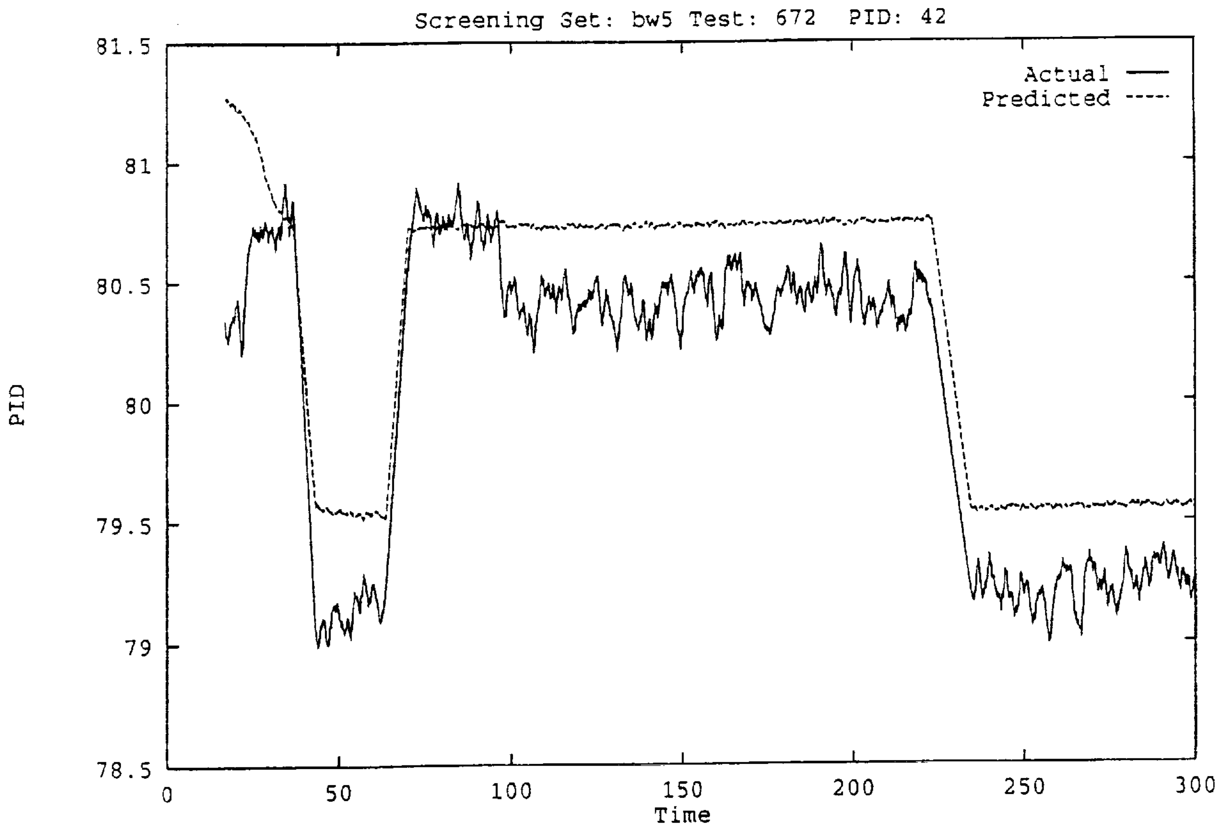
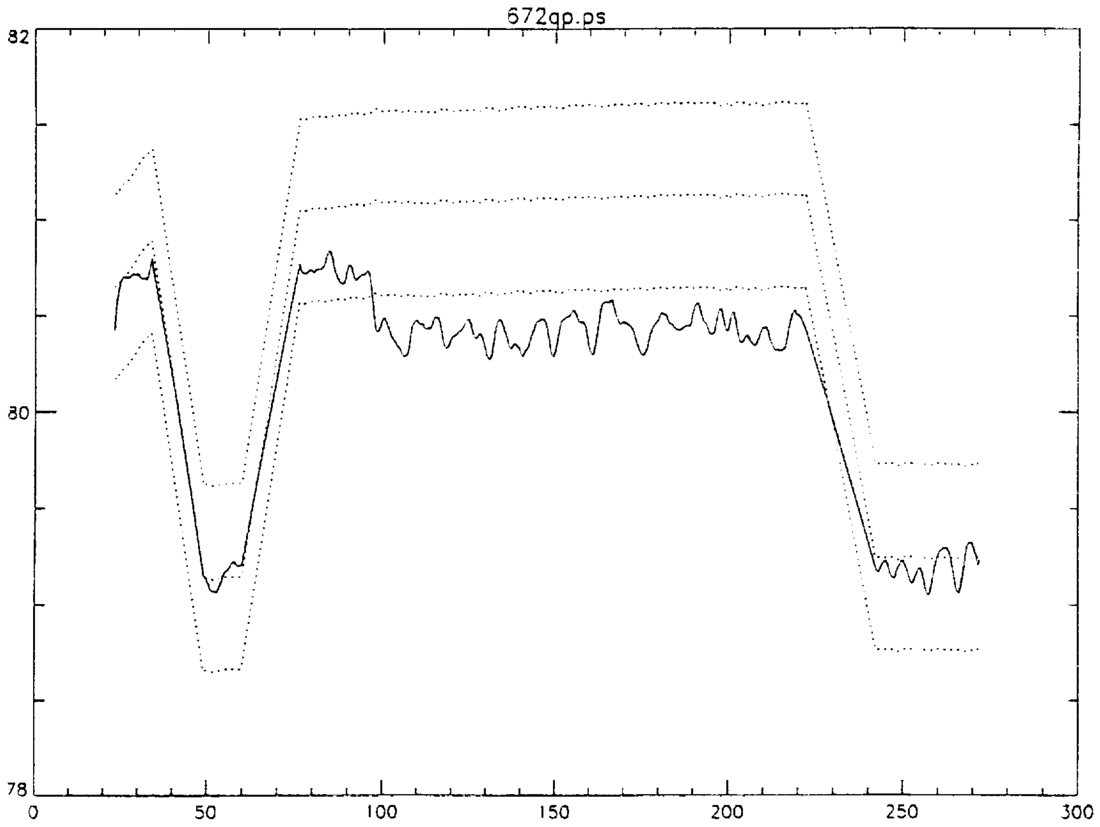
