Pergamon

Pattern Recognition, Vol. 27, No. 1, pp. 149-164, 1994
Elsevier Science Ltd
Copyright © 1994 Pattern Recognition Society
Printed in Great Britain. All rights reserved
0031-3203/94 $6.00 + .00

HIDDEN MARKOV MODELS FOR FAULT DETECTION IN DYNAMIC SYSTEMS

PADHRAIC SMYTH
Jet Propulsion Laboratory 238-420, California Institute of Technology, 4800 Oak Grove Drive, Pasadena, CA 91109, U.S.A.

(*Received* 21 *May* 1992; *in revised form* 7 *July* 1993; *received for publication* 6 *August* 1993)

Abstract—Continuous monitoring of complex dynamic systems is an increasingly important issue in diverse areas such as nuclear plant safety, production line reliability, and medical health monitoring systems. Recent advances in both sensor technology and computational capabilities have made on-line permanent monitoring much more feasible than it was in the past. In this paper it is shown that a pattern recognition system combined with a finite-state hidden Markov model provides a particularly useful method for mòdelling temporal context in continuous monitoring. The parameters of the Markov model are derived from gross failure statistics such as the mean time between failures. The model is validated on a real-world fault diagnosis problem and it is shown that Markov modelling in this context offers significant practical benefits.

Fault diagnosis   Classification   Neural networks   Hidden Markov models   Dynamic systems
Monitoring   Reliability   Novel classes

1. INTRODUCTION

Health monitoring of complex dynamic systems is a basic requirement in many domains where safety, reliability and longevity of the system under study are considered critical. The system of interest might be a nuclear power plant, a large antenna system, a telecommunications network, or a human heart. Health monitoring can involve a variety of tasks such as detection of abnormal conditions, identification of faulty components, or prediction of impending failures. The availability at low cost of highly sensitive sensor technology, data acquisition equipment, and VLSI computational power, has made round-the-clock *permanent* monitoring an attractive alternative to the more traditional periodic manual inspection.

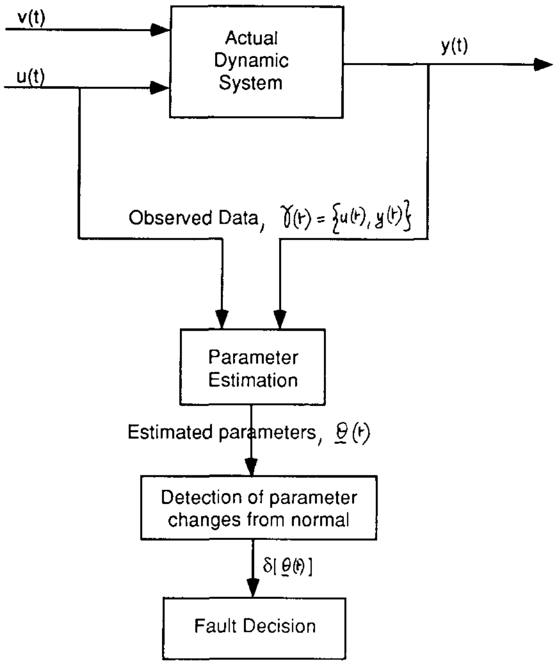
For the purposes of this paper we will restrict our attention to the problem of accurately determining the *state* of the monitored system as a function of time. In particular we assume that a sequence of observed sampled sensor readings γ are available at uniformly-spaced discrete time intervals—without loss of generality the sampling interval is assumed to be 1. Each γ is a k-dimensional measurement. Given a sequence of such sample vectors, $\gamma(t), \gamma(t-1), \ldots, \gamma(0)$, the task is to infer the current state of the system at time t.

We assume that the system must be in one, and only one, of a finite set of m states, ω_i, $1 \leq i \leq m$, at any time. Let Ω be the discrete random variable corresponding to the (unobservable) state of the system, taking values in the set $\{\omega_1, \ldots, \omega_m\}$. Note that the words "states" and "classes" will both be used in this paper but refer to the same thing. One of these states is deemed "normal", the other $m-1$ correspond to fault conditions. This assumption, that the known fault classes are mutually exclusive and exhaustive, limits the proposed method to problems where only single faults occur at any given time and all faults can be described in advance. The first limitation, single fault detection, is a known limitation of most fault detection methods and is inherent in the underlying nature of the sensor information available and the nature of the faults themselves. For example, it is possible that in some problems, multiple faults result in predictable combinations of single fault symptoms—however, this is usually a domain-specific issue and is beyond the scope of discussion in this paper. In practice, since faults are often relatively rare compared to the sampling interval at which decisions are made, the probability of two independent faults occurring within the same time interval is extremely small. We will later see that the second limitation, the assumption that the known faults $\{\omega_2, \ldots, \omega_m\}$ comprise the set of *all* faults which could potentially occur, can be relaxed in a general domain-independent manner. We will also assume throughout that our monitoring algorithm is entirely passive and cannot effect any changes in the system.

2. BACKGROUND ON FAULT DETECTION FOR DYNAMIC SYSTEMS

In the ideal case where the system dynamics and measurement process can be completely modelled in an accurate manner, a variety of optimal control-theoretic methods for fault detection can be derived based on on-line state estimation and statistical analysis of the residual error signals (see Willsky[1] for an overview of such methods).

In practice, however, particularly for large complex systems, it is common to find that the system model may not be that reliable, if indeed there is any system model available. A common technique (Isermann[2]



v(t)
u(t)
Actual Dynamic System
y(t)
Observed Data, γ(t) = {u(t), y(t)}
Parameter Estimation
Estimated parameters, θ(t)
Detection of parameter changes from normal
δ[θ(t)]
Fault Decision

Fig. 1. Parameter estimation approach to fault diagnosis: $u(t)$ is the measured (control) input, $y(t)$ is the measured output signal, $v(t)$ represents unmeasured disturbances to the system, $\boldsymbol{\theta}(t)$ are the estimated system parameters at time t, and $\delta[\boldsymbol{\theta}(t)]$ indicates the change in the values of the system parameters at time t from their nominal values.

and Frank[3]) is to fit a dynamic model to the relationship between the measured input and output signals of the system. Figure 1 shows a simplified block diagram of the basic method. $u(t)$ and $y(t)$ are the measured input and output signals respectively, and $v(t)$ represents unmeasured disturbances to the system.

The model is often a linear difference equation (in the discrete time case) relating inputs and outputs, e.g.

$$y(t) + \sum_{i=1}^{p} \alpha_i y(t-i) = \sum_{j=1}^{q} \beta_j u(t-j-\delta) + e(t) \quad (1)$$

where $e(t)$ is an additive noise term, p and q are the orders of the model, and δ is a delay term. In this example the observed data at time t would be $\gamma(t) = \{u(t), y(t)\}$ and the model parameters would be denoted as $\boldsymbol{\theta} = \{\alpha_1, \ldots, \alpha_p, \beta_1, \ldots, \beta_q\}$. Typically the order or structure of the model (p and q) can be judiciously estimated based upon known system properties—however, the parameters $\boldsymbol{\theta}$ of the model are estimated in an on-line manner using observed input/output data. The lumped parameters of the model can often be related to particular system components. Hence, fault detection occurs by observing changes in the values of the estimated parameter values of the fitted model (compared with some model of their normal condition), which in turn depend on the system components. This method has become known as the parameter method of fault detection—faults are detected by analysing changes in the parameters of the fitted model, as depicted in Fig. 1 by $\delta[\boldsymbol{\theta}(t)]$.

The focus of this paper is on the problem of detecting changes in the underlying system state from parameter estimates $\boldsymbol{\theta}(t), \boldsymbol{\theta}(t-1), \ldots$ using both data-derived estimates of the parameter-state dependence and prior knowledge of the temporal behaviour of the system. As mentioned earlier the system is assumed to always be in one, but only one, state ω_i, $1 \leq i \leq m$, at any point in time, i.e. the states are mutually exclusive and exhaustive. We will also assume that the distribution of parameters conditioned on a given state, $p(\boldsymbol{\theta}|\Omega = \omega_i)$ (where both are measured at the same time t) is stationary, but that there may be some overlap of these state-conditional distributions. We will refer to the dependence $p(\boldsymbol{\theta}|\Omega = \omega_i)$ as the instantaneous model between the parameters and states. In the case of complete overlap (where two or more states possess identical distributions) there is naturally no way to identify the underlying states just by observing the parameters and knowing the instantaneous model. However, as will be shown later in this paper, even when there is significant overlap in the instantaneous model, accurate state identification is still possible by taking the temporal context into account using a hidden Markov model.

We will assume that the application is such that a database or *fault library* can be generated for both the normal class ω_1 and the fault classes $\{\omega_2, \ldots, \omega_m\}$. The database consists of pairs of symptom vectors and class labels, $\{\boldsymbol{\theta}, \Omega\}$, where $\boldsymbol{\theta}$ is the d-dimensional parameter vector estimated from the observed system data. Note that the mapping from $\boldsymbol{\theta}$ to Ω need not be *one-to-one*, since the conditional dependence of $\boldsymbol{\theta}$ given that $\Omega = \omega_i$ is typically probabilistic in nature.

The assumption of availability of labelled training data rules out applications where it is not possible to gather such data—perhaps no such data has been collected in the past and it is not possible to simulate faults in a controlled manner. However, there are many applications where either a fault library already exists, or can be created under controlled conditions (perhaps by testing a particular system in a laboratory). The important point is that for fault diagnosis problems for which such symptom-fault data are readily available, standard supervised classification or discrimination methods can be used to learn a fault diagnosis model from this database.

It is important to note that the parameter estimation technique generally requires far less precise knowledge about the system than the previously-mentioned state-space approach and, hence, tends to be both more widely applicable and more robust from a practical standpoint. For example, in the case of the antenna monitoring problem to be described later, both the presence of nonlinearities and the inherent complexity of the system make it difficult to develop an accurate state-space model. In contrast, the parameter model method can be implemented with relative ease. Naturally, if there is enough knowledge of the system available such that the state-space approach is feasible, then this should give better results since it takes advantage of more information.

