CHAPTER 6
Bayesian Neural Networks

THE ESSENTIAL FEATURE of a residual processor is to analyze the residual process generated by all fault detection filters and announce whether or not a fault has occurred and with what probability. This requires higher level decision making and creation of rejection thresholds. Nominally, the residual process is zero in the absence of a fault and non-zero otherwise. However, when driven by sensor noise, dynamic disturbances and nonlinearities, the residual process fails to go to zero even in the absence of faults. This is noted in the simulation studies of the detection filters. Furthermore, the residual process may be nonzero when a fault occurs for which the detection filter is not designed. In this case, the detection filter detects but cannot isolate the fault because the residual directional properties are not defined.

The approach taken in this section is to consider that the residuals from all fault detection filters constitute a pattern, a pattern which contains information about the presence or absence of a fault. Hence, residual processing is treated as a pattern recognition problem. This class of problems is ideally suited for application to a neural network.

The objective of a neural network as a feature classifier is to associate a given feature vector with a pattern class taken from a set of pattern classes defined apriori. In an application to residual processing, the feature vector is a fault detection filter residual and the pattern classes are a partitioning of the residual space into fault directions which include the null fault.

Three types of neural network classifiers are considered for the pattern recognition problem: a single layer perceptron, a multilayer perceptron and a Bayesian neural network. The single layer perceptron is the simplest continuous input neural network classifier and has the ability to recognize only simple patterns. It decides whether an input belongs to one of the classes by forming decision regions separated by hyperplanes. It is shown later that the decision regions formed by the single layer perceptron are similar to those formed by a maximum likelihood gaussian classifier if the inputs are gaussian, uncorrelated and the distributions for different classes differ only in the mean values. Note that the perceptron training procedure may lead to oscillating decision boundaries if the underlying distributions of the input intersect, that is, if the classes are not mutually exclusive.

The multilayer perceptron is a feedforward network with input, output and, possibly, hidden layers. Unlike the single layer perceptron, which partitions the decision space with hyperplanes, the multilayer perceptron forms arbitrarily complex convex decision regions. Furthermore, since no assumptions are required about the shapes of the underlying input probability distributions, the multilayer perceptron is a robust classifier that may be used to classify strongly non-gaussian inputs driven by nonlinear processes.

The Bayesian neural network is a multilayer perceptron with output feedback and is modified to include a sigmoidal activation function at each ouput node. The output activation functions take values between zero and one. It is shown later, in Section 6.2.2, that the output activation functions of a Bayesian neural network provide posterior probabilities of classification conditioned on the applied input history. A stochastic training algorithm further enhances robustness in that training sets are considered as sample sets providing information about the entire population. This is explained in Section 6.3.

6.1 Notation

Notation for a q-layer multilayer perceptron is as follows.

n_i	number of nodes in layer i.
$u_k \in \mathrm{R}^{n_1}$	network input at time k.
$x_k^i \in \mathrm{R}^{n_i}$	input to layer i at time k where $i \in \{2, \ldots, q\}$.
$y_k^i \in \mathrm{R}^{n_i}$	output of layer i at time k where $i \in \{1, 2, \ldots, q\}$.
$S(x)$	activation function.
$\Phi^i \in \mathrm{R}^{n_i}$	bias vector of layer i where $i \in \{2, \ldots, q-1\}$.
$W^i \in \mathrm{R}^{n_i \times n_{i-1}}$	weighting matrix of layer i where $i \in \{2, \ldots, q\}$.

Connections for a q-layer multilayer perceptron with one step delayed output feedback are defined in (6.1). The connections are illustrated in Figure 6.1 for a five layer network.

$$x_k^1 = u_k + y_{k-1} \tag{6.1a}$$

$$x_k^i = W^i y_k^{i-1} + \Phi^i, \qquad \text{where } i \in \{2, \ldots, q\} \tag{6.1b}$$

$$y_k^i = S(x_k^i), \qquad \text{where } i \in \{1, \ldots, q\} \tag{6.1c}$$

$$S(x) = \frac{e^x}{e^x + 1} \tag{6.1d}$$

6.2 Bayesian Feature Classification and Neural Networks

A Bayesian feature classifier is optimal in the sense that it assigns a feature to the pattern class with the highest posterior probability, that is, a feature vector x is associated with a pattern class $\mathcal{A}_i$ if

$$P(\mathcal{A}_i/x) > P(\mathcal{A}_j/x) \qquad \forall j \neq i$$

Most classifiers use probabilities conditioned on the class $P(x/\mathcal{A}_i)$ and use Bayes' rule to generate posterior probabilities, that is,

$$P(\mathcal{A}_i/x) = \frac{p(x/\mathcal{A}_i)p(\mathcal{A}_i)}{p(x)}$$

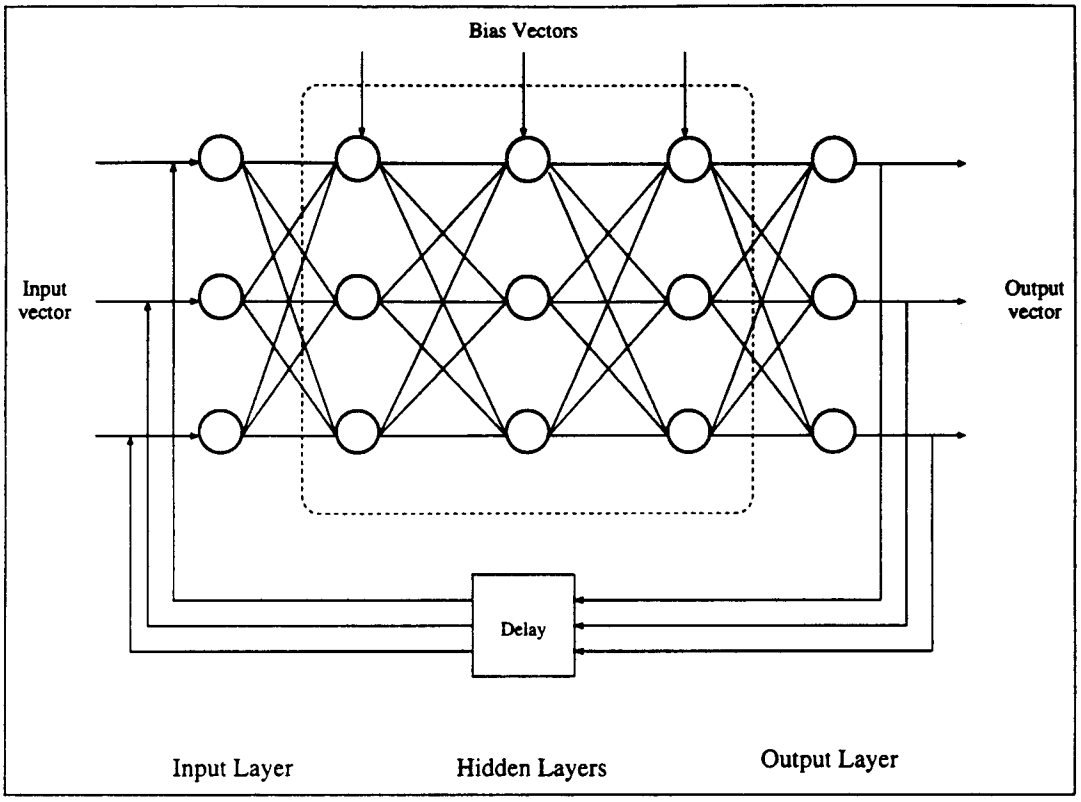

Figure 6.1: Bayesian neural network with feedback.

$$P(x) = \sum_{j=1}^{m} p(x/\mathcal{A}_j)p(\mathcal{A}_j)$$

This indirect way of calculating posterior probabilities makes assumptions about the form of the parametric models $P(x/\mathcal{A}_i)$ and the apriori probabilities (Morgan and Bourlard 1995).

Multilayer perceptrons do not require any assumptions about the pattern distributions and can form complex decision surfaces. Several authors (Richard and Lippmann 1991, Bourlard and Wellekens 1994) show that the outputs of multilayer perceptron classifiers can be interpreted as estimates of posterior probabilities of output classification conditioned on the input. Blaydon (Blaydon 1967) proved the same for a two-class linear classifier.

The following subsections provide two results that establish the utility of multilayer perceptrons as Bayesian feature classifiers. Section 6.2.1 shows that the decision regions created by a maximum likelihood gaussian classification algorithm can be generated using

a multilayer perceptron with sigmoidal node functions. Section 6.2.2 shows that the output of a Bayesian neural network can be interpreted as an estimate, conditioned on the input history, of the posterior probabilities of feature classification.

6.2.1 A Maximum Likelihood Gaussian Classifier as a Multilayer Perceptron

In this section, it is shown that the decision regions created by a maximum likelihood gaussian classification algorithm can be implemented using a multilayer perceptron with sigmoidal node functions. First, consider a binary hypothesis case, that is, one where the input is assumed to be associated with one of two classes.

Let the input of a maximum likelihood gaussian classification algorithm be $x \in \mathrm{R}^n$, the output $y \in \mathrm{R}^2$ and the two classes $\mathcal{H}_i$ and $\mathcal{H}_j$. For simplicity, assume that the underlying conditional probability density functions of x have identical covariances but different means as in

$$\begin{aligned} \mathcal{H}_i: &\qquad x \sim \mathcal{N}(m_i, \Lambda) \\ \mathcal{H}_j: &\qquad x \sim \mathcal{N}(m_j, \Lambda) \end{aligned}$$

where

$$f(x/\mathcal{H}_i) = \frac{1}{(2\pi)^{n/2}|\ \Lambda|^{1/2}} \exp\left\{-\frac{1}{2}\|x - m_i\|^2_{\Lambda^{-1}}\right\}$$

Now define a log likelihood function L_i as:

$$\begin{aligned} L_i &\triangleq 2\ln\left[(2\pi)^{n/2}|\Lambda|^{1/2} f(x/\mathcal{H}_i)\right] \\ &= -(x - m_i)^T \Lambda^{-1}(x - m_i) \end{aligned}$$

Then, the difference between the two liklihood functions $L_i - L_j$ has the form

$$\begin{aligned} L_{ij} &\triangleq L_i - L_j \\ &= 2(m_i - m_j)^T \Lambda^{-1} x + (m_j^T \Lambda^{-1} m_j - m_i^T \Lambda^{-1} m_i) \\ &= Wx + \Phi \end{aligned}$$

An input classification decision function follows as

$$\begin{aligned} L_{ij} &> 0 \quad \Rightarrow \text{Declare } \mathcal{H}_i \\ L_{ij} &< 0 \quad \Rightarrow \text{Declare } \mathcal{H}_j \end{aligned}$$

The same decision region could be obtained using a single layer perceptron in which the weighting matrix is W, the bias vector is Φ and the output is a sigmoidal function of the form $S(L_{ij})$. An easy extension to the multiple hypothesis case follows from a decision function based on $L_i - \max_{j \neq i} L_j$.

The similarity in form between a maximum likelihood gaussian classifier, as above, and a perceptron is obvious. However, note that traditional statistical classifiers require prior knowledge of the stochastic properties of the inputs. This is not so for perceptrons. Furthermore, it can be shown that multilayer perceptrons with hidden layers and sigmoidal nodal functions behave as universal approximators, that is, they have the capability of approximating any function to any degree of accuracy given a sufficient number hidden nodes. Refer to (Funahashi 1989, Hornik et al. 1989) for details.

6.2.2 A Bayesian Neural Network Provides Feature Classification Probabilities

This section shows that the output of a Bayesian neural network can be interpreted as an estimate of the posterior probabilities of feature classification conditioned on the input history. Let $x_k \in \mathrm{R}^n$ be a feature vector and $\mathcal{X}_k = \{x_1, \ldots, x_k\}$ be a history of feature vectors. Let

$$\mathcal{A} = \{\mathcal{A}_1, \ldots, \mathcal{A}_q\}$$

be a set of q pattern classes $\mathcal{A}_i$ into which a feature vector may be classified and let $y(w, x) \in \mathrm{R}^m$ be the output of a multilayer perceptron. The parameter w is a vector containing the perceptron weights and bias vectors. Let $z_k \in \mathrm{R}^m$ be a vector defined as:

$$z_k^T \triangleq \begin{cases} [0, \ldots, 1, \ldots, 0], & \mathcal{X}_k \in \mathcal{A}_i \\ [0, \ldots, 0, \ldots, 0], & \mathcal{X}_k \notin \mathcal{A}_i, \end{cases} \qquad i \in \{1, \ldots, m\}$$

From the definition of the Bayesian neural network connections, (6.1), the conditional expectation of z_k is

$$E[z_k/x_k] = P(\mathcal{A}/\mathcal{X}_k)$$

where $E[\cdot]$ is the expectation operator and where $P(\mathcal{A})$ is a vector of probabilities

$$P(\mathcal{A}/\mathcal{X}_k) = \begin{bmatrix} P(\mathcal{A}_1/\mathcal{X}_k) \\ \vdots \\ P(\mathcal{A}_q/\mathcal{X}_k) \end{bmatrix}$$

Note that if the $\mathcal{A}_i$ are mutually exclusive and exhaustive events, then $\|P(\mathcal{A}/\mathcal{X}_k)\|_1 = 1$.