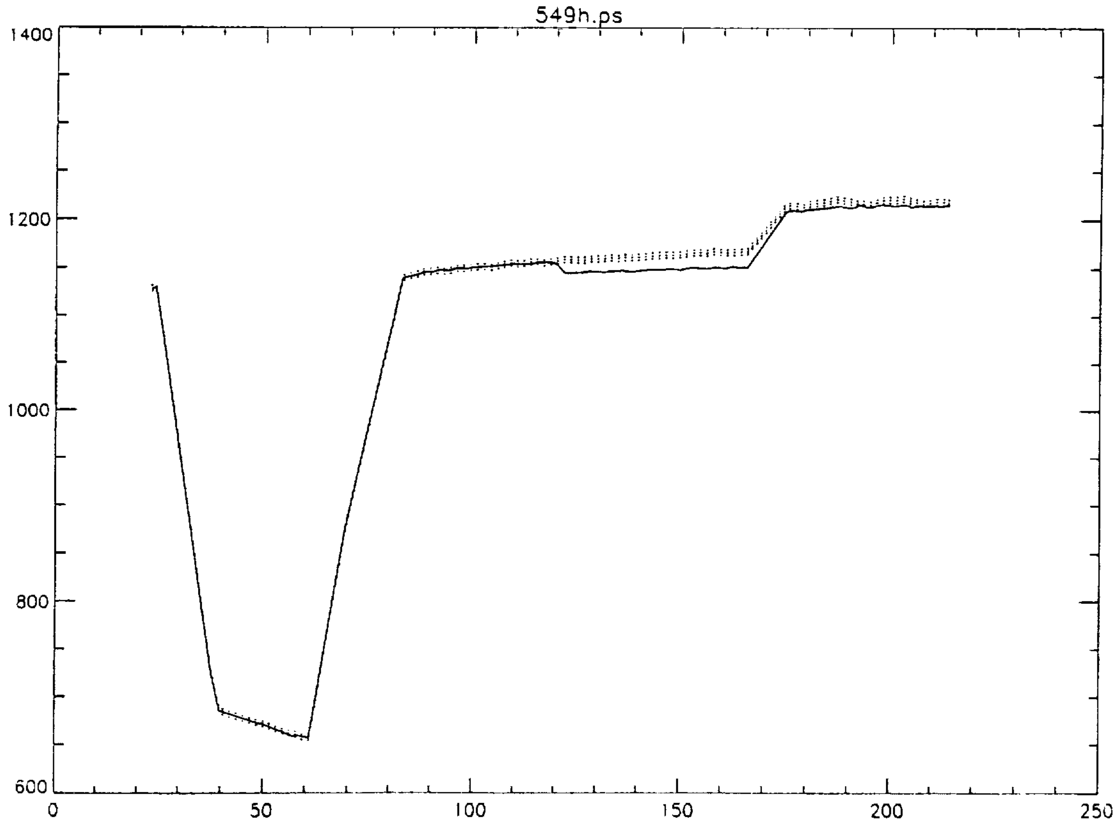