As an aside, mention should also be made of knowledge-based or artificial intelligence models which employ *qualitative* models of system behaviour to detect faults. First-generation knowledge-based systems typically use experiential heuristics (described in the form of expert-supplied rules) to describe symptom-fault relationships. More sophisticated second-generation methods (under the broad heading of "model-based reasoning") use qualitative causal models of the system to represent "first-principles" knowledge (Bratko *et al.*[4] and Davis[5]). In principle, this allows the system to identify faults which have never occurred before. Both approaches have limited applicability at present in terms of handling the dynamic and uncertain nature of many real-world problems. In general, the qualitative symbolic representation is not particularly robust for dealing with noisy, continuous data containing temporal dependencies. Furthermore there are many applications for which neither domain experts nor strong causal models exist, thus making the development of a knowledge base very difficult.

3. LEARNING SYMPTOM-FAULT MAPPINGS

The details of the particular classification model used to generate the symptom-fault mapping are not directly relevant to the general discussion. If there is prior knowledge that the probability dependence of the symptoms conditioned on the faults obeys a particular parametric form, such as multi-variate Gaussian, then a maximum-likelihood method to estimate the parameters of the conditional distributions may be appropriate. More commonly there is little prior knowledge regarding the symptom-fault dependencies. In this case non-parametric discriminative methods such as linear discriminants, nearest-neighbour (kNN) methods, decision trees, or neural networks may all be useful approaches depending on the exact nature of the problem at hand. Recent studies using several well-known data sets have shown that all of these classification models perform roughly equally well in terms of predictive accuracy, i.e. their classification performance on independent test data sets was often statistically indistinguishable from each other (Ng and Lippmann[6] and Weiss and Kapouleas[7]). Hence, other attributes of the classification method such as complexity, the ability to handle high dimensional problems, small-sample performance, explicit knowledge representation, and so forth, can become the deciding factors for a given application.

We will impose one particular requirement on the classification method to be used, namely that it produces estimates of the posterior probabilities of the classes ω_i, $1 \le i \le m$, given the input symptoms $\boldsymbol{\theta}$, i.e. $\hat{p}(\Omega = \omega_i|\boldsymbol{\theta})$. In many practical applications estimation of posterior probabilities (as opposed to a simple indication of which class is most likely) is very useful to allow one to control the false alarm rate, the rejection rate, and so forth.

Rather than deal with the time series data directly one usually seeks to extract invariant characteristics of the time series waveforms, where the invariance is with respect to different environmental conditions of operation of the system conditioned on a particular class. These invariant characteristics correspond directly to the estimated system parameters discussed earlier, i.e. what are called system parameters in the control literature can be treated as feature vectors for readers more familiar with pattern recognition terminology. This feature extraction stage can critically affect the classification performance of the overall system. Note that we will use the terms symptoms and features interchangeably in the remainder of the paper.

We will assume the use of a particularly simple feature extraction method whereby the data is windowed into separate consecutive blocks, each containing an integer number T samples. Many variations of this sampling scheme are possible, for example, the use of overlapping blocks or recursive estimates. For the purposes of this paper we will confine our discussion to the relatively simple case of disjoint, consecutive blocks, each of which contain T samples. In practice T is chosen large enough to give reasonably accurate estimates of the features so as to reduce the sampling variance across different windows. For autoregressive models such as equation (1), the $\boldsymbol{\theta}$ coefficients are estimated from all of the observations in a given window of consecutive samples using standard methods such as least squares estimation, i.e.

$$\boldsymbol{\theta}(t) = f(\gamma(t), \gamma(t-1), \ldots, \gamma(t-(T-1)))$$
$$\boldsymbol{\theta}(t-T) = f(\gamma(t-T), \gamma(t-(T+1)), \ldots, \gamma(t-(2T-1))) \quad (2)$$

and so forth.

What has been expressed at this point, assuming that a particular estimation method and classification algorithm had been chosen, is simply a framework for generating estimates of the state of the system at any point in time, i.e. at intervals of time T the classification system will produce estimates of the posterior class probabilities given the features which are estimated over the $[t, t-T]$ time interval. This approach makes an independent decision at each time instant, i.e. class probability estimates or symptom data from the past do not influence the present estimates. Clearly this is suboptimal given the fact that faults are *persistent* over time and, hence, that better class estimates could be obtained by making use of past information. Two obvious approaches spring to mind in order to model this temporal context. In the first, one could introduce some form of memory into the classification model. Examples of such memory methods include recurrent neural networks (i.e. networks where the outputs are fed back to the inputs after a unit delay, Pineda[8] and Pearlmutter[9]) or a "window in time" technique whereby the classifier is trained not only on feature values at time t, but also on values from time $t-T$ back to $t-MT$ where M is the memory of the model (Waibel *et al.*[10]). This approach of *implicitly* modelling temporal context has the significant disadvantage of making it much more difficult to train the classifier. The second approach (which we will now describe), of using a hidden Markov model, is much more elegant in that it combines over time the *instantaneous* estimates of the trained classifier by taking advantage of prior knowledge about the gross statistical properties of the failure modes of the system.

4. HIDDEN MARKOV MODELS FOR MODELLING TEMPORAL CONTEXT

We now describe the use of discrete-time, finite-state, hidden Markov models for smoothing classification decisions over time. Note that for the purposes of this discussion the terms "class" and "state" are equivalent, i.e. both refer to the set of normal and fault conditions $\{\omega_1, \ldots, \omega_m\}$.

A first-order temporal Markov model is characterized (in the present context) by the assumption that

$$\begin{aligned} &p(\Omega(t) = \omega_i | \Omega(t-T), \Omega(t-2T), \ldots, \Omega(0)) \\ &\quad = p(\Omega(t) = \omega_i | \Omega(t-T)), \quad 1 \le i \le m, \quad \forall t \qquad (3) \end{aligned}$$

i.e. that the conditional probability of any current state given knowledge of all previous states is the same as the conditional probability of the current state given knowledge of the system state at time $t - T$. Hence, assuming stationarity, to calculate the probability of any state at time t, one need only know the initial state probabilities $\pi(0) = [p(\Omega(0) = \omega_1), p(\Omega(0) = \omega_2), \ldots, p(\Omega(0) = \omega_m)]$ and the values $p(\Omega(t) = \omega_i | \Omega(t-T) = \omega_j)$, $1 \le i, j \le m$. The $m \times m$ matrix $\mathbf{A}$, where $a_{ij} = p(\Omega(t) = \omega_i | \Omega(t-T) = \omega_j)$, is known as the transition matrix and characterizes the Markov model. Given $\mathbf{A}$ and π one can calculate the probability of any state at any time t.

Let us assume at this point of the discussion that the discrete-time Markov model described above can be used to model the failure behaviour of the system of interest, i.e. at any time t, given that the system is in a particular state j, the probability that the system will be in state i at time $t + T$ is described by the state transition probability $a_{ij} = p(\Omega(t) = \omega_i | \Omega(t-T) = \omega_j)$. We will later discuss the implications of using such a model and the use of failure rates to estimate the transition probabilities: however, at this point we will focus on how the model is used. Markov models such as this can be used for reliability analyses to determine long-term failure rates and modes of a system (Papazoglou and Gyftopoulos[11]).

However, the goal here is somewhat different, namely to monitor the system in real-time. The key point is that the states of the system are not directly observable, but are *hidden*, i.e. the monitoring system has no direct way to measure the state of the system, even for past time. Instead, various symptoms or features $\boldsymbol{\theta}(t)$ are observable. These features are a probabilistic function of the states: in fact the classification models mentioned earlier can estimate an *instantaneous* symptom-state mapping $p(\Omega(t) = \omega_i | \boldsymbol{\theta}(t))$. By making the appropriate conditional independence assumptions we shall see how one can estimate $p(\Omega(t) = \omega_i | \boldsymbol{\theta}(t), \boldsymbol{\theta}(t-T), \ldots, \boldsymbol{\theta}(0))$ without explicitly providing the $\boldsymbol{\theta}(t-T), \ldots, \boldsymbol{\theta}(0)$ as direct inputs to the classifier.

The hidden Markov formalism provides an exact solution to this problem provided the underlying conditional independence assumptions are met. It has been widely applied with significant success in speech-recognition applications (Rabiner[12]). Let $\Phi_t = \{\boldsymbol{\theta}(t), \boldsymbol{\theta}(t-T), \boldsymbol{\theta}(t-2T), \ldots, \boldsymbol{\theta}(0)\}$ denote the observed symptoms up to time t with probability $p(\Phi_t) = p(\{\boldsymbol{\theta}(t), \ldots, \boldsymbol{\theta}(0)\})$. It is convenient to work in terms of an intermediate variable $\alpha_i(t)$, where

$$\alpha_i(t) = p(\Omega(t) = \omega_i, \Phi_t) \qquad (4)$$

is the joint probability of state i (at time t) and the observed symptoms. To find the posterior probabilities of interest it is sufficient to be able to calculate the values of α at any time t since by Bayes' rule

$$p(\Omega(t) = \omega_i | \Phi_t) = \frac{\alpha_i(t)}{p(\Phi_t)} = \frac{\alpha_i(t)}{\sum_{j=1}^{m} \alpha_j(t)}. \qquad (5)$$

We derive a recursive estimate as follows:

$$\begin{aligned} \alpha_i(t) &= \sum_{j=1}^{m} p(\Omega(t) = \omega_i, \Phi_t, \Omega(t-T) = \omega_j) \\ &= \sum_{j=1}^{m} p(\Omega(t) = \omega_i, \boldsymbol{\theta}(t), \Phi_{t-T}, \Omega(t-T) = \omega_j) \\ &= \sum_{j=1}^{m} p(\Omega(t) = \omega_i, \boldsymbol{\theta}(t) | \Phi_{t-T}, \Omega(t-T) = \omega_j) \\ &\quad \cdot p(\Phi_{t-T}, \Omega(t-T) = \omega_j) \\ &= \sum_{j=1}^{m} p(\Omega(t) = \omega_i, \boldsymbol{\theta}(t) | \Phi_{t-T}, \Omega(t-T) = \omega_j) \alpha_j(t-T) \end{aligned}$$