Consider the regression function

$$J(w) = E_{x,z}\left[\|z - y(w,x)\|^2\right] \tag{6.2}$$

An expansion of the norm and the expectation operator lead to

$$\begin{aligned} J(w) &= E_x\left[E_z\left[\|z - y(w,x)\|^2/x\right]\right] \\ &= E_x\left[E_z\left[\|z\|^2 - 2y^T(w,x)z + \|y(w,x)\|^2/x\right]\right] \\ &= E_x\left[\sum P(\mathcal{A}_i/\mathcal{X}) - 2y^T(w,x)P(\mathcal{A}/\mathcal{X}) + \|y(w,x)\|^2\right] \\ &= E_x\left[\sum P(\mathcal{A}_i/\mathcal{X}) - \|P(\mathcal{A}/\mathcal{X})\|^2\right] + E_x\left[\|P(\mathcal{A}/\mathcal{X}) - y(w,x)\|^2\right] \end{aligned}$$

Since the first expectation term is independent of the multilayer perceptron parameters, minimization of J is the same as minimization of F where

$$J(w) = E_x\left[\sum P(\mathcal{A}_i/\mathcal{X}) - \|P(\mathcal{A}/\mathcal{X})\|^2\right] + F(w)$$

and

$$F(w) \triangleq E_x\left[\|P(\mathcal{A}/\mathcal{X}) - y(w,x)\|^2\right]$$

Thus, when the network parameters are chosen to minimize a mean-squared error cost function, the outputs are estimates of the Bayesian posterior probabilities.

6.3 Learning Algorithms for Neural Networks

The learning phase of a neural network involves the determination of the synaptic weights and bias vectors of the network. The backpropagation algorithm, the most widely used learning algorithm in neural network applications, consists of two passes through the layers of the network: a forward pass and a backward pass. In the forward pass, an input is applied to the input layer and allowed to propagate through the network to produce an output. During this pass, the synaptic weights and bias vectors are held fixed. In the backward pass, the network output is compared to a desired output and an error vector is formed. As the error vector propagates backward through the network, the synaptic weights and bias vectors are adjusted with an error correction rule to minimize the error. Together, the applied input and desired output form a neural network *training set.*

The learning phase may be viewed as a nonlinear unconstrained parameter optimization problem. Depending upon the nature of the input, two types of algorithms are considered: deterministic and stochastic learning algorithms. With deterministic algorithms, the cost function is specific to the given training set. Networks trained this way tend to produce unexpected results when inputs are given that were not part of the training set. With stochastic algorithms, the cost function is the expected error for a given training set. Networks trained this way tend to be more robust to unknown inputs.

6.3.1 Deterministic Learning Algorithms

Define a learning cost function J as the mean squared error between the actual and desired output

$$J(w) = \sum_{k=1}^{N} \frac{e_k}{N}$$

where

$$e_k = (z_k - y_k)^T (z_k - y_k)$$

and where y_k is the network output for training set k, z_k is the desired output from training set k and N is the number of training sets. Recall that J depends on the network weight

and bias vectors.

A Davidon-Fletcher-Powell algorithm may be used to solve the unconstrained parameter optimization problem. For a quadratic cost with n parameters, the Davidon-Fletcher-Powell algorithm converges in n iterations. A rank two update for the Hessian matrix will ensure that the Hession is positive definite at the end of each iteration. A suitable test for convergence is to check whether the change in the Hessian matrix is small.

6.3.2 Stochastic Learning Algorithms

From Section 6.2.2, the problem of training a Bayesian neural network may be viewed as a nonlinear regression function minimization. Consider the cost $J(w)$, a function of the network weights and bias vectors, given by (6.2)

$$J(w) = E_{x,z}\left[\|z - y(w,x)\|^2\right]$$

Let

$$\begin{aligned}\phi(w) &\triangleq \|z - y(w,x)\|^2 \\ g(w) &\triangleq -2[z - y(w,x)]^T \frac{\partial y(w,x)}{\partial w}\end{aligned}$$

For the minimization problem $\min_w J(w)$, a necessary condition for a parameter vector w to be minimizing is that

$$\nabla J(w) = E_{x,z}[g(w)] = 0$$

Since both z and x are random variables, $g(w)$ is a noisy gradient of the cost. Samples of $\phi(w)$ and $g(w)$ are available for the minimization process.

The stochastic minimization $\min_w J(w)$, may be implemented with a Robbins-Munro algorithm. The algorithm is a variation of the steepest descent algorithm

$$w_{k+1} = w_k - \rho_k g(w_k)$$

where $\rho_k > 0$, $\sum \rho_k^2 < \infty$ and $\sum \rho_k = \infty$.

It can be shown that under the following three assumptions, the algorithm converges in the mean square sense, that is,

$$E[\|w_k - w_0\|^2] = 0 \qquad k \to \infty$$

1. $\phi(w)$ has a unique zero w_0, which is bounded.

2. $g(w)$ is linear near w_0.

3. The variance of $g(w)$ is bounded above by a quadratic function of w as in

$$E[\|g(w)\|^2] \leq h[1 + \|w - w_0\|^2] \qquad h > 0$$

Of course, from Chebyshev's inequality, the algorithm also converges with probability one.

An initial solution to the problem is found by removing the expectation operator and using the Davidon-Fletcher-Powell deterministic algorithm. This validates the first two assumptions for the multilayer perceptrons. By taking partial derivatives and exploiting the fact that multilayer perceptrons have sigmoidal activation functions, it is seen that the variance of $g(w)$ is always bounded. Thus all three assumptions hold for multilayer perceptrons and the stochastic training algorithm converges with probability one.

6.4 Bayesian Neural Networks as Residual Processors

The objective of a residual processor is not just to announce a fault but to provide an associated probability of false alarm. While, multilayer perceptrons have proved to be very successful in static pattern recognition problems (Haykin 1994), a recurrent Bayesian neural network can be shown to approximate the posterior probability of feature classification conditioned on an input history.

The residual processor designs described in this section are applied to the fault detection filters of (Douglas et al. 1995). These filters are used when the vehicle is operating at a nominal $27\frac{\text{m}}{\text{sec}}$ on a straight road so vehicle lateral dynamics are not considered. Four Bayesian neural network residual processors are designed, one for each fault detection filter.

A schematic of one network is provided in Figure 6.1. Each network has the following properties.

- Each has five layers: one input layer, three hidden layers and one output layer.
- A feedback loop is included where a one step delayed output is summed with the current input at the input layer.
- The activation function $S(\cdot)$ is a sigmoidal function. This function has a smooth nonlinearity which is useful for gradient calculations.
- Network connections are defined in (6.1) and are illustrated in Figure 6.1. All vectors are in R^3 except for the network associated with fault group four where the vectors are in R^2.

Figure 6.2 shows the residual processing scheme using Bayesian neural networks and the fault detection filters for the longitudinal simulation.

6.5 Simulation Results

Each Baysian neural network is trained to announce the probability of a fault in a particular sensor conditioned on the residual process. The training data for each network is obtained by simulating bias faults of some nominal size in the vehicle nonlinear simulation.

Two types of faults are considered for residual processor testing: step faults and ramp faults. Step faults are an abrupt change from a no fault situation to a nominally sized fault in a particular sensor. Step faults are considered in the pitch rate and air mass sensors. Since the Baysian neural networks are tested on the training set, no effort to generalize responses to unknown faults is made here.

Ramp faults correspond to a gradual, linear change from a no fault situation to a fault in a particular sensor. In contrast with the step faults, ramp faults necessarily represent fault sizes that have not been encountered by the Baysian neural networks in their respective training sets. These kinds of faults illustrate the generalization capability of the Baysian neural networks.

6.5.1 Step Faults

Figures 6.3, 6.4 and 6.5 each show one of the ouptuts of the Bayesian neural network for fault group three. This network analyzes the residuals from the fault detection filter which considers sensor faults for the pitch rate, forward symmetric wheel speed and the rear symmetric wheel speed sensors.

In each figure, no fault occurs from $t = 0$ to $t = 4$ sec. From $t = 4$ sec. onwards, step faults in different sensors and actuators are applied one at a time and in the following order:

- pitch rate sensor (T)
- front wheel speed sensor (FS)
- rear wheel speed sensor (RS)
- air mass sensor (M)
- engine speed sensor (W)
- longitudinal accelerometer (X)
- throttle actuator (A)
- brake torque actuator (Tb)

Note in the figures that there are two cases for the throttle fault. Figure 6.3 shows the posterior probability of a pitch rate sensor fault conditioned on the residual process. Figure 6.4 shows the posterior probability of a front wheel speed fault conditioned on the residual process. Figure 6.5 shows the posterior probability of a rear wheel speed sensor fault conditioned on the residual process.

Each figure shows that the Bayesian neural network gives a high probability of a fault when a fault occurs in the corresponding sensor or actuator and a low probability of a fault otherwise. Note that the residual process is nonzero when a fault occurs in any sensor apart from the sensors for which the filter is designed. Even though the residual is nonzero, the network correctly does not announce a fault.

6.5.2 Ramp Faults

In this section, ramp faults are considered in the pitch rate sensor, vertical accelerometer, longitudinal accelerometer and the air mass sensor.

Figures 6.6 and 6.7 show fault detection filter residuals and outputs of a Bayesian neural network for fault group three. In these figures, Z, T and RS denote the magnitudes of the vertical accelerometer, pitch rate and real wheel speed residuals and P(Z), P(T) and P(RS) denote the posterior probability of the corresponding fault conditioned on the residual process. Figures 6.8 and 6.9 show the same results but for fault group one. In these figures, M, W and X denote the magnitudes of the air mass sensor, engine speed sensor and longitudinal accelerometer residuals and P(M), P(W) and P(X) denote the posterior probability of the corresponding fault conditioned on the residual process. In each figure, no fault occurs from $t = 0$ to $t = 1$ sec. and from $t = 1$ sec. onwards, a ramp fault occurs.

- Figure 6.6 shows results when a ramp fault of size 0 to $0.5\frac{\text{rad}}{\text{sec}}$ occurs in the pitch rate sensor. Note that the Bayesian neural network has been trained with a nominal pitch rate sensor step fault of $0.05\frac{\text{rad}}{\text{sec}}$.

- Figure 6.7 shows results when a ramp fault of size 0 to $5\frac{\text{m}}{\text{sec}^2}$ occurs in the vertical accelerometer. The network has been trained with a nominal vertical accelerometer step fault of $0.5\frac{\text{m}}{\text{sec}^2}$.

- Figure 6.8 shows results when a ramp fault of size 0 to $1\frac{\text{m}}{\text{sec}^2}$ occurs in the longitudinal accelerometer. Training has been done with a nominal longitudinal accelerometer step fault of $0.1\frac{\text{m}}{\text{sec}^2}$.

- Figure 6.9 shows results when a ramp fault of size 0 to 0.14 kg. occurs in the air mass sensor. The network has been trained with a nominal air mass sensor step fault of 0.07 kg.

6.6 Discussion

At this stage, an interesting comparison may be made of the stochastic and deterministic training approachs. In the stochastic approach, the training sets are considered as sample sets which provide information about the entire population. In the deterministic approach, the training sets are the entire population hence no effort is made to generalize. The classes may intersect in the pattern space for the stochastic problem, while the deterministic approach theoretically considers mutually exclusive classes only.

From a theoretical perspective, when Bayesian neural networks are trained for pattern classification using the mean square criterion, their outputs are estimates of classification probabilities conditioned on the input. This conclusion is valid for *any* approach based on the minimization of the mean-squared error criterion. However, in theory, multilayer perceptrons can approximate any non-linear mapping (Lippmann 1987), hence, they are more likely to fit the posterior probabilities. The simulation studies conducted demonstrate the above assertion.