Figure 8a and 8b. Training Results of Nominal Data for Engine Firing A1-672, during Steady-state Operation for PID 42 with and without the 100% Power Level in the Training Data.





Figures 9a and 9b. Nominal Predictions. Screening of data for Anomalies in Ground Tests A1-672 (PID 42) and A2-549 (PID 221).

Similarly, to validate steady-state screening sets for a given power level, nominal data from at least two different engine firings, and two different anomalies at that power level from at least two different engine firings, would be the minimum required to give us confidence in the robustness of the screening set.

The anomalous data is not used to construct the indicator, but to test its validity. In order to screen for the appropriate PIDs, understand the results of our testing, and intelligently compare the neural network screening results with NASA analyses, we would need the NASA documentation of each anomaly which identifies the time(s) at which the anomaly occurs, the PID which best shows the anomaly, and the characteristics of this data which NASA engineers consider to be anomalous.

7.4 Function Approximation Approach Delivered

We choose to deliver the quickprop over the Gaussian bar basis function as the approximation technique. The Gaussian bar basis function does not generalize as well in situations where the test data has exists in regions of the input space where there was no training data. Because the basis functions are localized Gaussians, the Gaussian bar basis function does not do a very good job of smoothing over a big "hole"—a region of the input space where there are no data points in the training set. The basis functions which fall entirely within such holes will never see any training points and thus will never train. As a result these basis function will always give an output of zero (where zero gets scaled to the mean output for the whole data set.)

Quickprop, by contrast, uses nonlocal sigmoid basis functions which flatten out but do not fall to zero in such holes. The net effect is that the quickprop will tend to smooth over holes in the training data with a localized averaging from around the hole rather than with a global average like the Gaussian bar basis function.

The bottom line is that when you screen data which happens to fall in regions of the input space where there was no training data, the quality of the Gaussian bar basis function approximation will degrade severely. The quality of the quickprop will also degrade, but not as badly.

Also, setting up the quickprop network requires much less work for the user. In general, deciding the required spacing of the basis functions was often a black art. In comparison, with quickprop you can increase (within reason, we never used more than 20) the number of middle nodes until you get good fits of the training data.

8. Conclusion

8.1 Contract Goal

The main goal of this contract was to deliver a demonstration system that NASA engineers could use to perform some data screening. Secondary to that goal was to deliver a robust system that is easy to operate so NASA data analysts could better understand the APDS System capabilities and the advantages it can offer for data analysis.