(by the definition of α_j)

$$\begin{aligned} &= \sum_{j=1}^{m} p(\boldsymbol{\theta}(t) | \Omega(t) = \omega_i, \Phi_{t-T}, \Omega(t-T) = \omega_j) \\ &\quad \cdot p(\Omega(t) = \omega_i | \Phi_{t-T}, \Omega(t-T) = \omega_j) \alpha_j(t-T) \\ &= \sum_{j=1}^{m} p(\boldsymbol{\theta}(t) | \Omega(t) = \omega_i) p(\Omega(t) = \omega_i | \Phi_{t-T}, \\ &\qquad \Omega(t-T) = \omega_j) \alpha_j(t-T) \end{aligned}$$

(assuming that $\boldsymbol{\theta}(t)$ is independent of past observations and past states, *given* the present state)

$$\begin{aligned} &= \sum_{j=1}^{m} p(\boldsymbol{\theta}(t) | \Omega(t) = \omega_i) p(\Omega(t) = \omega_i | \Omega(t-T) = \omega_j) \\ &\quad \cdot \alpha_j(t-T) \end{aligned}$$

(since by the basic Markov assumption $\Omega(t)$ only depends on $\Omega(t-T)$)

$$= p(\boldsymbol{\theta}(t) | \Omega(t) = \omega_i) \sum_{j=1}^{m} a_{ij} \alpha_j(t-T). \qquad (6)$$

The first term can be derived (within a constant factor) from the classifier's estimate of $p(\Omega(t) = \omega_i|\theta(t))$ and Bayes' rule. The terms in the sum are just a linear combination of the values of α from the previous time-step. Hence, equation (6) gives the basic recursive relationship for estimating state probabilities at any time t.

The derivation of equation (6) made use of the two basic assumptions which are commonly used in hidden Markov models: that the state at time t depends only on the state at time $t - T$, and that the observed symptoms at time t are dependent only on the state at time t. The first assumption is the basic first-order Markov assumption on the states. The second assumption is that $\theta(t)$ is independent of both the most recent state and the observed past data, *given* that the present state is known. This implies that we assume that the observed symptoms are statistically independent from one time window to the next, given the state information. For disjoint, non-overlapping, blocks of data this will generally be true if the feature sampling rate $1/T$ is greater than any significant frequency component in the underlying observed time-series $\gamma(t)$. For overlapping blocks of data, or where T is comparable to the time constants of the dynamic system, observed symptoms would no longer be independent and the model would be modified to include a measure of this dependence.

Note that we have specified here only how to calculate the state probabilities based on *past* information. Alternative estimation strategies are possible. For example, using the well-known forward–backward recurrence relations (Rabiner[12]) one can update the state probability estimates using symptom information which occurred *later* in time, i.e. estimate $p(\Omega(t) = \omega_i|\theta(t + kT), \ldots, \theta(t), \ldots, \theta(0))$. From an operational standpoint this allows further smoothing of glitches and a consequent reduction in false alarms—the disadvantage is that there is a latency of time kT before such an estimate can be made. Another approach is to use the Veterbi algorithm to estimate the most likely *joint* sequence of states, i.e.

$$\max\{p(\Omega(t) = \omega_i, \ldots, \Omega(0) = \omega_j|\Phi_t)\}.$$

Which scheme is used depends largely on the particular application and each can easily be implemented using a variation of the recursive equations derived above. The probability estimation method based only on *past* and *present* measurements (as described in equations (5) and (6) is the most direct method for on-line monitoring and will be assumed throughout the rest of the paper.

5. THE NATURE OF THE MARKOV TRANSITION MATRIX

In the previous sections we have assumed the existence of the transition matrix **A**—the question naturally arises in practice as to how the entries in this matrix are obtained. For speech recognition applications there is typically an abundance of training data from which **A** can be estimated by the use of iterative maximum likelihood procedures such as the Baum–Welch algorithm. However, for reliability monitoring, while there may be data obtained under specific normal and fault conditions, there will typically not be a set of training data corresponding to a *sequence* of state transitions. Hence, in practice, prior knowledge regarding the overall system reliability and behaviour must be brought to bear in order to provide estimates of **A**. We adopt a divide-and-conquer approach by dividing the states into three categories: first is the normal state, then the intermittent states, and finally the "hard-fault" states. The difference between the latter two is that intermittent failures allow the possibility of returning to the normal state whereas the "hard-fault" states do not.

5.1. *Specification of the "normal-normal" transition probability*, a_{11}

The use of a first-order Markov model to describe failure processes implicitly assumes that the lengths of times between failures are distributed geometrically. This follows from the fact that for a discrete-time Markov model the probability that the system stays in state ω_i for n time steps is $p^{n-1}(1 - p)$ where $p = a_{ii}$. The memoryless assumption which leads to the geometric distribution of inter-failure durations is quite robust and plausible for many applications and is widely used in reliability analysis to model failure processes (Siewiorek and Swarz[13]).

By relating the Markov transition parameters to overall failure statistics of the system we can both check the validity of the geometric distribution assumption and also determine the transition probabilities themselves. The expected length l of time spent in state ω_1, given that it starts in state ω_1, is

$$E[l] = \sum_{n=1}^{\infty} n a_{11}^{n-1}(1 - a_{11}) \tag{7}$$

$$= \frac{1}{1 - a_{11}} \tag{8}$$

is units of time T. Thus, the mean time between failure (MTBF) of the system can be expressed as

$$\frac{\text{MTBF}}{T} = \frac{1}{1 - a_{11}} \tag{9}$$

and, hence

$$a_{11} = 1 - \frac{T}{\text{MTBF}} \tag{10}$$

where the MTBF and T are expressed in the same time units. In this manner, MTBF statistics can be used as the basis for estimating a_{11}. The MTBF of the system can typically be either specified by a reliability analysis (for a new system) or can be estimated from a problem database (for a system which has been in use for some time). Note that T will be chosen to be much smaller than the MTBF in practice.

5.2. *Specification of the fault transition probabilities*

Transition probabilities into both intermittent and hard faults from the normal state are found by weighting $1 - a_{11}$ (the probability of the system entering a fault state at the next time step given that it is currently in the normal state) by the anticipated relative likelihood of occurrence of each fault state. These relative likelihoods may be derived from reliability analysis or can be estimated empirically if a problem database exists.

The mean anticipated duration of intermittent failures can be used to calculate the self-transition probability for intermittent states in an analogous manner to the way in which the MTBF was used above to find a_{11}. Knowledge of intermittent fault duration is typically more subjective in nature than finding the MTBF and may require knowledge of the physics of the fault condition.

Conceptually, hard faults present a problem (in the context of Markov monitoring) since once such a fault occurs the system cannot return to the normal state until the fault is physically repaired, which in turn typically requires downtime of the system. In practice, a sensible approach is to define an "absorbing" state which indicates that the system has been halted. Hence, the only allowable transition out of a hard fault state is into the halt state. The length of time which the system may spend in the hard fault state, before the halt state is arrived at, is largely a function of the operational environment: if the Markov monitoring system itself is being used as part of an overall alarm system, or if the fault is detectable by other means, then an operator may shut down operations quickly. On the other hand, if the fault does not manifest itself in any significant observable manner and if the Markov monitoring system is being used only for off-line data analysis, then the system may remain in the hard fault state for a lengthy period of time. Hence, deciding how the self-transition probabilities are chosen for the hard-fault classes will be quite specific to particular operational environments.

To complete the Markov transition matrix it is sufficient to note that "fault-to-fault" transitions are normally disallowed except in cases where there is sufficient prior knowledge to believe that intermittent faults can occur directly in sequence.

5.3. *Comments on robustness and dynamics*

The process of defining the Markov transition matrix is obviously quite subjective in nature. While this could be viewed as a weakness of the overall methodology, one can argue that in fact it is a strength. In particular, it allows the effective coupling of relatively high-level prior knowledge (in the form of the Markov transition matrix **A**) with the "lower-level" data-driven estimation of $p(\Omega|\boldsymbol{\theta})$. Naturally, the latitude in specification of **A** leads to questions regarding the sensitivity of the method to misspecification. While a systematic sensitivity study is beyond the scope of this paper, empirical results using this method suggest that unless the parameter-state conditional densities are almost entirely overlapped, then the model is quite robust to variations in **A**—typically, only the length of time to switch between states ("time to detect") is directly affected.

The astute reader will have noticed that for a typically reliable system the dynamics of the Markov model will be such that it will remain in the normal state for long stretches of time. It is important to realize that the relatively static behaviour of the model should not undermine the reader's assessment of its practical utility: for many problems it is often extremely difficult to design detectors of rare events which have both a low false alarm rate and a high detection rate. For example, in the next section we will discuss an application where the system makes classification decisions every 6 s or so, while the MTBF is on the order of a few days. For this application, if the Markov model component of the method is omitted and only the instantaneous state estimates are used, the false alarm rate increases dramatically to the extent that this non-Markov method would be completely impractical for use in an operational environment.

6. BACKGROUND ON ANTENNA FAULT DIAGNOSIS

We now describe the application of the hidden Markov model to a real fault monitoring problem. It is first helpful to provide some background. The Deep Space Network (DSN) (designed and operated by the Jet Propulsion Laboratory for the National Aeronautics and Space Administration (NASA)) provides end-to-end telecommunication capabilities between earth and various interplanetary spacecraft throughout the solar system. The ground component of the DSN consists of three ground station complexes located in California, Spain and Australia, giving full 24 h coverage for deep space communications. Since spacecraft are always severely limited in terms of available transmitter power (for example, each of the Voyager spacecraft only use 20 W to transmit signals back to earth), all subsystems of the end-to-end communications link (radio telemetry coding, receivers, amplifiers) tend to be pushed to the absolute limits of performance. The large steerable ground antennas (70 and 34 m dishes) represent critical potential single points of failure in the network. In particular there is only a single 70 m antenna at each complex because of the large cost and calibration effort involved in constructing and operating a steerable antenna of that size—the entire structure (including pedestal support) weighs over 8000 tons.