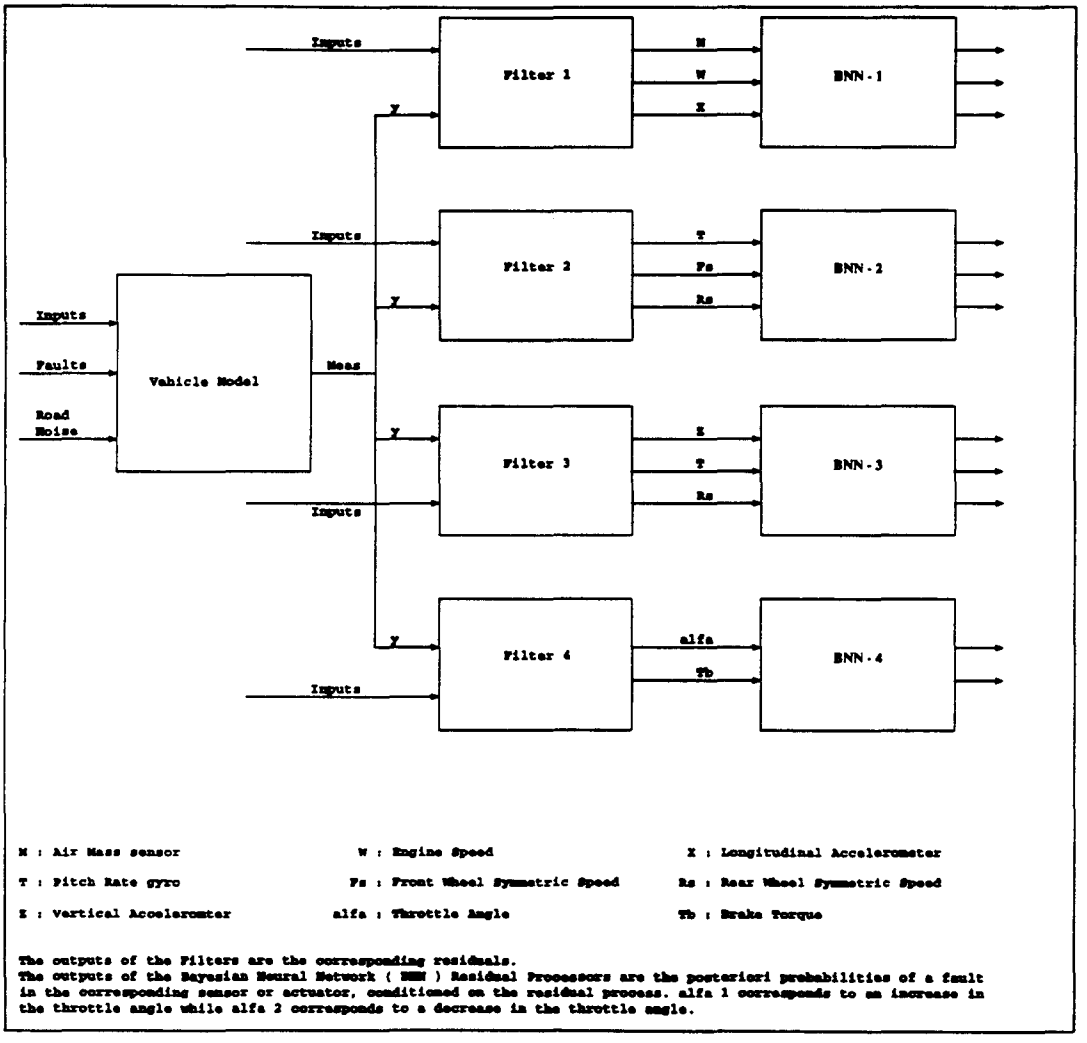

Figure 6.2: Residual processing scheme for the longitudinal simulation.

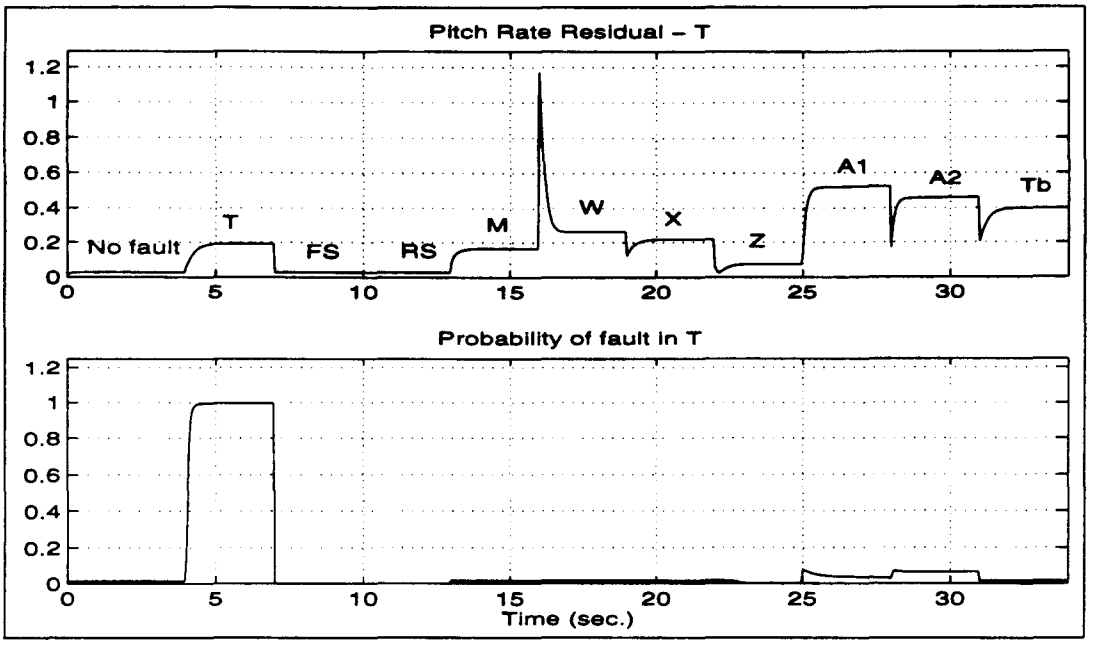

Figure 6.3: Posterior probability of a fault in the pitch rate sensor.

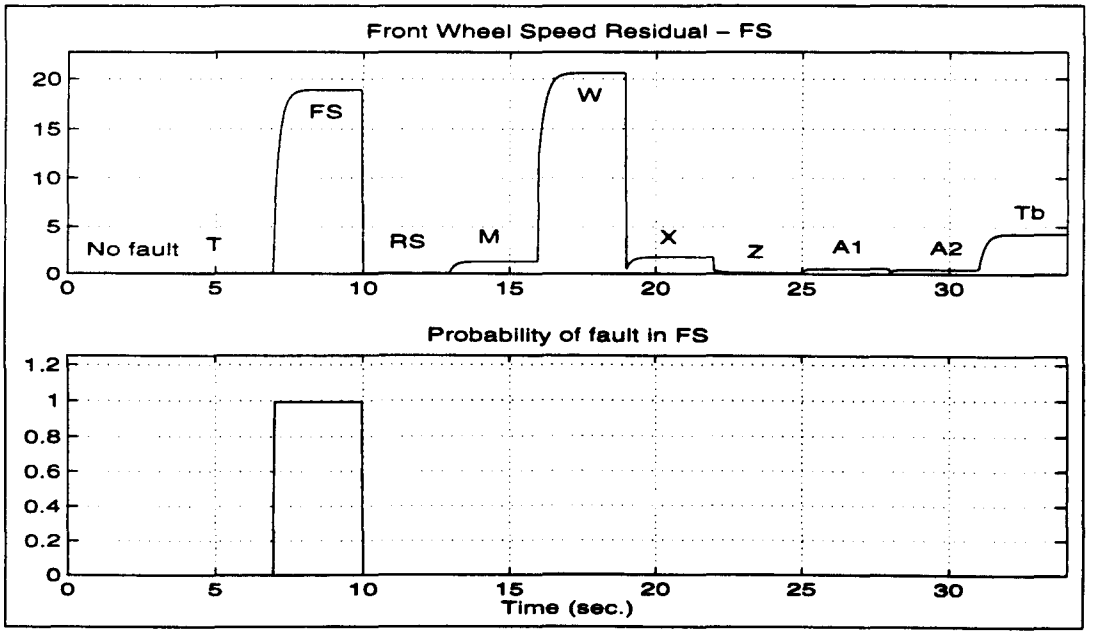

Figure 6.4: Posterior probability of a fault in the front wheel speed sensor.

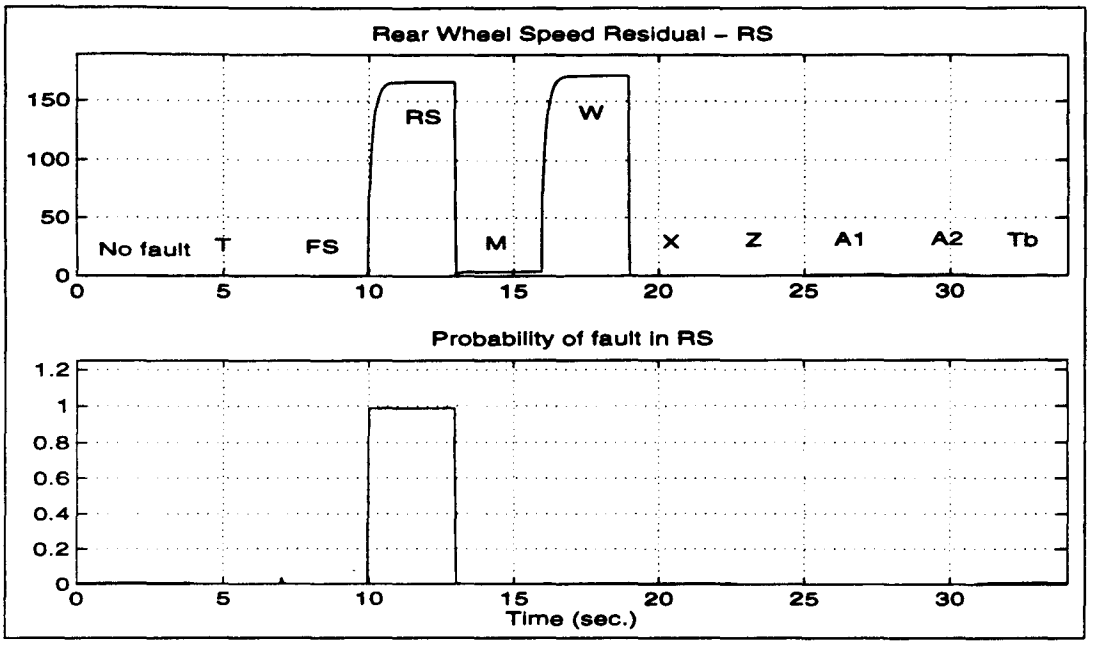

Figure 6.5: Posterior probability of a fault in the rear wheel speed sensor.

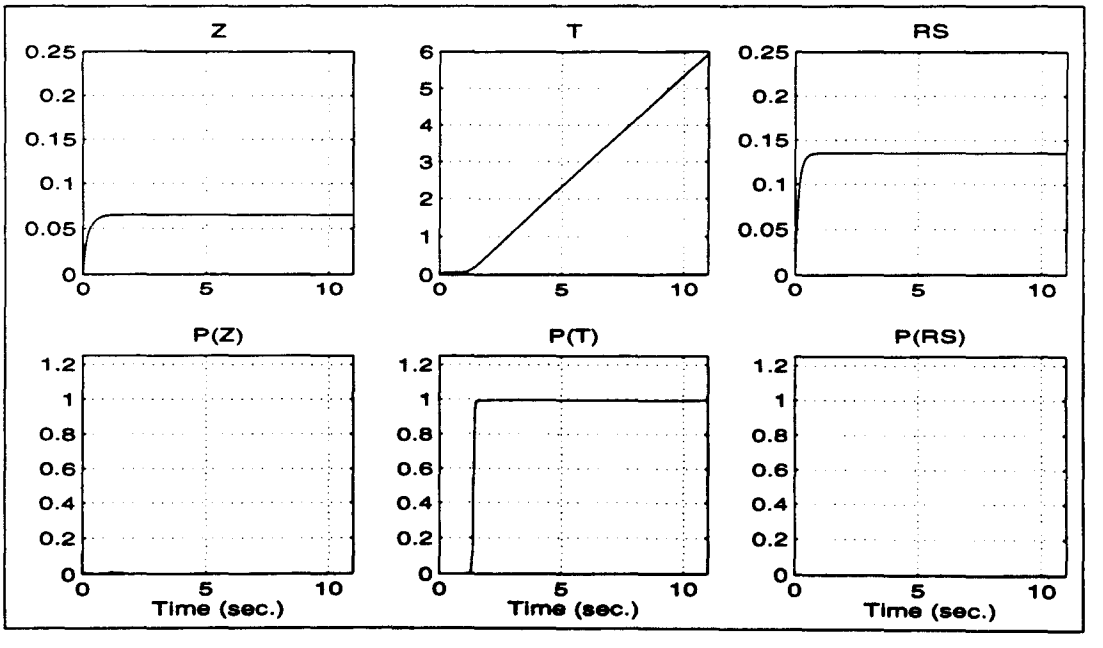

Figure 6.6: Ramp fault in pitch rate sensor.

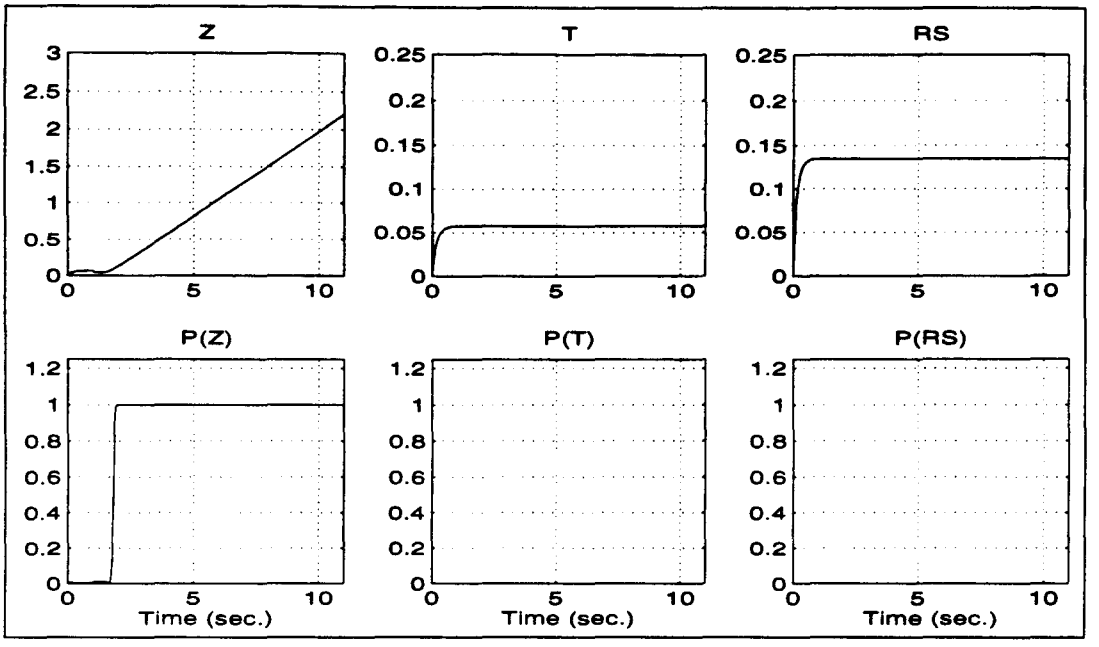

Figure 6.7: Ramp fault in vertical accelerometer.