The system as delivered is easy to run and can be used to screen data and flag anomalies. The current limitation is that the system *must* have sufficient data input to it during the training process for the screening process to yield useful results. The input data stream to the training process must contain data for all the hardware at all power levels that exist in any data sets that will be screened in the future using a specific screening set. However, since the system is easy to operate and engineers can readily build new screening sets as data becomes available, new screening sets built using sufficient input data can be constructed on a daily basis.

Although we conducted significant research with the APDS System, we strove to deliver a system that can demonstarte to NASA engineers the use of applying neural networks as a data screening system. We believer the current APDS System and its User Interface provides such a system.

8.2 Hypothesis

The main hypothesis that enabled development of the APDS System was that the nominal value of a given engine parameter at a given time could be predicted from external influences on the SSME. Based on the boundary we drew around the SSME, these external influences included not only the commanded power level, but also the pressures in the fuel and oxidizer inlets, venting lines, and repressurization lines. These measurements of external influences were used to predict the nominal values of two internal engine parameters. More specifically, our hypothesis was that this prediction could be made with a tight enough confidence interval to be useful for detecting anomalies, in spite of the many sources of variability within the SSME and its measurement channels.

This hypothesis appears to be supported within the limitations of data available to build screening sets. If the anomalies to be detected are considered signals, and nominal fluctuations in the measurements are considered noise, then both anomalies used as test cases were detected at over three times the RMS noise level. With nominal predictions surrounded by a confidence interval of five times the RMS prediction error, no false alarms would be expected from nominal fluctuations, and none were observed. However, the input data sets had to be limited to no more than three engine firings. Also, to screen tests containing anomalies, nominal data from the engine firing where the anomaly occurred was the only available data for the hardware used in the firing at the proper power level of engine operation.

8.3 Recommendations

The chief appeal of the function-approximation approach to anomaly detection is that it is not limited to detecting specific, foreseen classes of anomalies, in contrast with most other types of neural network and expert system approaches. Instead, training requires nominal data only, and anomaly detection is based on nominal confidence intervals. Based on the results we have presented, it appears that the function-approximation approach merits more extensive investigation of its practicality for screening large amounts of propulsion system firing data. The approach needs continued testing on a larger scale, and it needs to be extended to cover transient during flights of the STS.

Validation of the results are needed and will take a joint effort between NASA data analysis experts and contract personnel to choose and screen data from appropriate engine firings for all the PIDs that NASA wants to screen and analyze.