The antenna pointing systems consist of azimuth and elevation axes drives which respond to computer-generated trajectory commands to steer the antenna in real-time. Pointing accuracy requirements for the antenna are such that there is little tolerance for component degradation. Achieving the necessary degree of positional accuracy is rendered difficult by various non-linearities in the gear and motor elements and environmental disturbances such as gusts of wind af-

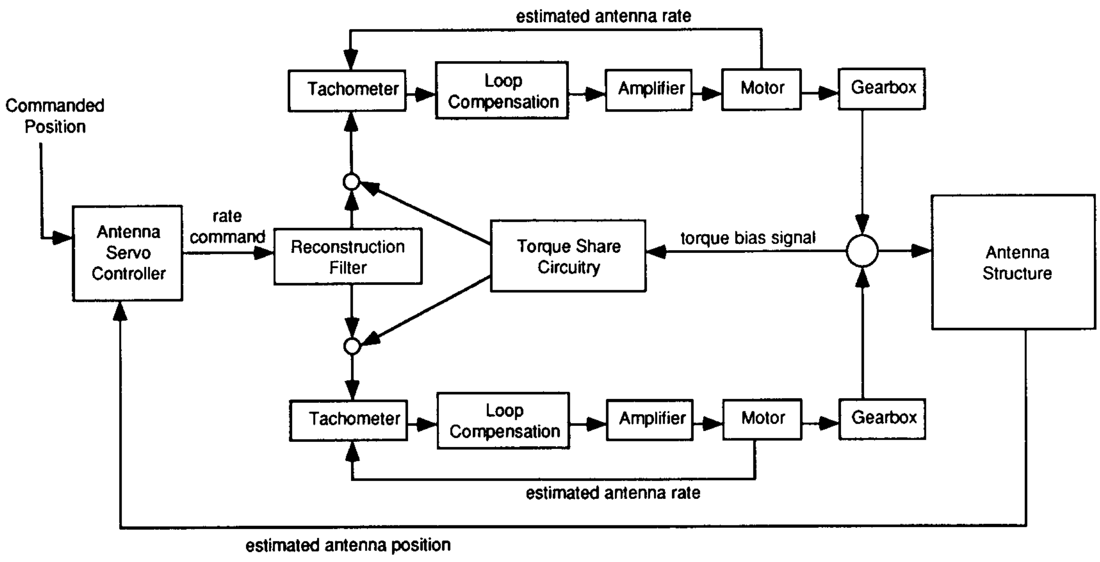


Fig. 2. Simplified block diagram of antenna pointing system.

fecting the antenna dish structure. Off-beam pointing can result in rapid fall-off in signal-to-noise ratios and consequent potential loss of irrecoverable scientific data from the spacecraft.

The antenna servo pointing systems are a complex mix of electro-mechanical components. Figure 2 shows a simple block diagram of the elevation pointing system for a 34 m antenna—see Appendix B for a brief description of how the pointing system works. A faulty component manifests itself indirectly via a change in the characteristics of observed sensor readings in the pointing control loop. Due to the non-linearity and feedback present, direct causal relationships between fault conditions and observed symptoms can be difficult to establish—this makes manual fault diagnosis a slow and expensive process. In addition, if a pointing problem occurs while a spacecraft is being tracked, the antenna is often shut-down to prevent any potential damage to the structure and the track is transferred to another antenna if possible. Hence, at present, diagnosis often occurs after the fact, where the original fault conditions may be difficult to replicate. In the next section we describe the application to this problem of the methods outlined earlier.

7. EXPERIMENTAL RESULTS

7.1. *Data collection and feature extraction*

The observable antenna data consists of various sensor readings (in the form of sampled time series) which can be monitored while the antenna is in tracking mode. To generate a fault library hardware faults were introduced in a controlled manner by switching faulty components in and out of the control loop. Sensor variables monitored included wind speed, motor currents, tachometer voltages, estimated antenna position, and so forth.

The time series data were initially sampled at 50 Hz (well above the estimated Nyquist sampling rate for signals of interest) and segmented into windows of 4 s duration (200 samples) to allow reasonably accurate estimates of the various features. The features are derived by applying an autoregressive-exogenous (ARX) modelling technique using the rate feedback command as the input to the model and motor current as output (see Figs 1 and 2):

$$y(t) + \sum_{i=1}^{p} a_i y(t-i) = \sum_{j=1}^{q} b_j u(t-j) + e(t), \qquad t = 1, 2, \ldots, N \quad (11)$$

where $y(t)$ is the motor current, $u(t)$ the rate command input, $e(t)$ an additive white noise process, and a_i and b_j are the model coefficients. The model order was chosen by finding an empirical minimum (using data from normal conditions) of the Akaike Information Criterion (AIC) which trades-off goodness-of-fit to the data with model complexity (Ljung[14]). An eighth-order model was chosen in this manner with $p = 6$ and $q = 2$, resulting in 8 ARX features. Using this model structure, a separate set of ARX coefficients was estimated from each successive 4 s window of data using direct least mean squares estimation. Hence a new set of features, $\boldsymbol{\theta}(t)$, is available at a rate of 0.25 Hz compared to the original sampling rate of 50 Hz—for this particular application this rate of decision-making is more than adequate. Hence, note that for the sequence of feature vectors $\{\ldots, \boldsymbol{\theta}(t-T), \boldsymbol{\theta}(t), \boldsymbol{\theta}(t+T), \ldots\}$, each feature vector is estimated from a separate, non-overlapping, window of 200 samples from the original time series. Note also that the autoregressive representation is particularly useful for discriminative purposes when dealing with time series (Kashyap[15]).

In addition to the ARX features we also included four time domain features (such as the estimated standard deviations of tachometers and torque sensors) which were judged to have useful discriminative power. It is worth pointing out that for the chosen sample size of 200 it was found that the assumption that feature estimates do not have any temporal dependence across windows was justified. This observation is based on empirical results obtained by analysing the correlation structure in the training data.

7.2. *Model development*

Data were collected at a 34 m antenna site in Goldstone, California, in early 1991, under both normal and fault conditions. The two faults correspond to a failed tachometer in the servo loop and a short circuit in the electronic compensation loop—these are two of the most problematic components in terms of reliability. The data consisted of 15,000 labelled sample vectors for each fault, which was converted to 75 feature vectors per class. Data were collected on two separate occasions in this manner. Because the antenna is in a remote location and is not permanently instrumented for servo component data acquisition, data collection in this manner is a time-consuming and expensive task. Hence, the models were trained with relatively few data points per class.

Experiments were carried out with both a feedforward multi-layer neural network and a simple maximum-likelihood Gaussian classifier. A description of the neural network model used is given in Appendix A. The neural network was chosen over alternative classification models because of its ability to approximate arbitrary decision boundaries in a relatively non-parametric manner. In addition, by using a mean-square error objective function, the outputs of the network can be used as estimates of posterior class probabilities (Richard and Lippmann(16) and Miller *et al.*(17)). Based on cross-validation results, a network with a single hidden layer of 12 units was chosen as the working model. The networks were trained using a conjugate gradient variation of the well-known backpropagation method (Barnard and Cole(18) and Powell(19)). The Gaussian classifier used a separate, diagonal covariance matrix for each class, where the components consisted of maximum likelihood estimates. Using the full covariance matrix was considered impractical given only 150 samples per class in 12 dimensions. Note that the use of feature transformation techniques such as principal components analysis or the use of structured covariance matrices could in principle alleviate this problem, however, these approaches were not pursued here. Components of the Markov transition matrix **A** were estimated using a database of trouble reports which are routinely collected at all antenna sites—see Appendix C for a more detailed discussion.

Figure 3 shows the overall model structure and the sequence of calculations which occur at time t. At the top is the pointing system being monitored as described in Appendix B. The next level down shows the parameter estimation stage where the parameters $\boldsymbol{\theta}(t)$ are estimated based on 200 samples (which have taken 4 s to collect). This is followed below by the parameter/state conditional probability model, which in our case is either a Gaussian or neural network classifier which produces the instantaneous estimate of the state probabilities $p(\Omega(t) = \omega_i|\boldsymbol{\theta}(t))$, $1 \leq i \leq m$. Finally, at the bottom of the figure is the Markov component, which combines the past state estimates $p(\Omega(t) = \omega_i|\boldsymbol{\theta}(t-T))$, $1 \leq i \leq m$ and the current instantaneous estimates using the recursive structure of equation (6). In November 1991, these models were implemented in software as part of the data acquisition system (using a Macintosh II computer). The results of testing the models on previously unseen data in real-time at the antenna site are discussed in the next section.

7.3. *Classification results*

The neural and Gaussian models, both with and without the Markov component, were tested by monitoring the antenna as it moved at typical deep-space tracking rates of about 4 mdeg s^{-1}. The results reported below consist of summary results over a variety of different short tests: the cumulative monitoring time was about 1 h in duration.

Table 1 summarizes the overall classification performance for each of the models, and both for each individual class and for all classes averaged together. Clearly, from the final column, the neural-Markov model is the best model in the sense that no windows at all were misclassified. It is significantly better than the Gaussian classifier which performed particularly poorly under fault conditions. However, under normal conditions it was quite accurate having only 1 false alarm during the roughly 30 min of time devoted to monitoring normal conditions—this is not too surprising since in theory at least the ARX coefficients should obey a multi-variate Gaussian distribution given that the model is correct, i.e. for the non-fault case (Ljung(14)). The effect of the Markov model is clearly seen to have beneficial effects, in particular reducing the effects of isolated random errors. However, for the compensation loss fault, the Markov model actually worsened the already poor Gaussian model results, which is to be expected if the non-Markov component is doing particularly poorly as in this case.