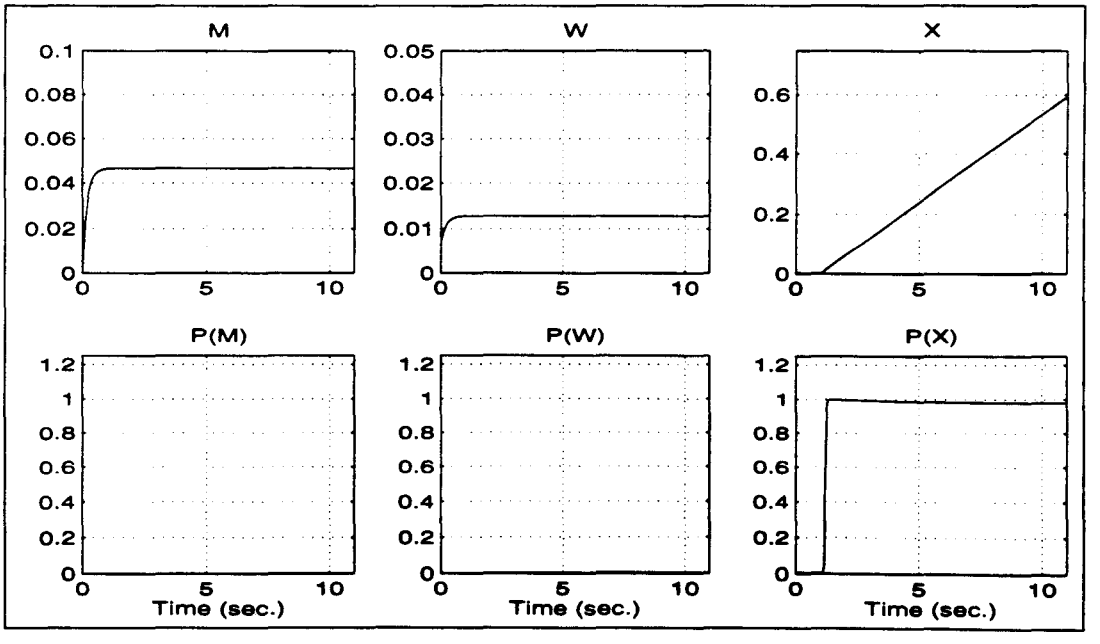

Figure 6.8: Ramp fault in longitudinal accelerometer.

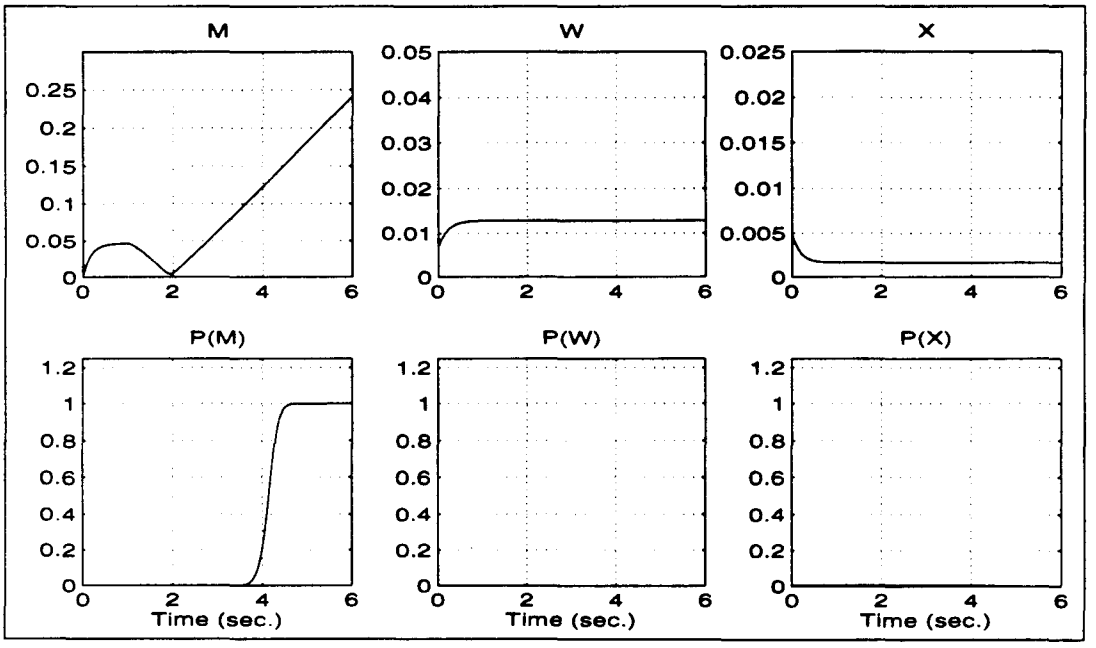

Figure 6.9: Ramp fault in air mass sensor.

CHAPTER 7
Sequential Probability Ratio Tests

THE RESIDUAL PROCESSING PROBLEM is considered in this section as a hypothesis detection and identification problem. Both Bayesian (Shiryayev 1977) and non-Bayesian approaches (Nikiforov 1995, Basseville and Nikiforov 1995) to the classical change detection problem have been developed. A binary hypothesis Shiryayev test, which is a Bayesian approach, is formulated by Speyer and White (Speyer and White 1984) as a dynamic programming problem. A similar approach, one also using a dynamic programming formulation, is taken here to derive an online multiple hypothesis Shiryayev Sequential Probability Ratio Test (SPRT).

It is shown that for a certain criterion of optimality, this extended Shiryayev SPRT detects and isolates the occurrence of a failure in a conditionally independent measurement sequence in minimum time. The algorithm is shown to be optimal even in the asymptotic sense and the theoretical results have been extended to the detection and identification of changes with unknown parameters. The dynamic programming analysis includes the measurement cost, the cost of a false alarm and the cost of a miss-alarm.

Note that with the Shiryayev SPRT, a change in the residual hypothesis is detected in minimum time. In contrast, the Wald SPRT detects the presence or absence of a failure in the entire measurement sequence. Here, the residual hypothesis is unknown but is assumed to be constant through the measurement sequence.

A non-Bayesian approach to the classical change detection problem is the Generalized CUMulative SUM (CUSUM) algorithm (Nikiforov 1995, Basseville and Nikiforov 1995). It has been shown that there exists a lower bound for the worst mean detection delay and that the CUSUM algorithm reaches this lower bound. This establishes the algorithms worst mean detection time minimax optimality.

Recently, the algorithm has been extended to solve the change detection and isolation problem (Nikiforov 1995). This extension is based on the log likelihood ratio between two hypotheses $\mathcal{H}_i$ and $\mathcal{H}_j$. When the difference between the log likelihood ratio and its current minimum value for a given hypotheses $\mathcal{H}_i$ and other hypotheses exceeds a chosen threshold, hypothesis $\mathcal{H}_i$ is announced. This implies that a hypothesis announcement requires that the recent measurements be significant enough to support the announcement.

Several important observations are made regarding the extended CUSUM algorithm.

- The algorithm is computationally intensive and is not recursive. If the number of hypotheses is m, the number of computations is of the order of m^2. This problem can be avoided by modifying the algorithm to compare all the hypotheses to the null hypothesis $\mathcal{H}_0$ while doing the computations. This modification would reduce the number of computations from the order of m^2 to m.

- No assumption is made about the apriori probability of change from hypothesis $\mathcal{H}_0$ to $\mathcal{H}_i$ from one measurement to the next. This probability is embedded explicitly in the Shiryayev SPRT.

- Unlike the Shiryayev SPRT, the posterior probability of a hypothesis change is not calculated in the CUSUM algorithm.

- Thresholds for hypothesis change announcements must be made apriori whereas in the Shiryayev SPRT, a methodology for interpreting the choice of the threshold is explicit.

- The CUSUM algorithm is similar to the Wald SPRT in that a finite size, sliding data window allows for changes in hypothesis to be detected but that the hypothsis essentially is assumed to be constant throughout the window.

This chapter is organized as follows. Notation is defined in Section 7.1. Section 7.2 has the main development of a multiple hypothesis Shiryayev sequential probability ratio test. First, a conditional probability propagation equation is developed. Next, a dynamic programming problem is defined and some of the asymptotic properties of the cost function are demonstrated. Next, a decision rule is defined by building thresholds. Finally, the test is generalized to the detection and isolation of changes with unknown parameters. In Section 7.3 a few illustrative examples are given and in Section 7.4, the algorithm is applied to a health monitoring system for automated vehicles using a high-fidelity nonlinear simulation. The performance of the algorithm is evaluated by implementing it in a fault detection and identification scheme in the longitudinal nonlinear vehicle simulation. Finally, in Section 7.5, a few comments are made about assumptions underlying the MHSSPRT.

7.1 Preliminaries and Notation

Let x_k be a measurement vector at time t_k and $\mathrm{X}_k \triangleq \{x_k\}$ be a conditionally independent measurement sequence. A fault is said to occur when there exists a discrete jump in the probabilistic description of X_k. The probabilistic description of X_k is assumed to be known both before and after a fault occurs. The fault hypotheses are enumerated as faults of type i with the total number of faults $m+1$ being fixed. The fault type 0 is also called the no-fault or null-fault hypothesis.

The probability density function of x_k in the no-fault or type i fault state is denoted $f_0(\cdot)$ or $f_i(\cdot)$. These probability density functions are constant so no subscript k is indicated.

However, note that in the following development, the density functions are not required to be constant.

During any time interval $t_k < t \leq t_{k+1}$, the probablility that the measurement sequence X_k will switch from a no-fault state to a type i fault state is known apriori and is denoted p_i. The time that the meaurement sequence switches from a no-fault state to a type i fault state is not known and is denoted θ_i.

The probability that a type i fault has occured before time t_0 is $\pi_i \triangleq P(\theta_i \leq t_0)$. The probability, conditioned on the measurement sequence X_k, that a type i fault has occured before time t_k is $F_{k,i} = P(\theta_i \leq t_k/\mathrm{X}_k)$. The above notation and definitions are summarized as follows.

$x_k \triangleq$ Measurement vector at time t_k.

$\mathrm{X}_k \triangleq$ Measurement history through t_k.

$m \triangleq$ Number of fault types.

$f_0(\cdot) \triangleq$ Probability density function of x_k under no-fault hypothesis.

$f_i(\cdot) \triangleq$ Probability density function of x_k under type i fault hypothesis.

$p_i \triangleq$ Apriori probability of change from no-fault to type i fault for $t_k < t \leq t_{k+1}$.

$\theta_i \triangleq$ Time of type i fault.

$\pi_i \triangleq P(\theta_i \leq t_0)$.

$F_{k,i} \triangleq P(\theta_i \leq t_k/\mathrm{X}_k)$

7.2 Development of a Multiple Hypothesis Shiryayev SPRT

An extension of the Shiryayev sequential probability ratio test to allow multiple hypotheses is as follows. First, a conditional probability propagation equation is developed. Next, a dynamic programming problem is defined and some of the asymptotic properties of the cost function are demonstrated. Next, a decision rule is defined by building thresholds. Finally, the test is generalized to the detection and isolation of changes with unknown parameters.

7.2.1 Recursive Relation for the Posteriori Probability

The results of this section are encapsulated in two propositions. The first proposition provides a recurrsive update for $F_{k,i}$, the conditional probability that a type i fault has occured. The second proposition shows that $F_{k,i}$, as given by the recurrsion, is consistent with the definition of a probability.