9. References

1. D. Broomhead And D. Lowe, "Multivariable functional interpolation and adaptive networks," *Complex Systems*, Vol. 2, pp. 321-355, 1988.
2. T. Poggio And F. Girosi, "Networks for approximation and learning," *Proceedings Of The IEEE*, Vol. 78, no. 9, pp. 1481-1497, 1990.
3. J. Park And I. Sandberg, "Universal approximation using radial-basis-function networks," *Neural Computation*, Vol. 3, *pp*. 246-257, 1991.
4. S. Geva And J. Sitte, "A constructive method for multivariate function approximation by multilayer perceptrons," *IEEE Transactions On Neural Networks*, Vol. 3, *pp*. 621-624, 1992.
5. Y. Ito, "Approximation of functions on a compact set by finite sums of a sigmoid function without scaling," *Neural Networks*, Vol. 4, *pp*. 817-826, 1991
6. Y. Ito, "Approximation of continuous functions on R^d by linear combinations of shifted rotations of a sigmoid functions with and without scaling," *Neural Networks*, vol. 5, pp. 105-115, 1992.
7. V. Venkatasubramanian and K. Chan, "A neural network methodology for process fault diagnosis," *AIChE*, vol. 35, pp. 1993-2002, 1989.
8. T. H. Guo and J. Nurre, "Sensor failure detection and recovery by neural networks, in *IJCNN-91-Seattle: International Joint Conference on Neural Networks*, (Seattle, WA), pp. I-221-I-226, IEEE, July 1991.
9. M. Kotani, H. Matsumoto, and T. Kanagawa, "Acoustic diagnosis for compressor with hybrid neural network," in *IJCNN-91-Seattle: International Joint Conference on Neural Networks*, (Seattle, WA), pp. I-251-I-256, IEEE, July 1991.
10. P. A. Jokinen, "Comparison of neural network models for process fault detection and diagnosis problems," in *IJCNN-91-Seattle: International Joint Conference on Neural Networks,*, (Seattle, WA), pp. I-239-I-244, IEEE, July 1991.
11. K. J. Cios, R. E. Tjia, N. Liu, and R. A. Langenderfer, "Study of continuous ID3 and radial basis function algorithms for the recognition of glass defects," in *IJCNN-91-Seattle: International Joint Conference on Neural Networks*, (Seattle, WA), pp. I-49-I-54, IEEE, July 1991.
12. M. yuen Chow and S. O. Yee, "Robustness test of an incipient fault detector artificial neural network," in *IJCNN-91-Seattle: International Joint Conference on Neural Networks*, (Seattle, WA), pp. I-73-I-78, IEEE, July 1991.
13. B. A. Whitehead, H. J. Faber, and M. Ali "Neural network approach to space shuttle main engine health monitoring," in AIAA/ASME/SAE/ASEE 26th Joint Propulsion Conference, (Orlando, FL), July 1990.
14. B. A. Whitehead, E. L. Kiech, and M. Ali, "Rocket engine diagnostics using neural networks' in *AIAA/ASME/SAE/ASEE 26th Joint Propulsion Conference*, (Orlando, FL) pp. 2-11, July 1990.
15. D. R. Hush and J. M Salas, "A performance of neural network classifiers for the 1-class classifier problem," in *Proceeding of the International Joint Conference on Neural Networks*, (Washington, D.C.), pp. I-396-I-407, IEEE, January 1990.
16. J. Leonard, M. Kramer, and L. Ungar, "Using radial basis function to approximate a function and its error bounds," *IEEE Transactions on Neural Networks*, vol. 3, no. 4, pp. 624-627, 1992.

17. J. Moody and C. J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Computation*, vol. 1, pp. 281-294, 1989.

18. E. Hartman and J. D. Keeler, "Semi-local units for prediction," in *IJCNN-91-Seattle: International Joint Conference on Neural Networks*. (Seattle, WA), pp. II-561-II-566, IEEE, July 1991.

19. E. Hartman and J. D. Keeler, "Predicting the future: Advantage of semilocal units," *Neural Computation*, vol. 3, pp. 566-578, 1991.

20. D. Rumelhart, G. Hinton, and R. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations* (D. E. Rumelhart, J. L. McClelland, and T. P. R. Group, eds.), pp. 318-364, Cambridge, Ma: The MIT Press, 1986.

21. Fahlman, Scott E. "Faster-Learning Variations on Back-Propagation: An Empirical Study," *Proceedings of the 1988 Connectionist Models Summer School*, p. 38 ff, 1988, Morgan Kaufmann.

22. Whitehead, Bruce A. and Hoyt, W. Andes, "A Function Approximation Approach to Anomaly Detection in Propulsion System Test Data," paper AIAA 93-1776, AIAA 29th Joint Propulsion Conference, American Institute of Aeronautics and Astronautics, Washington, DC, 1993.

23. Hoyt, Susan B., "The Automated Propulsion Data Screening System User's Guide," NASA Contract NAS8-39782, Marshall Space Flight Center; ERC, Incorporated; Tullahoma, TN, 1994.

24. Choate, Timothy B., "The Automated Propulsion Data Screening System Software Specifications," NASA Contract NAS8-39782, Marshall Space Flight Center; ERC, Incorporated; Tullahoma, TN, 1994.