Table 2 presents the same data summarized in terms of the logarithm (base 10) of the mean-square error (MSE), calculated as follows:

$$\mathrm{MSE} = \frac{1}{N} \sum_{j=1}^{N} \sum_{i=1}^{m} (\hat{p}(\omega_i(j)) - o_i(j))^2 \qquad (12)$$

where $\hat{p}(\omega_i(j))$ is the classifier's estimate of the posterior probability of class i for input $j, o_i(j) = 1$ if ω_i is the true class for input j and zero otherwise, and N the size of the training data set. The mean-square error provides more information on the probabilities being produced by the classifier than the classification error rates. Lower values imply that the probabilities are sharper, i.e. the classifier is more certain in its conclusion.

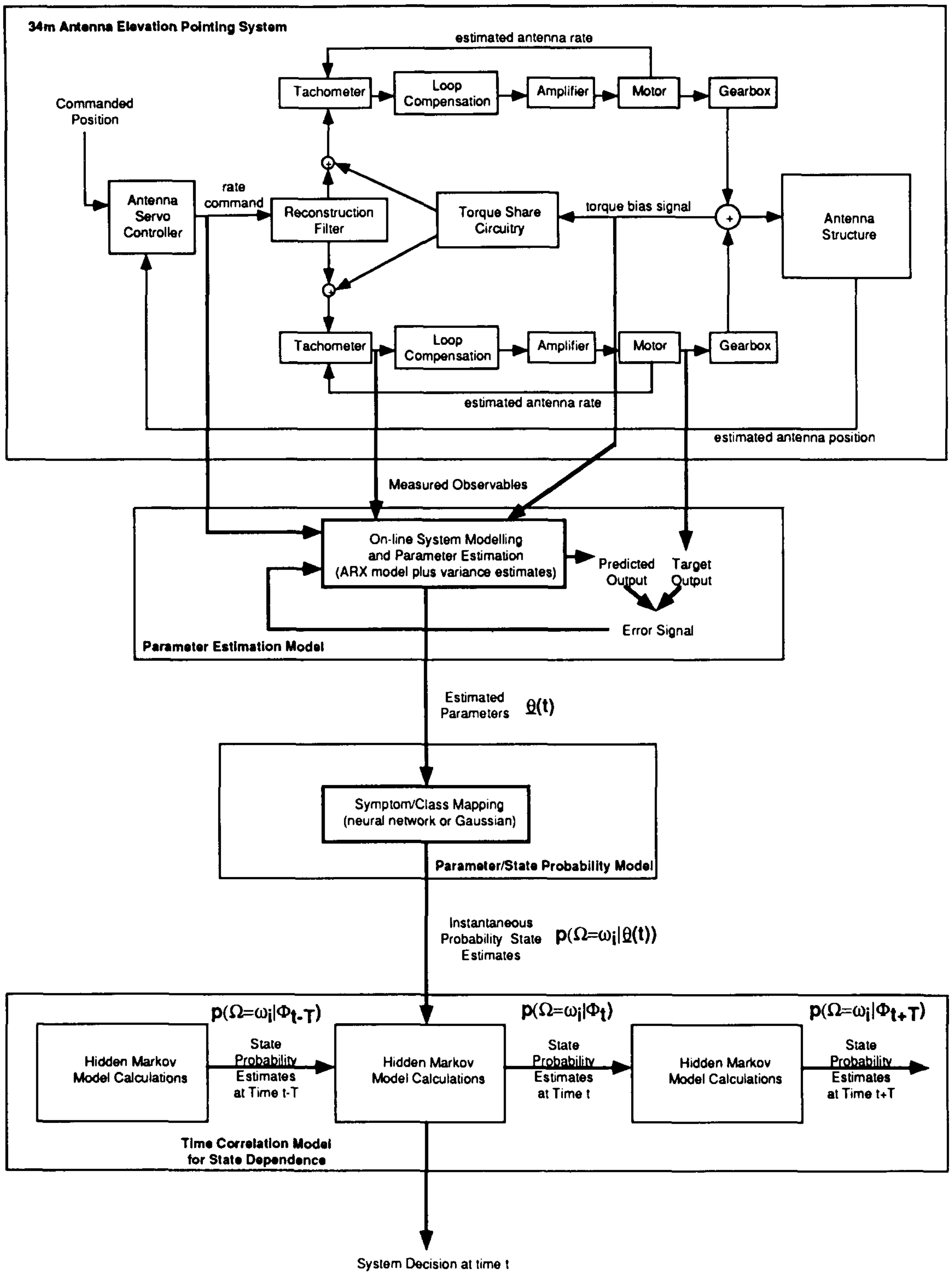

34m Antenna Elevation Pointing System
estimated antenna rate
Tachometer
Loop Compensation
Amplifier
Motor
Gearbox
Commanded Position
Antenna Servo Controller
rate command
Reconstruction Filter
Torque Share Circuitry
torque bias signal
Antenna Structure
Tachometer
Loop Compensation
Amplifier
Motor
Gearbox
estimated antenna rate
estimated antenna position
Measured Observables
On-line System Modelling and Parameter Estimation (ARX model plus variance estimates)
Predicted Output
Target Output
Error Signal
Parameter Estimation Model
Estimated Parameters θ(t)
Symptom/Class Mapping (neural network or Gaussian)
Parameter/State Probability Model
Instantaneous Probability State Estimates p(Ω=ωi|θ(t))
p(Ω=ωi|Φt-T)
p(Ω=ωi|Φt)
p(Ω=ωi|Φt+T)
Hidden Markov Model Calculations
State Probability Estimates at Time t-T
Hidden Markov Model Calculations
State Probability Estimates at Time t
Hidden Markov Model Calculations
State Probability Estimates at Time t+T
Time Correlation Model for State Dependence
System Decision at time t

Fig. 3. Block diagram of antenna system and monitoring model. The parameter estimation stage estimates $\theta(t)$, followed below by the parameter/state conditional probability model which produces the instantaneous estimate of the state probabilities $p(\Omega(t) = \omega_i|\theta(t))$, $1 \leq i \leq m$ and finally, the Markov component, which combines the past state estimates $p(\Omega(t) = \omega_i|\theta(t - T))$ and the current instantaneous estimates using the recursive structure of equation (6).

Table 1. Percentage misclassification rates for Gaussian and neural models both with and without Markov component

	Without Markov model		With Markov model	
Class	Gaussian	Neural	Gaussian	Neural
Normal conditions	0.36	1.72	0.36	0.00
Tachometer failure	27.78	0.00	2.38	0.00
Compensation loss	34.21	0.00	43.16	0.00
All classes	16.92	0.84	14.42	0.00

Table 2. Logarithm of mean-squared error for Gaussian and neural models both with and without Markov component

	Without Markov model		With Markov model	
Class	Gaussian	Neural	Gaussian	Neural
Normal conditions	−2.44	−1.97	−2.46	−4.24
Tachometer failure	−0.40	−3.52	−0.42	−4.22
Compensation loss	−0.82	−3.48	−1.39	−4.71
All classes	−0.87	−2.29	−1.02	−4.34

The general trend in Table 2 is that the neural-Markov combination is significantly better than any of the other combinations.

Figures 4–6 plot the estimated probability of the true class as a function of time for various models to allow a more detailed interpretation of the results. Note that, given that the true class is labelled i, the estimated probability of class i from the neural network corresponds to the *normalized* output of output unit i of the network at time t, i.e.

$$\hat{p}_i(t) = \frac{\hat{o}_i(t)}{\sum_{k=1}^{4} \hat{o}_k(t)} \tag{13}$$

(where $\hat{o}_i(t)$ is the value of the ith network output node) while the Markov probabilities correspond to the estimates of $p(\Omega(t) = \omega_i | \Phi(t))$, as described earlier in equation (6).

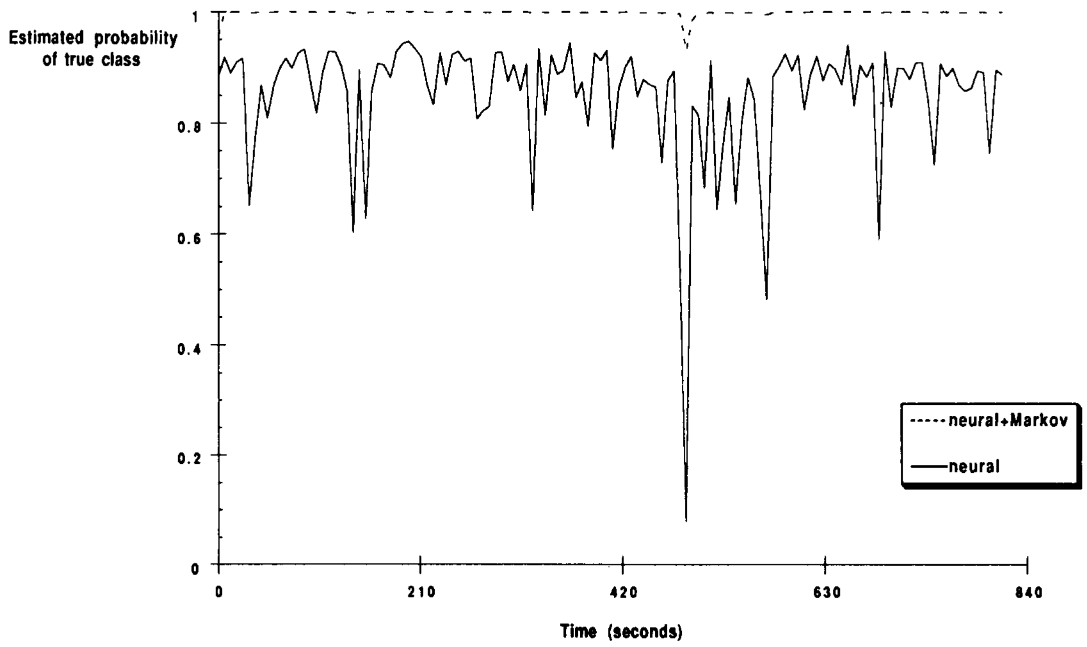


Fig. 4. Comparison of Markov and non-Markov models: estimate of probability of true class (normal conditions) as a function of time.