Proposition 7.1. A recursive update formula for $F_{k,i}$ is

$$F_{0,i} = \pi_i \tag{7.1a}$$

$$F_{k+1,i} = \frac{M_{k,i} f_i(x_{k+1})}{(\sum_{i=1}^{m} M_{k,i})\, f_i(x_{k+1}) + (1 - \sum_{i=1}^{m} M_{k,i})\, f_0(x_{k+1})} \tag{7.1b}$$

where

$$M_{k,i} = F_{k,i} + p_i(1 - F_{k,i}) \tag{7.1c}$$

Proof. The proof is done by induction. The probability $F_{1,i}$ that a type i fault has occured before t_1 given a meausurement x_1 is given by Bayes' rule as

$$P(\theta_i \leq t_1 / x_1) = \frac{P(x_1/\theta_i \leq t_1) P(\theta_i \leq t_1)}{P(x_1)} \tag{7.2}$$

where

$$P(x_1) = \sum_{i=1}^{m} [P(x_1/\theta_i \leq t_1) P(\theta_i \leq t_1) + P(x_1/\theta_i > t_1) P(\theta_i > t_1)] \tag{7.3a}$$

$$\begin{aligned} P(\theta_i \leq t_1) &= P(\theta_i \leq t_0) + P(t_0 < \theta_i \leq t_1) \\ &= \pi_i + p_i(1 - \pi_i) \end{aligned} \tag{7.3b}$$

$$P(x_1/\theta_i > t_1) = f_0(x_1) dx_1 \tag{7.3c}$$

$$P(x_1/\theta_i \leq t_1) = f_i(x_1) dx_1 \tag{7.3d}$$

$$\sum_{i=1}^{m} P(\theta_i > t_1) = 1 - \sum_{i=1}^{m} P(\theta_i \leq t_1) \tag{7.3e}$$

Strictly, (7.3d) denotes the probability that the measurement lies between x_1 and $x_1 + dx_1$ given the occurrence of a type i fault at or before t_1. Expanding (7.2) with the identities

of (7.3) produces the fault probability $F_{1,i}$.

$$F_{1,i} = \frac{[\pi_i + p_i(1-\pi_i)]f_i(x_1)}{\sum_{i=1}^{m}[\pi_i + p_i(1-\pi_i)]f_i(x_1) + (1 - \sum_{i=1}^{m}[\pi_i + p_i(1-\pi_i)])f_0(x_1)} \tag{7.4}$$

The probability $F_{k,i}$, conditioned on a measurement sequence $mathrmX_k$, that a type i fault has occured before t_k is given by Bayes' rule as

$$P(\theta_i \leq t_{k+1}/\mathrm{X}_{k+1}) = \frac{P(\mathrm{X}_{k+1}/\theta_i \leq t_{k+1})P(\theta_i \leq t_{k+1})}{\mathrm{X}_{k+1}}$$

Since the measurement sequence is conditionally independent, this expands to

$$P(\theta_i \leq t_{k+1}/\mathrm{X}_{k+1}) = \frac{P(x_{k+1}/\theta_i \leq t_{k+1})P(\mathrm{X}_k/\theta_i \leq t_{k+1})P(\theta_i \leq t_{k+1})}{P(\mathrm{X}_{k+1})}$$

and finally to

$$P(\theta_i \leq t_{k+1}/\mathrm{X}_{k+1}) = \frac{P(x_{k+1}/\theta_i \leq t_{k+1})P(\theta_i \leq t_{k+1}/\mathrm{X}_k)}{P(x_{k+1}/\mathrm{X}_k)} \tag{7.5}$$

which follows from the identity

$$P(\mathrm{X}_{k+1}) = P(x_{k+1}/\mathrm{X}_k)P(\mathrm{X}_k)$$

Now, consider the following identities

$$P(x_{k+1}/\mathrm{X}_k) = \sum_{i=1}^{m}[P(x_{k+1}/\theta_i \leq t_{k+1})P(\theta_i \leq t_{k+1}/\mathrm{X}_k) + P(x_{k+1}/\theta_i > t_{k+1})P(\theta_i > t_{k+1}/\mathrm{X}_k)] \tag{7.6a}$$

$$P(\theta_i \leq t_{k+1}/\mathrm{X}_k) = P(\theta_i \leq t_k/\mathrm{X}_k) + P(t_k < \theta_i <= t_{k+1}/\mathrm{X}_k) \tag{7.6b}$$

$$= F_{k,i} + p_i(1 - F_{k,i}) \tag{7.6c}$$

$$P(x_{k+1}/\theta_i > t_{k+1}) = f_0(x_{k+1})dx_{k+1} \tag{7.6d}$$

$$P(x_{k+1}/\theta_i \leq t_{k+1}) = f_i(x_{k+1})dx_{k+1} \tag{7.6e}$$

$$\sum_{i=1}^{m} P(\theta_i > t_{k+1}/\mathrm{X}_k) = 1 - \sum_{i=1}^{m} P(\theta_i \leq t_{k+1}/\mathrm{X}_k) \tag{7.6f}$$

Expanding (7.5) with the identities of (7.6) produces the fault probability $F_{k+1,i}$.

$$F_{k+1,i} = \frac{M_{k,i}f_i(x_{k+1})}{(\sum_{i=1}^{m} M_{k,i})\, f_i(x_{k+1}) + (1 - \sum_{i=1}^{m} M_{k,i})\, f_0(x_{k+1})} \tag{7.7}$$

where $M_{k,i}$ is defined in (7.1c) Relations (7.4) and (7.7) together prove the induction. ●

The following proposition states that a simple requirement on the initial conditions ensures that the $F_{k,i}$ are consistent with the definition of a probability

Proposition 7.2. The condition $\sum_{i=1}^{m} \pi_i \leq 1$ implies that

$$0 \leq F_{k,i} \leq 1 \qquad \forall\, k$$

and

$$\sum_{i=1}^{m} F_{k,i} \leq 1 \qquad \forall\, k$$

Proof. The proof follows as a direct application of the recursion (7.1).

Note that $F_{k,0} = 1 - \sum_{i=1}^{m} F_{k,i}$. Finally, note that (7.1) reduces to a multiple hypothesis Wald SPRT if $p_i = 0 \ \forall i$.

7.2.2 Dynamic Programming Formulation

At each time t_k one of two actions are possible:

1. Terminate the measurement sequence and announce a fault of type i. The cost of making a correct announcement is zero while the cost of a false alarm of type i is Q_i.

2. Take another measurement. The cost of the measurement is C and the cost of a miss-alarm of type i is S_i.

An optimal decision algorithm is derived by minimizing the expected cost at a time t_N. Suppose N measurements are taken and that at time t_N, a type i fault is announced. Assuming further that only one fault may have occurred, the cost is

$$J_{N,i} \;=\; (1 - F_{N,i})Q_i$$

so the optimal cost at t_N is

$$J_N^* \;=\; \min_i (1 - F_{N,i})Q_i \tag{7.8}$$

The expected cost at time t_{N-1} is

$$J_{N-1,i} = \min\left[(1 - F_{N-1,i})Q_i,\ C + S_i F_{N-1,i} + E_{x_N}[J_N^*/\mathrm{X}_{N-1}]\right]$$

and the optimal cost at t_{N-1} is

$$J_{N-1}^* = \min_i \min\left[(1 - F_{N-1,i})Q_i,\ C + S_i F_{N-1,i} + E_{x_N}[J_N^*/\mathrm{X}_{N-1}]\right]$$

In general, the optimal expected cost at time t_k is

$$J_k^* = \min_i \min\left[(1 - F_{k,i})Q_i,\ C + S_i F_{k,i} + A_k(F_k)\right]$$

where

$$A_k(F_k) \triangleq E_{x_{k+1}}\left[J_{k+1}^*/\mathrm{X}_k\right] \tag{7.9a}$$

$$F_k \triangleq [F_{k,1}, F_{k,2}, \ldots, F_{k,m}]^T \tag{7.9b}$$

The expectation is taken with respect to the conditional probability density functions $f_i(x_{k+1}/\mathrm{X}_k)$.

The optimal policy, one that minimizes the expected cost at each time t_k, is stated with respect to a threshold probability $F_{T_k,i}$:

- If $F_{k,i} \geq F_{T_k,i}$, announce a type i fault.
- If $F_{k,i} < F_{T_k,i}$ for each $i \in \{1, \ldots, m\}$, take another measurement.

The threshold probability $F_{T_k,i}$ is determined at each time t_k as the value at which the expected cost of terminating the test by announcing a fault, and possibly a false alarm, is the same as the expected cost of continuing the test by taking another measurement.

$$(1 - F_{T_k,i})Q_i = C + A_k(F_{T_k}) + S_i F_{T_k,i} \tag{7.10a}$$

with

$$Q_i > C + A_k(0) \tag{7.10b}$$

Unfortunately, determining the threshold probabilities $F_{T_k,i}$ is a numerically intractable problem, even in the scalar case where $m = 1$. This is because the $A_k(F_{T_k})$ expectations are evaluated with respect to the conditional probability density functions $f_i(x_{k+1}/\mathrm{X}_k)$ or (7.6a) in the proof of Proposition 7.1,

$$P(x_{k+1}/\mathrm{X}_k) = \sum_{i=1}^{m} [P(x_{k+1}/\theta_i \leq t_{k+1})P(\theta_i \leq t_{k+1}/\mathrm{X}_k) + P(x_{k+1}/\theta_i > t_{k+1})P(\theta_i > t_{k+1}/\mathrm{X}_k)]$$

The following lemma establishes properties of $A_k(F_{T_k})$ which allow for a tractable policy, one which is optimal in the limit as $(N-k) \to \infty$.

Lemma 7.3. The functions $A_k(F_k)$ satisfy the following properties $\forall k \in \{1, \ldots, m\}$

1. If $\pi_i = 1$ for any $1 \leq i \leq m$, then $A_k(F_k) = 0$
2. $A_k(F_k) \leq A_{k-1}(F_{k-1})$
3. $A_k(F_k)$ is concave

Proof.

Property 1: Note that by the recursion relation (7.1) of Proposition 7.1, $\pi_i = 1 \Rightarrow F_{1,i} = 1$ and $F_{k,i} = 1 \Rightarrow F_{k+1,i} = 1$. By induction, $\pi_i = 1 \Rightarrow F_{k,i} = 1$, $\forall k \in \{1, \ldots, N\}$. Also, note that by Proposition 7.2, $\pi_i = 1 \Rightarrow F_{k,j} = 0$ for $j \neq i$ and $\forall k \in \{1, \ldots, N\}$.

Suppose $\pi_i = 1$ for some i. By the definition of $A_k(F_k)$

$$A_{N-1}(F_{N-1}) = E_{x_N}[J_N^*/\mathrm{X}_{N-1}] \tag{7.11a}$$

$$= E_{x_N}[\min_i (1 - F_{N,i})Q_i/\mathrm{X}_{N-1}] \tag{7.11b}$$

$$= 0 \tag{7.11c}$$

since $\pi_i = 1 \Rightarrow F_{N-1,i} = 1$. Now, suppose $A_k(F_k) = 0$ where again $\pi_i = 1$. Then

$$A_{k-1}(F_{k-1}) = E_{x_k}[J_k^*/\mathrm{X}_{k-1}] \tag{7.12a}$$

$$= E_{x_k}\left[\min_i \min\left[(1 - F_{k,i})Q_i,\ C + S_i F_{k,i} + A_k(F_k)\right]/\mathrm{X}_{k-1}\right] \tag{7.12b}$$

$$= 0 \tag{7.12c}$$

Relations (7.11) and (7.12) prove property 1 by induction.

Property 2: By the definition of $A_{k-1}(F_{k-1})$,

$$\begin{aligned} A_{k-1}(F_{k-1}) &= E_{x_k}[J_k^*/\mathrm{X}_{k-1}] \\ &= E_{x_k}\left[\min_i \min\left[(1-F_{k,i})Q_i,\ C+S_iF_{k,i}+A_k(F_k)\right]/\mathrm{X}_{k-1}\right] \end{aligned}$$

Since the test terminates at time t_N, it must happen that for the minimizing i,

$$\min\left[(1-F_{k,i})Q_i,\ C+S_iF_{k,i}+A_k(F_k)\right] = C+S_iF_{k,i}+A_k(F_k)$$

So,

$$\begin{aligned} A_{k-1}(F_{k-1}) &= E_{x_k}\left[\min_i \left(C+S_iF_{k,i}+A_k(F_k)\right)/\mathrm{X}_{k-1}\right] \\ &= C+\min_i S_iF_{k,i}+E_{x_k}\left[A_k(F_k)/\mathrm{X}_{k-1}\right] \end{aligned}$$

Therefore,

$$A_{k-1}(F_{k-1}) \geq A_k(F_k)$$

Property 3: Now show that $A_k(\cdot)$ is concave. By inspection of (7.8) and (7.9), J_N^* is concave. Since the test ends at t_N:

$$J_{N-1}^* = C + A_{N-1} + \min_i S_iF_{N-1,i} \tag{7.13}$$

Clearly, J_{N-1}^* is concave if A_{N-1} is concave. Let the elements of the countably infinite measurement space be denoted by x_k^j where $j = 1, 2, \ldots, \infty$ and $k = 1, 2, \ldots, N$. From (7.9) and (7.6a) :

$$\begin{aligned} A_k &= \sum_{j=1}^{\infty}\left[\left(\sum_{i=1}^{m} M_{k,i}\right) f_i(x_{k+1}^j) + \left(1-\sum_{i=1}^{m} M_{k,i}\right) f_0(x_{k+1}^j)\right] J_{k+1}^*(F_{k+1}) \\ &= \sum_{j=1}^{\infty}\sum_{i=1}^{m} h_i^j(F_k) \\ &= \sum_{j=1}^{\infty} h^j(F_k) \end{aligned} \tag{7.14}$$

where

$$h_i^j = \left(M_{k,i}\left[f_i(x_{k+1}^j) - f_0(x_{k+1}^j)\right] + \frac{1}{m} f_0(x_{k+1}^j)\right) J_{k+1}^*(F_{k+1})$$

and where

$$M_{k,i} = F_{k,i} + p_i(1 - F_{k,i})$$

If each of the h_i^j is concave, the summation is concave. Therefore, it remains to show that

$$h^j\left(\lambda F_k^1 + (1-\lambda)F_k^2\right) \geq \lambda h^j(F_k^1) + (1-\lambda)h^j(F_k^2) \tag{7.15}$$

where $\lambda,\ F_k^1,\ F_k^2\ \in [0,1]$. Define

$$\xi_{k,i}^r \triangleq \left[F_{k,i}^r + p_i(1 - F_{k,i}^r)\right]\left[f_i(x_{k+1}^j) - f_0(x_{k+1}^j)\right] + \frac{1}{m} f_0(x_{k+1}^j) \qquad \text{for } r = 1, 2.$$

so that the convexity inequality (7.15) becomes

$$\left[\lambda\xi_{k,i}^1 + (1-\lambda)\xi_{k,i}^2\right] J_{k+1}^*(\widehat{F}_{k+1}) \geq \lambda\xi_{k,i}^1 J_{k+1}^*(F_{k+1}^1) + (1-\lambda)\xi_{k,i}^2 J_{k+1}^*(F_{k+1}^2) \tag{7.16}$$

where $\widehat{F}_{k+1} = \widehat{F}_{k+1}(\lambda, F_k^1, F_k^2)$. Now,

$$F_{k+1,i}^{1,2} = \frac{M_{k,i}^{1,2} f_i(x_{k+1})}{\sum_{s=1}^m \xi_{k,s}^{1,2}} \tag{7.17}$$

$$\widehat{F}_{k+1,i} = \frac{\left[\lambda M_{k,i}^1 + (1-\lambda)M_{k,i}^2\right] f_i(x_{k+1})}{\sum_{s=1}^m \lambda\xi_{k,s}^1 + (1-\lambda)\xi_{k,s}^2} \tag{7.18}$$

Let $\tilde{\xi}_k^{1,2} = \sum_{s=1}^m \xi_{k,s}^{1,2}$. Then, from (7.17) and (7.18)

$$\widehat{F}_{k+1,i} = \frac{\lambda F_{k+1,i}^1 \tilde{\xi}_k^1 + (1-\lambda)F_{k+1,i}^2 \tilde{\xi}_k^2}{\lambda\tilde{\xi}_k^1 + (1-\lambda)\tilde{\xi}_k^2}$$

Take a summation from $i = 1, \ldots, m$ in (7.16) to get

$$J_{k+1}^*\left[\frac{\lambda\tilde{\xi}_k^1 F_{k+1}^1 + (1-\lambda)\tilde{\xi}_k^2 F_{k+1}^2}{\lambda\tilde{\xi}_k^1 + (1-\lambda)\tilde{\xi}_k^2}\right] \geq \frac{\lambda\tilde{\xi}_k^1}{\lambda\tilde{\xi}_k^1 + (1-\lambda)\tilde{\xi}_k^2} J_{k+1}^*(F_{k+1}^1) + \frac{(1-\lambda)\tilde{\xi}_k^2}{\lambda\tilde{\xi}_k^1 + (1-\lambda)\tilde{\xi}_k^2} J_{k+1}^*(F_{k+1}^2) \tag{7.19}$$

From (7.8) and (7.9), J_N^* is concave and hence satisfies (7.19). This implies that A_{N-1} is concave. But from (7.13), J_{N-1}^* is concave if A_{N-1} is concave. Hence, by induction, $A_k(\cdot)$ are concave $\forall k$. ◀

7.2.3 Thresholds for the Optimal policy

Lemma 7.3 showed that the $A_k(F_k)$ are monotonically decreasing in k and bounded because of the concavity property in F. This implies that for an infinite number of stages, that is, $(N-k) \rightarrow \infty$, each threshold probability $F_{T_k,i}$ also approaches a limit. To see this, rearrange (7.10) as

$$F_{T_k,i} = \frac{Q_i - C - A_k(F_{T_k})}{Q_i + S_i}$$

Then,

$$F_{T_k,i} \leq F_{T_{k-1},i} \leq \ldots \leq F_{T,i} \leq \frac{Q_i - C}{Q_i + S_i}$$

The dynamic programming algorithm for infinite time reduces to

$$J^*(F) = \min_i \min[(1 - F_{T,i})Q_i,\ C + S_i F_{T,i} + A(F_T)] \tag{7.20}$$

where

$$A(F) =$$

and the threshold probabilities $F_{T,i}$ are determined by

$$(1 - F_{T,i})Q_i = C + S_i F_{T,i} + A(F_T) \tag{7.21}$$

Since $A(F_T)$ is still hard to evaluate, a workaround is proposed. The idea is to choose the $F_{T,i}$ where

$$\alpha_i \triangleq 1 - F_{T,i}$$

are interpreted as false alarm rates and imply unknown Q_i, S_i and C through (7.21). In the context of (Q_i, S_i, C), the Shiryayev SPRT, extended here to multiple hypotheses, gives the minimum stoping time out of the set of stopping times $\{\tau_i\}$. This comes from an interpretation of the dynamic programming algorithm for infinite time (7.20) as a Bayes' risk minimizing cost (Shiryayev 1977)

$$J^*(F) = \min_i \inf_{\tau_i} E\left[(1 - F_{\tau_i,i})Q_i + F_{\tau_i,i}S_i + C\max\{\tau_i - \theta_i, 0\}\right]$$

Here, $E[1 - F_{\tau_i,i}]$ is the expected type i false alarm probability and $E[\max\{\tau_i - \theta_i, 0\}]$ is the expected delay of detecting a type i fault correctly.

The optimization problem is to minimize the mean time of delay in announcing a type i fault subject to the constraint that the probability of false alarm $\alpha_i = 1 - F_{\tau,i}$ is fixed at $\alpha_i = 1 - F_{T,i}$. The quantity $(\frac{S_i - Q_i}{C})$ becomes the Lagrange multiplier.

In the binary hypothesis case, $m = 1$, the Shiryayev SPRT policy is often expressed in a likelihood ratio form (Speyer and White 1984). This allows for an easy comparison with the Generalized Likelihood Ratio Test of (Nikiforov 1995),(Basseville and Nikiforov 1995). Define the likelihood ratio L_k, where the i subscript is dropped because there are only two hypotheses,

$$L_k = \frac{F_k}{1 - F_k}$$

and use (7.1) of Proposition 7.1 to develop a recursion relation

$$\begin{aligned} L_0 &= \frac{\pi}{1 - \pi} \\ L_{k+1} &= \left[\frac{f_1(x_{k+1})}{f_0(x_{k+1})}\right]\left(\frac{L_k + p}{1 - p}\right) \end{aligned}$$

Given a threshold likelihood ratio,

$$L_T = \frac{F_T}{1 - F_T}$$

the fault announcement policy becomes

- If $L_k \geq L_T$, announce that a fault has occured.
- If $L_k < L_T$, take another measurement.

Liklihood ratios can be defined in the obvious way for the multiple hypotheses case $m > 1$

$$\begin{aligned} L_{0,i} &= \frac{\pi_i}{1 - \pi_i} \\ L_{k,i} &= \frac{F_{k,i}}{1 - F_{k,i}} \end{aligned}$$

but no simple recursion relation can be developed from (7.1) to propagate the liklihood ratios.

7.2.4 Detection of Unknown Changes

The Shiryayev SPRT and the multiple hypotheses generalization, as described above, are developed for measurement sequences with known probability density functions, known both before and after a fault. It is an easy extension to allow the density functions to depend on a scalar unknown parameter α. Assume that the unknown parameter is also a random variable defined over a set Ω and has probability density function $\psi_\alpha(\Omega)$. Then, the conditional density function of the measurement sequence becomes

$$\begin{aligned} \tilde{f}_i(x) &\triangleq f(x/\mathcal{H}_i) \\ &= \int_\Omega f(x/\mathcal{H}_i, \eta)\ \psi_\alpha(\eta) d\eta \end{aligned} \tag{7.22}$$

Now, replace $f_i(\cdot)$ with the new density function $\tilde{f}_i(\cdot)$ in the recursive relation (7.1). The rest of the analysis remains the same.

7.3 Examples

Before considering the development of a residual processing module for Advanced Vehicle Control Systems, two examples are considered to illustrate the application of a multiple hypothesis Shiryayev SPRT. In the first example, the measurement sequence is taken as a white noise sequence with one of five possible means. In the second example, the measurement sequence is modeled as a scalar white noise sequence with unit power spectral density however, the mean is unknown.

7.3.1 Example 1

Here, the measurement sequence is modeled as a scalar white noise sequence with unit power spectral density and one of five possible means. The five hypotheses including the null hypothesis are summarized as

$$\mathcal{H}_i: \quad x \sim \mathcal{N}(0.5i, 1) \qquad \text{where } i \in \{0, 1, 2, 3, 4\}$$

For example, introduction of a bias with unit magnitude means the measurement sequence switches from the state $\mathcal{H}_0$ to $\mathcal{H}_2$. Extension to the case of vector valued measurements is trivial.

A simulated white noise measurement sequence is illustrated in Figure 7.1. Each measurement has a Gaussian distribution with unit variance and is uncorrelated with other meaurements. During the interval $0 \leq t < 1$ the measurements have zero mean. This is hypothesis $\mathcal{H}_0$. During the interval $1 \leq t \leq 5$ a unit bias is introduced so at $t = 1$, the measurement sequence switches from $\mathcal{H}_0$ to $\mathcal{H}_2$. The posteriori probabilities found from the recursion relation (7.1) and illustrated in Figure 7.1 very clearly show the measurement hypothesis switch. The apriori probabilities π_i are taken as 0.001 for $i \in \{1, 2, 3, 4\}$.

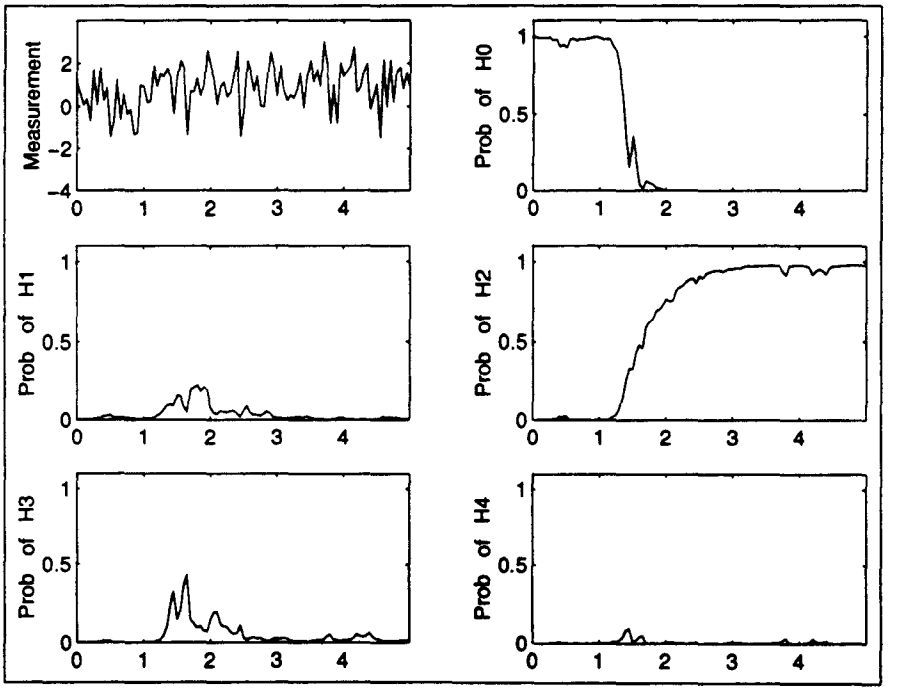

Figure 7.1: Change from $\mathcal{H}_0$ to $\mathcal{H}_2$ at time t = 1 sec.

7.3.2 Example 2

Again, the measurement sequence is modeled as a scalar white noise sequence with unit power spectral density. However, here the mean is also taken as random variable with one

of five possible uniform distributions. The five hypotheses including the null hypothesis are summarized as

$$\mathcal{H}_i : x \sim \mathcal{N}(m_i, 1)$$

where

$$\begin{aligned} m_0 &= 0 \\ m_1 &\sim \text{Unif } [0,\ 1] \\ m_2 &\sim \text{Unif } [0.5,\ 1.5] \\ m_3 &\sim \text{Unif } [1,\ 2] \\ m_4 &\sim \text{Unif } [1.5,\ 2.5] \end{aligned}$$

Following are two propositions that provide relations for normally distributed random variables with unknown means. The first proposition shows that if the measurement means have a Gaussian distribution, the problem reduces to one in which the measurement means are known and the covariances take on a larger value.

Proposition 7.4. Consider a vector valued random variable $x \in \mathrm{R}^n$ where both the mean and the distribution about the mean are Gaussian

$$\begin{array}{rcll} x & \sim & \mathcal{N}(m, \Lambda_x) & mathrmwherem \in \mathrm{R}^n, \quad \Lambda_x \in \mathrm{R}^{n \times n} \\ m & \sim & \mathcal{N}(m^*, \Lambda_m) & \text{where} m^* \in \mathrm{R}^n, \quad \Lambda_m \in \mathrm{R}^{n \times n} \end{array}$$

Then

$$x \sim \mathcal{N}(m^*, \Lambda_x + \Lambda_m)$$

Proof. A proof is provided at the end of this section.

The second proposition provides a probability density function for a Gaussian random variable where the mean has a uniform distribution.