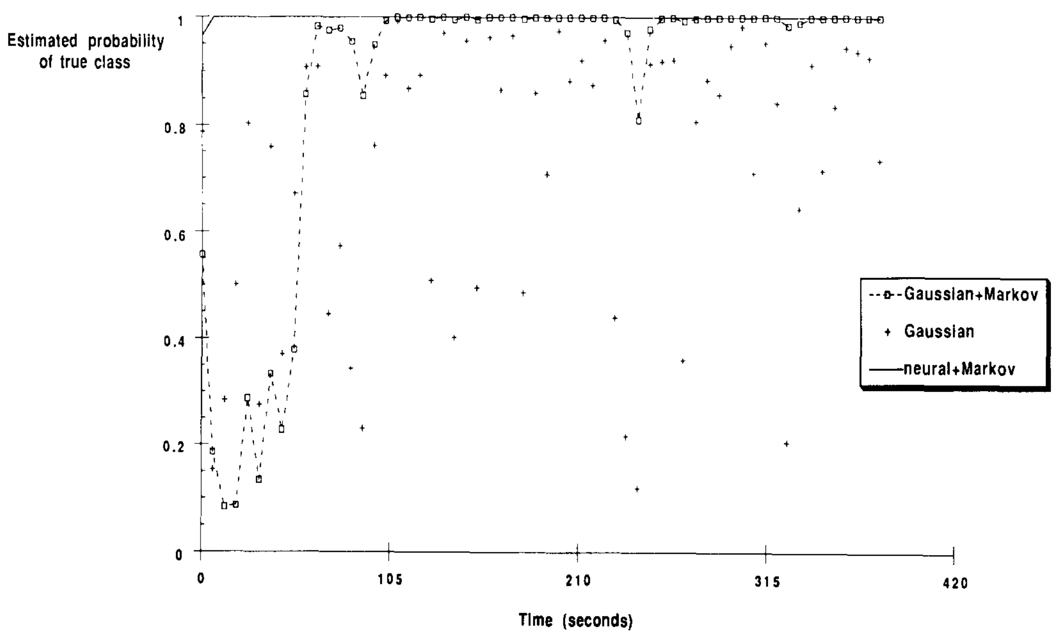


Fig. 5. Comparison of Gaussian–Markov and neural-Markov models: estimate of probability of true class (compensation loss) as a function of time.

Figure 4 corresponds to normal conditions and compares the neural model with and without the Markov processing. The instantaneous probability estimates from the neural model have a large variation over time and are quite noisy. This is essentially due to the variation in the sensor data from one window to the next, since as might be expected, signals such as motor current contain significant noise. In addition, a large glitch is visible at about 460 s. The neural model gives a low probability that the condition is normal for that particular window (in fact a large glitch such as this looks like a tachometer failure problem), however, the Markov model remains relatively unaffected by this single error. Overall, the stability of the Markov model is clearly reflected in this plot and has significant advantages in an operational environment in terms of keeping the false alarm rate to a minimum. Note that at any particular instant the neural network only ever assigns a probability of up to 0.8 or 0.9 to the true class. In contrast, by modelling the temporal context, the neural-Markov model assigns a much greater degree of certainty to the true class.

Figure 5 compares the performance of the Gaussian, Gaussian–Markov and neural-Markov models on detecting the compensation loss fault. The variation in the Gaussian estimates is quite noticeable. The Gaussian–Markov model combination, after some initial uncertainty for the first 90 s or so, settles down to yield reasonable estimates. However, the overall superiority of the neural–Markov model (the upper curve) is evident.

Figures 6(a) and (b) show the performance of the neural network classifier with and without the hidden Markov model monitoring the antenna for a total duration of about 1 h. Tachometer failure and compensation loss fault are introduced into the system after 14 and 44 min, respectively, each lasting roughly 15 min in duration. The difference in the quality of the two approaches is clearly visible in the figures and leaves little doubt as to the utility of the Markov method.

The results presented above clearly demonstrate the ability of a hidden Markov model to enhance the overall quality and reliability of a monitoring system's decisions. From a practical standpoint, the difference is significant: the non-Markov systems would not be reliable for actual operational use since they are too noisy and would have an unacceptably large false alarm rate. In contrast, the Markov-based system is a serious candidate for field implementation—we are currently evaluating such systems for installation in all new antenna designs. However, there are significant opportunities for further improvement in models of this nature. In the next section we discuss one such area where some progress is being made.

8. DETECTING NOVEL CLASSES

While the neural model described above exhibits excellent performance in terms of discrimination, there is another aspect to classifier performance which we must consider for applications of this nature; how will the classifier respond if presented with data from a class which was not included in the training set? Ideally, one would like the model to detect this situation. For fault diagnosis the chance that one will encounter such novel classes under operational conditions is quite

(a)

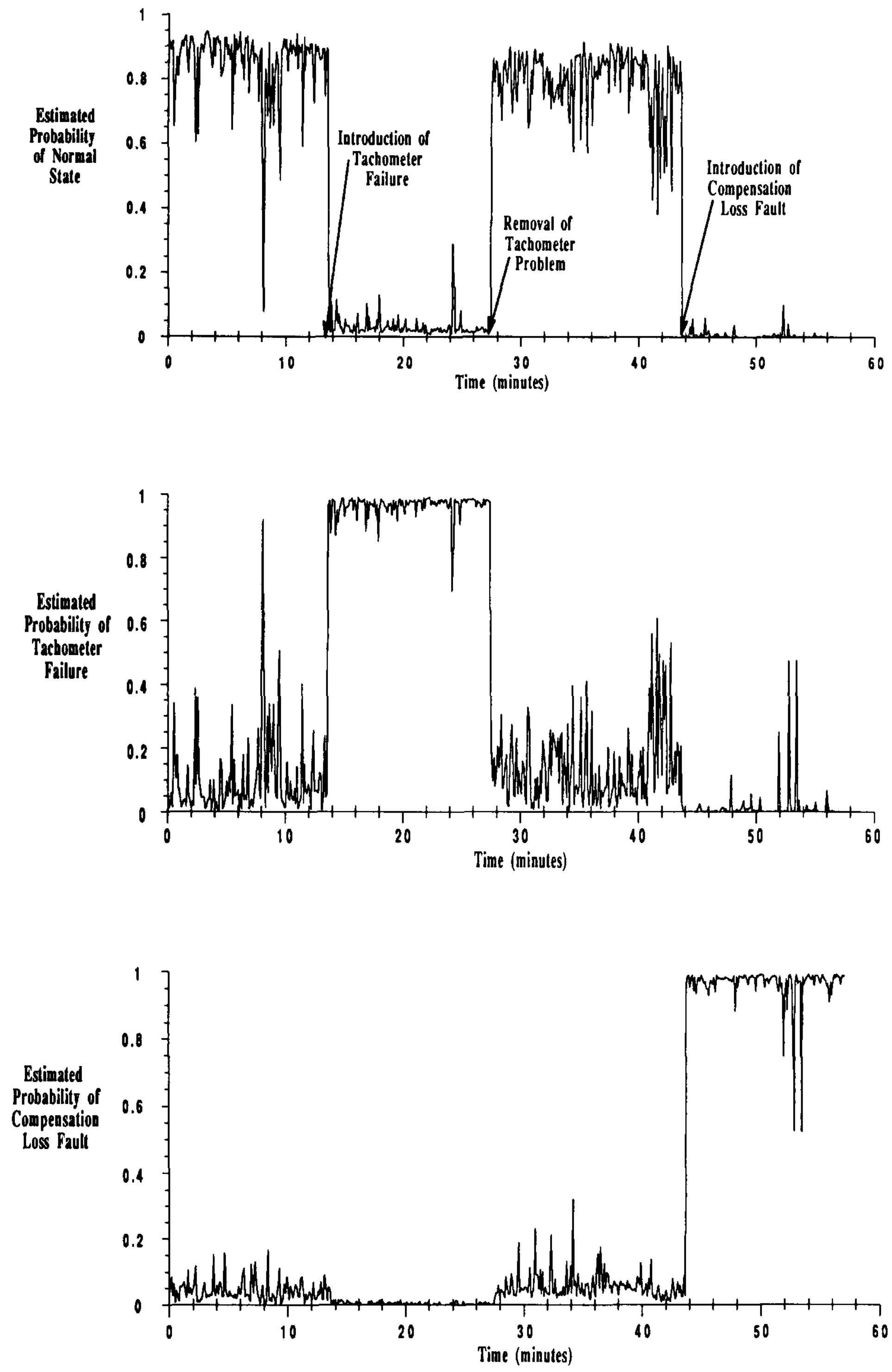


Fig. 6. Monitoring performance over 1 h with two faults introduced: (a) neural; (b) neural-Markov.