Proposition 7.5. Consider a vector valued random variable $x \in \mathrm{R}^n$ where the mean has a uniform distribution and where the distribution about the mean is Gaussian

$$\begin{array}{rcll} x & \sim & \mathcal{N}(m, \Lambda_x) & \text{where} m \in \mathrm{R}^n, \quad \Lambda_x \in \mathrm{R}^{n\times n} \\ m & \sim & \mathrm{Unif}[b, b+2m^*] & \text{where} b,\ m^* \in \mathrm{R}^n \end{array}$$

Then the probability density function $f(x)$ is

$$f(x) = \frac{1}{4^n \Pi_j m_j^*}\left[\mathrm{erf}\{\frac{1}{\sqrt{2}}\Lambda_x^{-0.5}(x-b)\} - \mathrm{erf}\{\frac{1}{\sqrt{2}}\Lambda_x^{-0.5}(x-b-2m^*)\}\right]$$

where

$$m_j^* = [m_1^*, \ldots, m_n^*]^T$$

Note that a property of the error function $\mathrm{erf}(x)$ is that for $x \in \mathrm{R}^n$

$$\mathrm{erf}(x) = \mathrm{erf}(x_1)\,\mathrm{erf}(x_2)\cdots\mathrm{erf}(x_n)$$

Proof. A proof is provided at the end of this section.

A simulated measurement sequence is illustrated in Figure 7.2. Each measurement has a Gaussian distribution with unit variance and is uncorrelated with other meaurements. During the interval $0 \le t < 1$ the measurements have zero mean. This is hypothesis $\mathcal{H}_0$. During the interval $1 \le t \le 5$ a constant but unknown bias with uniform distribution Unif[0.5, 1.5] is introduced. Thus, at $t = 1$, the measurement sequence switches from $\mathcal{H}_0$ to $\mathcal{H}_2$. The measurements were generated as

$$x = n + s$$

where

$$\begin{array}{rcl} n & \sim & \mathcal{N}(0,\ 1) \\ s & = & \begin{cases} 0 & 0 \le t < 1 \\ \mathrm{Unif}[0.5,\ 1.5] & 1 \le t \le 5 \end{cases} \end{array}$$

As in the previous example, the posteriori probabilities found from the recursion relation (7.1) and illustrated in Figure 7.2 very clearly show the measurement hypothesis switch. Again, the apriori probabilities π_i are taken as 0.001 for $i \in \{1, 2, 3, 4\}$.

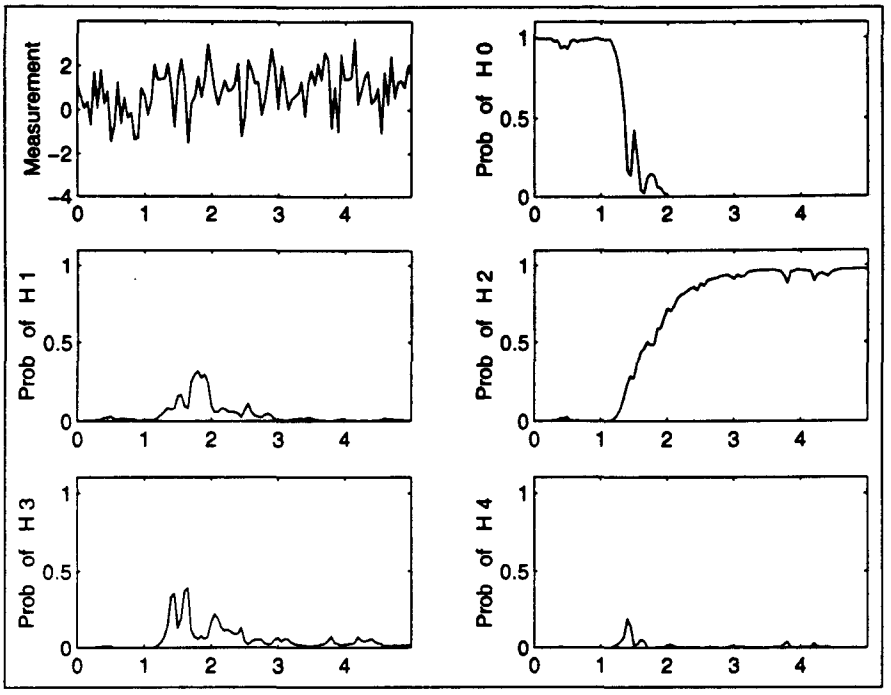

Figure 7.2: Change from $\mathcal{H}_0$to $\mathcal{H}_2$at time t = 1 sec.

Proof. (Of Proposition 7.4) From (7.22)

$$\begin{aligned} f(x/\mathcal{H}_i) &= \int_{\mathrm{R}^n} f(x/\mathcal{H}_i, m_i)\psi(m_i)dm_i \\ &= \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} \frac{1}{2\pi^n|\Lambda_{x_i}\Lambda_{m_i}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\|x-m_i\|^2_{\Lambda^{-1}_{x_i}} + \frac{1}{2}\|m_i-m_i^*\|^2_{\Lambda^{-1}_{m_i}}\right\}|dm_i| \\ &= \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} \frac{1}{2\pi^n|\Lambda_{x_i}\Lambda_{m_i}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}D\right\}|dm_i| \end{aligned} \tag{7.23}$$

Define

$$C_1 = \Lambda^{-1}_{x_i} + \Lambda^{-1}_{m_i} \tag{7.24a}$$

$$C_2 = \Lambda^{-1}_{x_i}x + \Lambda^{-1}_{m_i}m_i^* \tag{7.24b}$$

$$C_3 = x^T\Lambda^{-1}_{x_i}x + m_i\Lambda^{-1}_{m_i}m_i \tag{7.24c}$$

$$D = m_i^TC_1m_i - 2m_i^TC_2 + C_3 \tag{7.24d}$$

Then

$$D = \|m_i - C_1^{-1}C_2\|_{C_1}^2 + \left[C_3 - \|C_2\|_{C_1^{-1}}^2\right]$$

We note that since Λ_{x_i} and Λ_{m_i} are covariance matrices, they are invertible and so is C_1. Now, from (7.23)

$$f(x/\mathcal{H}_i) = \frac{\exp\left\{-\frac{1}{2}(C_3 - \|C_2\|_{C_1^{-1}}^2)\right\}}{2\pi^n|\Lambda_{x_i}\Lambda_{m_i}|^{\frac{1}{2}}} \int_{\mathbf{R}^n} \exp\left\{-\frac{1}{2}\|m_i - C_1^{-1}C_2\|_{C_1}^2\right\} |dm_i|$$

Now change the variable m_i. Let

$$\tilde{m}_i = \frac{1}{\sqrt{2}} C_1^{0.5}[m_i - C_1^{-1}C_2]$$

so that

$$f(x/\mathcal{H}_i) = \frac{1}{2\pi^{\frac{n}{2}}|C_1\Lambda_{x_i}\Lambda_{m_i}|}^{\frac{1}{2}} \exp\left\{-\frac{1}{2}(C_3 - \|C_2\|_{C_1^{-1}}^2)\right\}$$

Now from (7.24) it follows that

$$\begin{aligned} C_1\Lambda_{x_i}\Lambda_{m_i} &= \Lambda_{x_i} + \Lambda_{m_i} \\ C_3 - \|C_2\|_{C_1^{-1}}^2 &= x^T A x + m_i^{*T} B m_i^* - 2x^T C \cdot m_i^* \end{aligned}$$

where

$$\begin{aligned} C &= \Lambda_{x_i}^{-1}(\Lambda_{x_i}^{-1} + \Lambda_{m_i}^{-1})^{-1}\Lambda_{m_i}^{-1} \\ &= \left[\Lambda_{m_i}(\Lambda_{m_i}^{-1} + \Lambda_{x_i}^{-1})\Lambda_{x_i}\right]^{-1} \\ &= (\Lambda_{x_i} + \Lambda_{m_i})^{-1} \\ A &= \Lambda_{x_i}^{-1} - \Lambda_{x_i}^{-1}(\Lambda_{x_i}^{-1} + \Lambda_{m_i}^{-1})^{-1}\Lambda_{x_i}^{-1} \\ &= \Lambda_{x_i}^{-1} - \Lambda_{x_i}^{-1}\Lambda_{m_i}\Lambda_{m_i}^{-1}(\Lambda_{x_i}^{-1} + \Lambda_{m_i}^{-1})^{-1}\Lambda_{x_i}^{-1} \end{aligned}$$

Therefore,

$$\begin{aligned} \Lambda_{x_i}A &= I - \Lambda_{m_i}(\Lambda_{x_i} + \Lambda_{m_i})^{-1} \\ &= [(\Lambda_{x_i} + \Lambda_{m_i}) - \Lambda_{m_i}](\Lambda_{x_i} + \Lambda_{m_i})^{-1} \end{aligned}$$

so that

$$\begin{aligned} A &= (\Lambda_{x_i} + \Lambda_{m_i})^{-1} \\ B &= \Lambda_{m_i}^{-1} - \Lambda_{m_i}^{-1}(\Lambda_{x_i}^{-1} + \Lambda_{m_i}^{-1})^{-1}\Lambda_{m_i}^{-1} \\ &= (\Lambda_{x_i} + \Lambda_{m_i})^{-1} \end{aligned}$$

This implies

$$f(x/\mathcal{H}_i) = \frac{1}{2\pi^{\frac{n}{2}}|\Lambda_{x_i} + \Lambda_{m_i}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\|x - m_i^*\|^2_{(\Lambda_{x_i}+\Lambda_{m_i})^{-1}}\right\}$$

and finally that

$$x \sim \mathcal{N}(m_i^*, \Lambda_{x_i} + \Lambda_{m_i})$$

Proof. (Of Proposition 7.5) From (7.22)

$$f(x/\mathcal{H}_i) = \int_{\mathrm{R}^n} f(x/\mathcal{H}_i, m_i)\psi(m_i)dm_i$$

If the mean m_i has a uniform distribution, then

$$\psi(m_i) = \frac{1}{2^n \Pi_{j=1}^n m_{ij}^*} \qquad b_i \le m_i \le b_i + 2m_i^* \qquad \text{where } m_i^* = [m_{i1}^*, \ldots, m_{in}^*]^T \tag{7.25}$$

From (7.22) and (7.25)

$$f(x/\mathcal{H}_i) = \frac{1}{2\pi^n|\Lambda_{x_i}|^{\frac{1}{2}}} \int_{b_i}^{b_i+2m_i^*} \exp\left\{-\frac{1}{2}\|x_i - m_i\|^2_{\Lambda_{x_i}^{-1}}\right\} |dm_i|$$

Now change the variable m_i

$$\begin{aligned} \tilde{m}_i &= \frac{1}{\sqrt{2}}\Lambda_{x_i}^{-0.5}(m_i - x_i) \\ f(x/\mathcal{H}_i) &= \frac{1}{4^n \Pi_j m_{ij}^*} \frac{2^n}{\pi^{\frac{n}{2}}} \int_{\frac{1}{\sqrt{2}}\Lambda_{x_i}^{-0.5}(b_i - x_i)}^{\frac{1}{\sqrt{2}}\Lambda_{x_i}^{-0.5}(b_i+2m_i^* - x_i)} \exp\left\{-\|\tilde{m}_i\|^2\right\} |dm_i| \end{aligned}$$

The desired result follows as

$$f(x/\mathcal{H}_i) = \frac{1}{4^n \Pi_j m_{ij}^*} \left[\mathrm{erf}\left\{\frac{1}{\sqrt{2}}\Lambda_{x_i}^{-0.5}(x - b_i)\right\} - \mathrm{erf}\left\{\frac{1}{\sqrt{2}}\Lambda_{x_i}^{-0.5}(x - b_i - 2m_i^*)\right\}\right]$$

7.4 Application to Advanced Vehicle Control Systems

In this section, a multiple hypothesis Shiryayev SPRT residual processor is applied to the same fault detection filters as the Bayesian neural networks of Section 6.4. These fault detection filters are designed with the Berkeley nonlinear vehicle simulation operating at $27\frac{\text{m}}{\text{sec}}$ on a straight road. Vehicle lateral dynamics are not considered. A complete description of the fault detection filter design is in (Douglas et al. 1995). Figure 7.3 shows the residual processing scheme using the multiple hypothesis Shiryayev SPRT and the fault detection filters for the longitudinal simulation.

A detailed description of the modeled sensor and actuator faults can be found in (Douglas et al. 1995). Recall that the vehicle longitudinal model has seven two-dimensional sensor faults and two three-dimensional actuator faults. These are combined in output separable and mutually detectable groups with seven or fewer directions. The following list shows the fault groups with fault notation as indicated in Figure 7.3.

Fault detection filter 1.

- (M) : Manifold air mass sensor.
- (W) : Engine speed sensor.
- (X) : Forward acceleration sensor.

Fault detection filter 2.

- (T) : Heave acceleration sensor.
- (Fs) : Rear symmetric wheel speed sensor.
- (Rs) : Forward symmetric wheel speed sensor.

Fault detection filter 3.

- (T) : Pitch rate sensor.
- (Z) : Heave acceleration sensor.
- (Rs) : Rear symmetric wheel speed sensor.

Fault detection filter 4.

$(alfa)$: Throttle angle actuator.

(Tb) : Brake torque actuator.