(b)

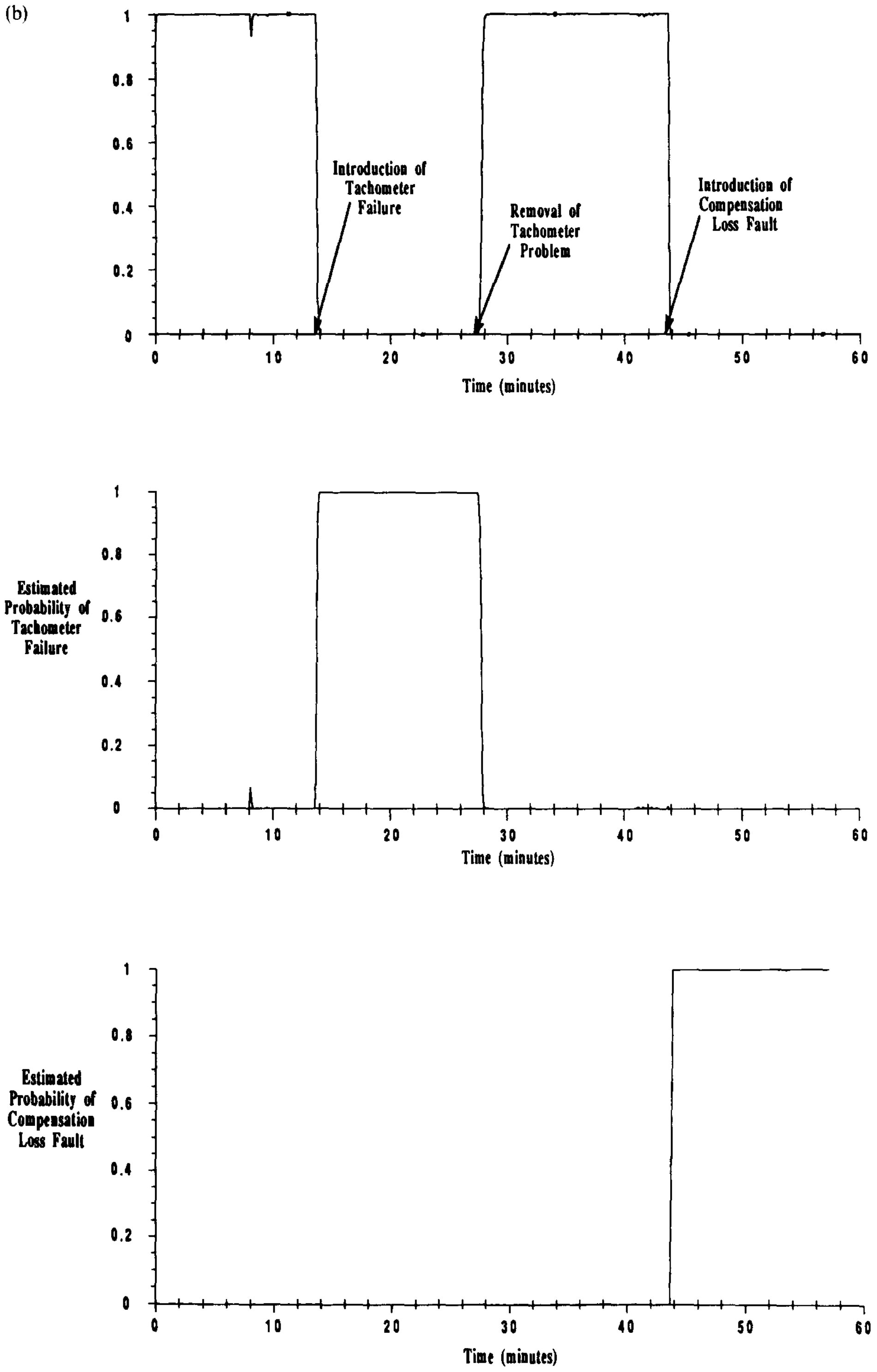


Fig. 6. (*Continued.*)

high since there is little hope of having an exhaustive library of faults to train on.

In general, with any non-parametric learning algorithm, there can be few guarantees about the *extrapolation* behaviour of the resulting model (Geman *et al.*[20]). The response of the trained model to a point far away from the training data may be somewhat arbitrary, since it may lie on either side of a decision boundary, the location of which in turn depends on a variety of factors such as initial conditions for the training algorithm, objective function used, particular training data, and so forth. One might hope that for a feedforward multi-layer perceptron, novel input vectors would lead to low response for all outputs. However, if neural activation units with non-local response functions are used in the model (such as the commonly used sigmoid function), the tendency of training algorithms such as backpropagation is to generate mappings which have a large response for at least one of the classes as the attributes take on values which extend well beyond the range of the training data values. Kramer and Leonard[21] discuss this particular problem of poor extrapolation in the context of fault diagnosis of a chemical process plant. The underlying problem lies in the basic nature of *discriminative* models which focus on estimating decision boundaries based on the differences between classes. In contrast, if one wants to detect data from novel classes, one must have a *generative* model for each known class, namely one which specifies how the data is generated for these classes (an example would be the Gaussian classifier used in the experiments in Section 7). Hence, in a probabilistic framework, one seeks estimates of the probability density function of the data given a particular class, $f(\boldsymbol{\theta}|\Omega = \omega_i)$, from which one can in turn use Bayes' rule for prediction:

$$p(\Omega = \omega_i|\boldsymbol{\theta}) = \frac{f(\boldsymbol{\theta}|\Omega = \omega_i)p(\Omega = \omega_i)}{\sum_{k=1}^{m} f(\boldsymbol{\theta}|\Omega = \omega_k)p(\Omega = \omega_k)}. \quad (14)$$

Generative models have certain disadvantages: they can perform poorly in high dimensions, and for a fixed amount of data may not be as efficient in terms of approximating the Bayes decision boundary as a purely discriminative method. We are currently investigating the use of hybrid generative/discriminative models in this context and have experimented with the use of kernel density estimators and mixture models—preliminary results are described in Smyth and Mellstrom[22] and Smyth.[23]

9. DISCUSSION

The hidden Markov method for on-line health monitoring proposed in this paper relies on certain key assumptions which may or may not be true for particular applications. In particular, for the purposes of this discussion we have assumed that:

(1) Faults are discrete in nature (i.e. they are "hard" failures rather than gradual degradation) and are known in advance.

(2) There is a fault library of classified data available in order to train the model.

(3) State-dependence is first-order Markov.

(4) Symptom estimates are statistically independent from one window to the next, conditioned on the classes.

However, it should be pointed out that these assumptions could potentially be relaxed and the model further refined. For example, a fault library may not be necessary if the symptom-fault dependence can be specified based on prior knowledge. Similarly, the assumption of independence of symptom estimates across windows is not strictly necessary—it makes the model much simpler, but could be included in equation (6) if such dependence is known to exist and can be modelled.

10. CONCLUSION

Effective modelling of temporal context in continuous monitoring applications can considerably improve the reliability and accuracy of a decision system. In particular, we have shown in this paper that hidden Markov models provide an effective method for incorporating temporal context in conjunction with traditional classification methods. The Markov model approach has the ability to significantly reduce the false alarm rate of a classification system by taking advantage of any time domain redundancy which may be present. The model was demonstrated on a real-world antenna fault diagnosis problem—the empirical results demonstrate clearly the advantage of the Markov approach. The data used in these experiments are available on request from the author. In general, the use of hidden Markov models for continuous monitoring seems to have promise: applications to other critical applications such as medical diagnosis in intensive care situations, nuclear plant monitoring, and so forth, appear worthy of further investigation.

Acknowledgements—The author would like to thank Jeff Mellstrom and Todd Chauvin for assistance in obtaining the experimental results described in this paper. In addition the author would like to thank the anonymous referees for their useful suggestions which improved the quality of the paper in terms of notation and presentation. The research described in this paper was performed at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration and was supported in part by DARPA under grant number AFOSR-90-0199 and the ONR under grant number N00014-92-J-1860.

REFERENCES

1. A. S. Willsky, A survey of design methods for failure detection in dynamic systems, *Automatica* 601–611 (1976).
2. R. Isermann, Process fault detection based on modeling and estimation methods—a survey, *Automatica* **20**, 387–404 (1984).
3. P. M. Frank, Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy—a survey and some new results, *Automatica* **26**(3), 459–474 (1990).
4. I. Bratko, I. Mozetic and N. Lavrac, *A Study in Deep and Qualitative Knowledge for Expert Systems*. MIT Press, Cambridge, Massachusetts (1989).
5. R. Davis, Diagnostic reasoning based on structure and behaviour, *Artif. Intell.* **24**(3), 347–410 (1984).
6. K. Ng and R. P. Lippmann, A comparative study of the practical characteristics of neural network classifiers and conventional pattern classifiers, *Advances in Neural Information Processing 3*, R. P. Lippmann, J. Moody and D. S. Touretzky, eds, pp. 970–976. Morgan Kaufmann, Los Gatos, California (1991).
7. S. M. Weiss and I. Kapouleas, An empirical comparison of pattern recognition, neural nets, and machine learning

classification methods, *Proc. Int. Joint Conf. on Artificial Intelligence 1989*, pp. 781–787. Morgan Kaufmann, Palo Alto, California (1989).

8. F. J. Pineda, Dynamics and control in neural computation, *J. Complexity* **4**, 216–245 (1988).
9. B. Pearlmutter, Learning state-space trajectories in recurrent neural networks, *Neural Computation* **1**(2), 263–269 (1989).
10. A. Waibel, T. Hanazawa, G. Hinton, K. Shikano and K. Lang, Phoneme recognition using time-delay neural networks, *IEEE Trans. Acoustics Speech Signal Process.* **ASSP-37**(3), (March 1989).
11. I. A. Papazoglou and E. P. Gyftopoulos, Markov processes for reliability analyses of large systems, *IEEE Trans. Reliability* **R-26**, 232–237 (August 1977).
12. L. R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, *Proc. IEEE* **77**(2), 257–286 (February 1989).
13. D. P. Siewiorek and R. S. Swarz, *The Theory and Practice of Reliable System Design.* Digital Press (1982).
14. L. Ljung, *System Identification—Theory for the User.* Prentice-Hall, Englewood Cliffs, New Jersey (1987).
15. R. L. Kashyap, Optimal feature selection and decision rules in classification problems with time series, *IEEE Trans. Inf. Theory* **IT-24**(3), 281–288 (1978).
16. M. D. Richard and R. P. Lippmann, Neural network classifiers estimate Bayesian a posteriori probabilities, *Neural Computation* **3**(4), 461–483 (1992).
17. J. Miller, R. Goodman and P. Smyth, On loss functions which minimize to conditional expected values and posterior probabilities, *IEEE Trans. Inf. Theory.* **39**, (1993).
18. E. Barnard and R. Cole, A neural net training program based on conjugate-gradient optimization, Oregon Graduate Centre Technical Report No. CSE 89-014, Oregon (1989).
19. M. J. D. Powell, Restart procedures for the conjugate gradient method, *Math. Programming* **12**, 241–254 (April 1977).
20. S. Geman, E. Bienenstock and R. Doursat, Neural networks and the bias/variance dilemma, *Neural Computation* **4**, 1–58 (1992).
21. M. A. Kramer and J. A. Leonard, Diagnosis using backpropagation neural networks—analysis and criticism, *Comput. Chem. Engng* **14**(12), 1323–1338 (1990).
22. P. Smyth and J. Mellstrom, Fault diagnosis of antenna pointing systems using hybrid neural networks and signal processing techniques, *Advances in Neural Information Processing Systems 4*, R. Lippmann, ed. Morgan Kaufmann, Los Altos, California (1992).
23. P. Smyth, Probability density estimation and local basis function neural networks, *Computational Learning Theory and Natural Learning Systems*, T. Petsche, M. Kearns, S. Hanson and R. Rivest, eds. MIT Press, Cambridge, Massachusetts (1993).

APPENDIX A. NEURAL NETWORK MODEL DESCRIPTION

We describe an example of a popular feedforward multilayer neural network model to familiarize the reader with the general notation and concepts. Figure A1 shows an example of a network. The input nodes are labeled n_i, $1 \le i \le K+1$, the hidden nodes are labelled h_j, $1 \le j \le H$, and the output layers are labelled o_k, $1 \le k \le m$. In general, there are $K+1$ input units, where K is the number of features. The extra node is always in the "on" state, providing a threshold capability. Similarly, there are m output nodes, where m is the number of classes.

The number of hidden units H in the hidden layer can influence the classifier performance in the following manner: too many and the network overfits the data, whereas too few hidden units leaves the network with insufficient representational power. The appropriate network size is typically

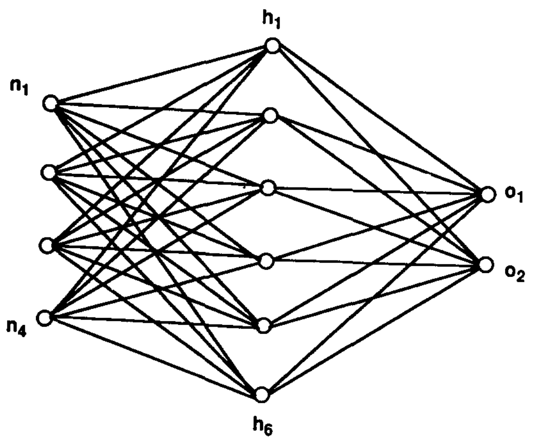


Fig. A1. Diagram of a "generic" feedforward neural network model with four inputs $\{n_1,\ldots,n_4\}$, a single hidden layer with six hidden units $\{h_1,\ldots,h_4\}$, and two outputs $\{o_1,o_2\}$.

chosen by varying the number of hidden units and observing cross-validation performance.

Each input unit i is connected to each hidden unit j by a link with weight w_{ij}, and each hidden unit j is connected to each output unit k by a weighted link w_{jk}. Each hidden unit calculates a weighted sum and passes the result through a non-linear function $F(\,)$, i.e.

$$a(h_j) = F\left(\sum_{i=1}^{i=K+1} w_{ij}a(n_i)\right)$$

where $a(n_i)$ is the activation of input unit i—typically, this is just a linear (scaled) function of the input feature. A commonly used non-linear function in the hidden unit nodes $F(x)$ is the so-called sigmoid function, defined as

$$F(x) = \frac{1}{1+e^{-x}}.$$

Output unit k calculates a similar weighted sum using the weights w_{jk} between the jth hidden unit and the kth output unit, i.e.

$$a_k = G\left(\sum_j w_{jk}a(h_j)\right)$$

where a_k is the activation of the kth output node. The function $G(x)$ can be chosen either as linear (e.g. $G(x) = x$) or as a non-linear function. For example for a classification problem such as that described in this paper the sigmoid function is used to restrict the range of the output activations to the range $[0, 1]$. A classification decision is made by choosing the output unit with the largest activation for a given set of inputs (feature values); i.e. choose class k such that

$$k = \arg\max_i \{a_i\}.$$

The network design problem is then to find the best set of weights such that a particular objective function is minimized on the N training data samples—the training data is in the form of input–output pairs $\{\mathbf{x}_j, y_j\}$, $1 \le j \le N$ where $\mathbf{x}_j$ is a feature vector and y_j the desired output (for simplicity of notation assume that we just have a single output model). Let $\hat{y}_j(\Omega, \mathbf{x}_j)$ be the network output for a particular set of weights Ω and input vector $\mathbf{x}_j$. The objective function is typically some metric on y_j and $\hat{y}_j$, whose mean value is estimated on the training data. Commonly used such objective functions include the mean-squared error

$$E_{\text{MSE}} = \frac{1}{N}\sum_{j=1}^{N}(y_j - \hat{y}_j(\Omega, \mathbf{x}_j))^2$$

and the cross-entropy error

$$E_{CE} = \frac{1}{N}\sum_{j=1}^{N} y_j \log\frac{y_j}{\hat{y}_j(\Omega, \mathbf{x}_j)} + (1-y_j)\log\frac{1-y_j}{1-\hat{y}_j(\Omega, \mathbf{x}_j)}.$$

From a maximum likelihood perspective the mean-squared error approach essentially assumes that the training data is perturbed by additive Gaussian noise, while the cross-entropy function assumes a multinomial distribution on the class labels. Despite these significantly different assumptions, for classification problems there appears to be little practical difference in terms of classification performance between these objective functions. For the experiments reported in this paper the mean-squared error objective function was used.

APPENDIX B. DESCRIPTION OF THE ANTENNA POINTING SYSTEM

Figure 2 is a block diagram of the elevation axis antenna drive subsystem (there is a corresponding azimuth axis drive for positioning the antenna in the azimuth axis). The elevation drive subsystem is a closed-loop control system that consists of a digital control computer, two 7.5 hp d.c. motors, two servo amplifiers, two cycloid gear reducers, two tachometers, and various electronic components for signal conditioning and servo compensation. The two forward tachometer/ amplifier/motor/gear paths operate in tandem to drive a large bull gear which is attached to the antenna structure (a 34 m dish plus supporting metal structure). Feedback control is provided by both rate feedback from each motor to its tachometer and a position feedback loop. The antenna position is estimated by an optical encoder and fed back to the antenna servo controller. The antenna servo controller is a microprocessor-based system which implemented a PI (proportional plus integral) control algorithm by integrating both the commanded position (which is a digital signal sent from a ground station control computer describing the desired position) and the actual position estimate. The digital portion of the control loop (the antenna servo controller) updates at a 50 Hz rate. The reconstruction filter and the loop compensation components are filters for signal conditioning and control loop compensation. Finally, the torque bias signal is a voltage measurement proportional to load torque which is fed back from the gears in order to share the torque between the two motors, reduce the effect of parameter variations between them and to effectively bias the cycloid gears away from non-linear regions of operation.

APPENDIX C. SPECIFICATION OF THE MARKOV TRANSITION MATRIX FOR THE ANTENNA POINTING PROBLEM

Training and test data under fault conditions were obtained by switching faulty components in and out of the servo control loop. Hence, for the purposes of this experiment, the two fault conditions were modelled as intermittent faults and fault transitions between these two states were allowed. The Markov transition matrix **A** was set as follows:

$$\mathbf{A} = \begin{pmatrix} 0.999 & 0.005 & 0.005 \\ 0.0005 & 0.99 & 0.005 \\ 0.0005 & 0.005 & 0.99 \end{pmatrix}.$$

This corresponds to a system MTBF of about 1 h and 7 min given the 4 s decision interval (using equation (11)). It also assumes that each fault is equally likely to occur and that the mean duration of each fault is about 6 min and 40 s. The initial state probabilities were chosen to be equally likely:

$$\pi(0) = (1/3, 1/3, 1/3).$$

The actual MTBF of the system under operational conditions was estimated from a problem database to be about 30 h if only hard faults are considered. However, if intermittent transient faults are also considered, the MTBF is effectively reduced to about 1 h—this estimate is based on empirical observations of the antenna in an operational tracking mode. Hence, while the self-transition probabilities of the fault states are set in a somewhat artificial manner for this experiment, the value chosen for a_{11} correlates well with the effective MTBF of the system.

As mentioned in Section 5.3, the state estimates of the model are relatively robust to changes in the values of the transition probabilities. For example, increasing $1 - a_{11}$ by an order of magnitude causes the estimates to be slightly less stable but does not introduce any additional false alarms, while reducing $1 - a_{11}$ by an order of magnitude causes no significant difference in the results other than the time for the model to switch from normal to a fault state (after a fault has actually occurred) increases from a single 4 s interval to two or three such intervals. It should be pointed out that the robustness of the method in general to misspecification errors in the transition matrix is a topic for further investigation.

About the Author—Padhraic Smyth received a first-class honours B.E. (Bachelor of Engineering) degree from University College Galway (National University of Ireland) in 1984, and the M.S. and Ph.D. degrees in electrical engineering, in 1985 and 1988, respectively, from the California Institute of Technology. From 1985 to 1988 he worked part-time as a Research Consultant to Pacific Bell in the areas of telecommunications switching systems and automated network management. In 1988 he joined the Communications Systems Research Section at the Jet Propulsion Laboratory, Pasadena, where he currently works as a Technical Group Leader on the theory and application of pattern recognition algorithms to a variety of problems of interest to NASA. Dr Smyth's research interests include statistical pattern recognition, machine learning, decision theory, source coding, information theory, telecommunications, and the application of probability and statistics to problems in artificial intelligence.