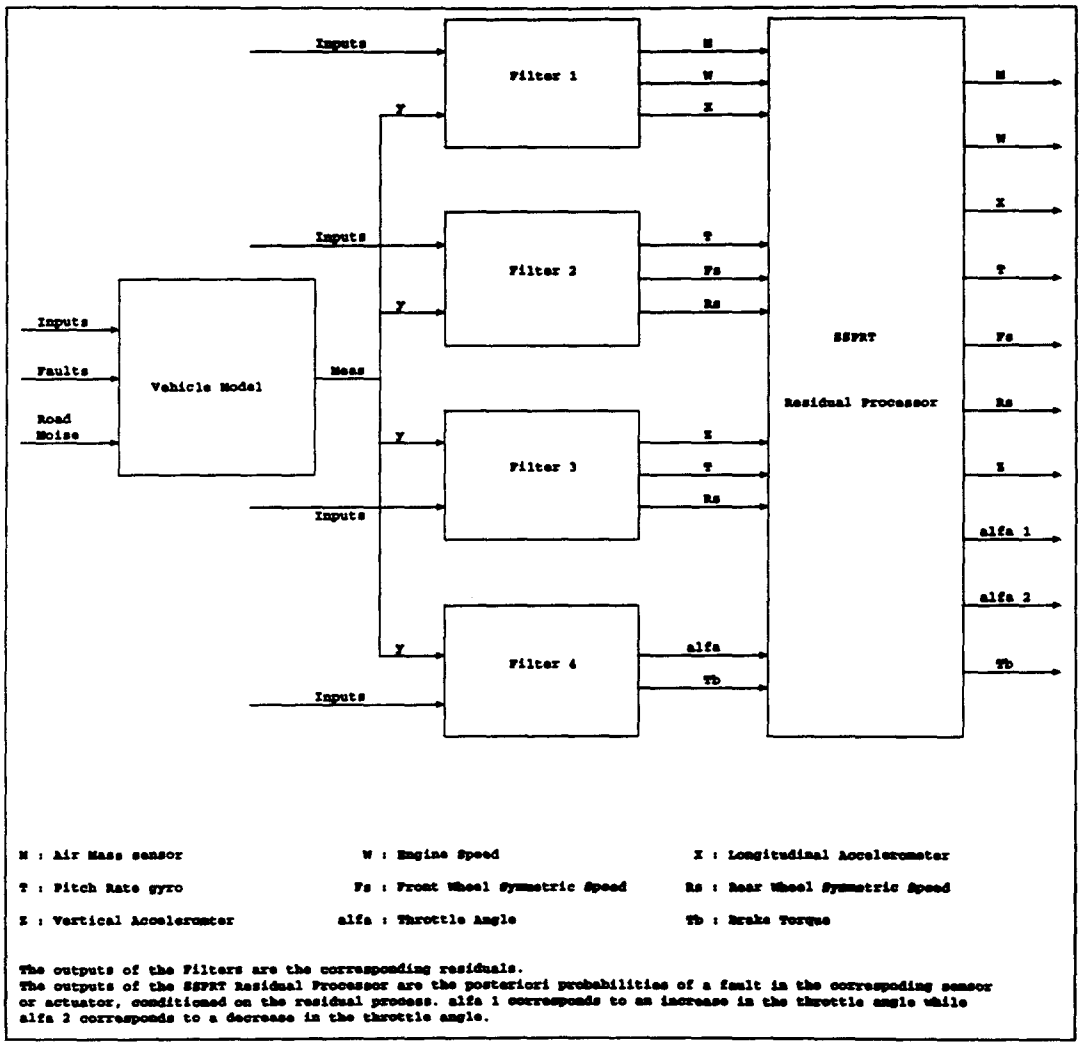

Figure 7.3: Fault detection scheme for AVCS.

A residual processor design should focus on resolving two issues. First, when residuals are driven by model uncertainties, nonlinearities, sensor noise and dynamic disturbances such as road noise, a nonzero residual need not indicate that a fault has occured. The

residual processor should distinguish between a nonzero residual driven by a fault and a nonzero residual driven by something else.

Second, when a fault occurs and the fault is not one included in the fault detection filter design, the directional properties of the residual are undefined. The residual processor should recognize the pattern of a design fault and ignore all other patterns.

Both issues are addressed by a multiple hypothesis SSPRT residual processor. Consider each fault direction as corresponding to a particular hypothesis. Thus, in the present application, there are ten hypotheses $\{\mathcal{H}_0, \ldots, \mathcal{H}_9\}$. Now consider the fault detection filter residual sequence as the meaurement sequence for the SPRT. In the present application, the measurement sequence $\{x_k \in \mathrm{R}^{11}\}$ is assumed to be conditionally independent and gaussian. The density functions for all hypotheses are constructed by computing the sample means and covariance matrices. Finally consider that a step fault models a sudden increase in the mean of the residual process while a ramp fault models a gradual increase in the mean. For the detection and identification of an unknown fault size, the mean of the residual process was assumed to be uniformly distributed.

As an example, step faults are considered in the pitch rate gyro, vertical accelerometer and longitudinal accelerometer. For simplicity, only the residuals corresponding to the particular fault direction are shown in the figures. Figure 7.4 shows a step fault of size $0.05\frac{\mathrm{rad}}{\mathrm{sec}}$ in the pitch rate sensor occuring at 8 seconds. Note that the posteriori probability of a fault in the pitch rate sensor jumps to one almost immediately after the fault occurs. The posteriori probabilities of faults in other sensors and actuators are zero and are not shown.

Figure 7.5 shows a step fault of size $0.5\frac{\mathrm{ft}}{\mathrm{sec}^2}$ in the vertical accelerometer occuring at 8 seconds. Again, the posteriori probability of a fault in the vertical accelerometer jumps to one almost immediately after the fault occurs.

Figure 7.6 shows a step fault of size $0.1\frac{\mathrm{ft}}{\mathrm{sec}^2}$ in the longitudinal accelerometer occuring at 8 seconds. Once again, the posteriori probability of a fault in the longitudinal accelerometer jumps to one almost immediately after the fault occurs.

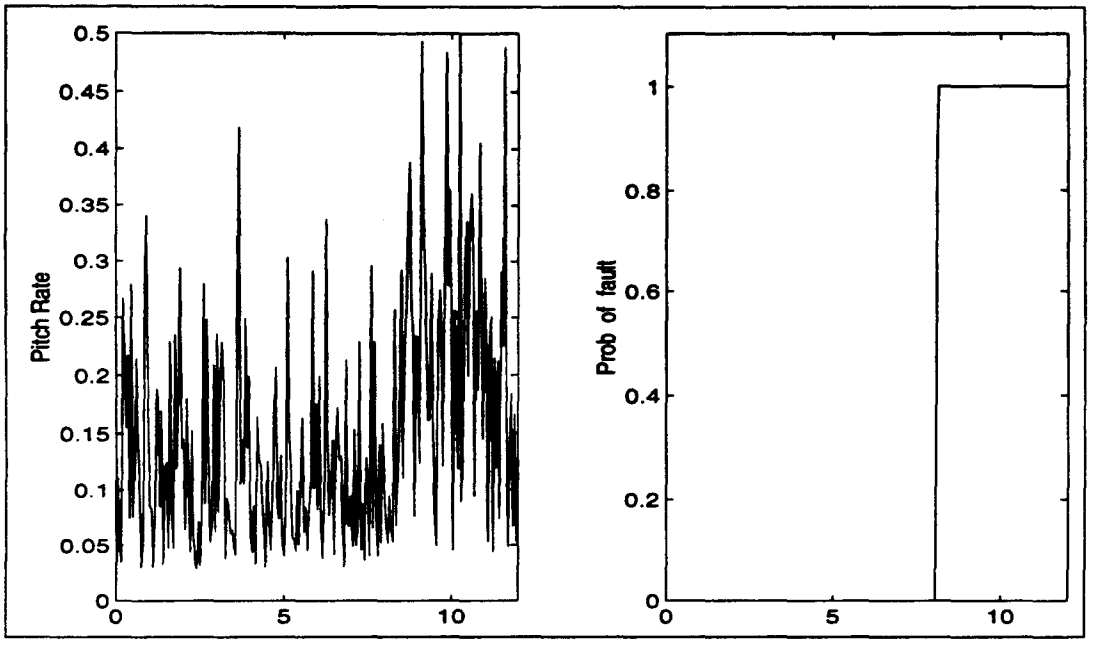

Figure 7.4: Pitch rate sensor fault occurs at 8 sec.

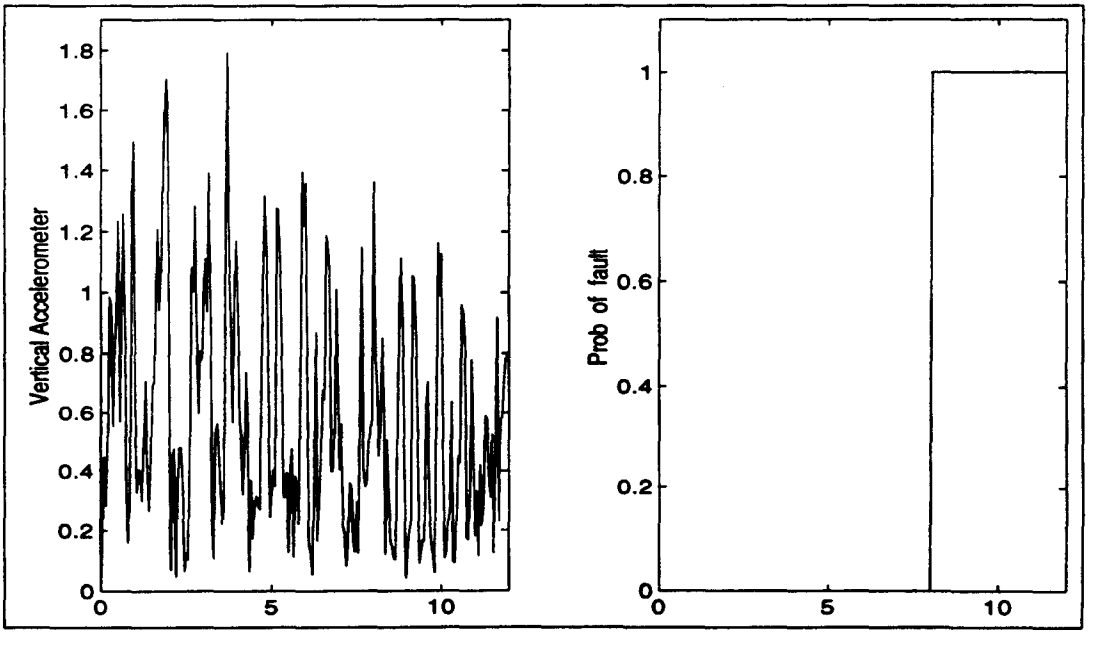

Figure 7.5: Vertical accelerometer fault occurs at 8 sec.

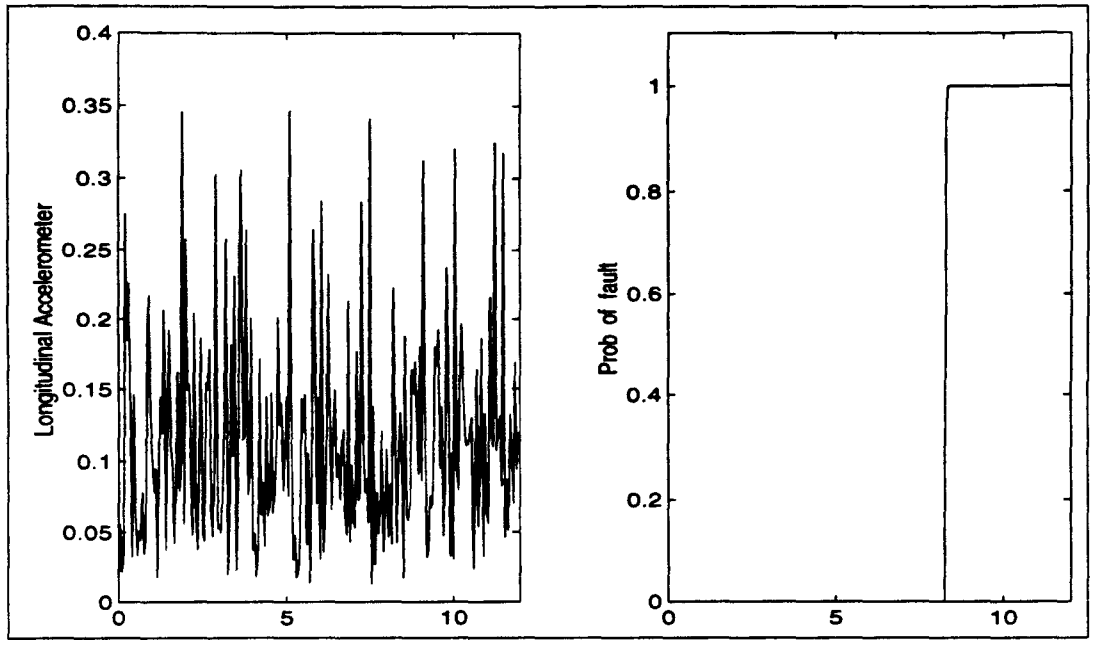

Figure 7.6: Longitudinal accelerometer fault occurs at 8 sec.

7.5 Summary of SPRT Development and Application

A multiple hypothesis SSPRT is derived for the detection and isolation of changes in a conditionally independent measurement sequence. The recursive relation which propagates the posteriori probabilities of all hypotheses requires an approximate knowledge of their apriori probabilities π_i and the probability of change of state p_i from $\mathcal{H}_0$ to $\mathcal{H}_i$. This is not considered as an impediment as the test is found to be insensitive to both parameters as long as they assume reasonable values. The derivation makes no assumption about the structure of the density functions corresponding to all hypotheses and hence, the measurement sequence can be quite general. The generalized Shiryayev SPRT is found to be extremely sensitive to changes even when the underlying density functions for the hypotheses overlap to a large extent. This enhances applicability to practical situations where the fault sizes are typically unknown.