Improved Fault Detection in Cable Television Networks

by

Nicolaos P. Kourounakis
B.Sc., Aristotle University of Thessaloniki, 1994

A Thesis Submitted in Partial Fulfillment of the
Requirements of the Degree of

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

We accept this thesis as conforming
to the required standard

[redacted]

Dr. N. J. Dimopoulos, Supervisor (Department of Electrical and Computer Engineering)

[redacted]

Dr. K. F. Li, Departmental Member (Department of Electrical and Computer Engineering)

[redacted]

Dr. V. K. Bhargava, Departmental Member (Department of Electrical and Computer Engineering)

[redacted]

Dr. G. C. Shoja, Outside Member (Department of Computer Science)

[redacted]

Dr. E. G. Manning, External Examiner (Department of Computer Science)

Supervisor: Dr. Nikitas J. Dimopoulos

Abstract

This thesis presents a novel model-based fault detection system specifically designed to offer improved fault detection for the reverse pilot of cable television networks. The approach, however, is general and can be applied to a variety of large scale engineering plants. For the modeling part of the system, feedforward neural networks and statistical analysis techniques are employed. A description of these techniques as well as the fault detection methodology used is provided. Finally, this thesis presents the results of a large scale experiment on which the fault detection properties of the developed technique were evaluated and compared to those of limit checking fault detection.

Examiners:

[redacted]

Dr. N. J. Dimopoulos, Supervisor (Department of Electrical and Computer Engineering)

[redacted]

Dr. K. F. Li, Departmental Member (Department of Electrical and Computer Engineering)

[redacted]

Dr. V. K. Bhargava, Departmental Member (Department of Electrical and Computer Engineering)

[redacted]

Dr. G. C. Shoja, Outside Member (Department of Computer Science)

[redacted]

Dr. E. G. Manning, External Examiner (Department of Computer Science)

Table of Contents

Chapter 4 Cable Television Amplifier Networks

Chapter 5 Model-Based Fault Detection for the Reverse Pilot

Chapter 6 Result Analysis and Discussion

List of Figures

List of Tables

Acknowledgements

I wish to sincerely thank my supervisor Dr. Nikitas Dimopoulos for his exceptional guidance and support throughout the research and preparation of this thesis. I would also like to express my gratitude to my graduate advisor Dr. Kin Li for his friendly encouragement and advice over the course of my graduate studies. As well, I am grateful to Dr. Vassilis Dimakopoulos for assisting me in my first steps as a graduate student in the field of engineering, Dr. Elias Kosmatopoulos for his friendship and for assisting me in the early stages of my research work, and Stephen Neville for introducing me to the field of fault detection and for critically discussing with me some of the material in this thesis. Finally, I would like to thank my family and all my friends for being behind me and for keeping in touch.

It should be acknowledged that this research would not have been possible without the financial support from the Canadian Cable Labs Funds and NSERC.

Chapter 1:

Introduction

1.0 Introduction

Many modern engineering plants, such as petrochemical plants mills and communication networks, are complex, highly automated and generally must maintain a high throughput. Hence, the requirement for reliable and high quality operation is of paramount importance. During operation, faults may occur in various components of such plants, necessitating repair. The early identification and location of faults is critical to the proper operation of the plant. To this end, many engineering plants employ some form of operation monitoring and fault detection. The implementation of fault detection systems can offer significant benefits: from scheduling maintenances and increasing the plant's longevity, to avoiding plant breakdowns, costly material damages, or even major catastrophes with possible human fatalities. Fault detection is, therefore, used in many engineering plants where quality of operation, reliability and safety are considered critical. Communication networks are one example of such plants.

The rapid progress of our information-oriented society has created a trend towards the exploitation of all facets of existing communication technology. Vast amounts of information, from television images and sound, to email and computer data, are being transmitted at high rates. As a result of this communication explosion, the need to use communication networks to their fullest capacity (wireless or wire-linked, satellite or terrestrial) has increased. However, as communication rates increase, so does the associated cost of network faults and down-time.

In general, the construction of new physical networks is costly. Therefore, alternative methods for using existing wire-linked networks, such as power distribution and cable television networks for data transmission, are in demand. For example, cable television networks, which have traditionally offered only a one-way path for the delivery of cable

television services, are used today, to provide high speed duplex data paths, through which computer communication, internet connections, and other interactive services are made available to subscribers.

Most cable television networks employ some form of fault detection, typically in the form of limit checking (Chapter 3). In this method, real-time status measurements of monitored components are compared against preset fixed thresholds, and an alarm is generated whenever the status data exceeds these thresholds. However, this type of fault detection is not very accurate; generally, it generates a large number of false alarms and at the same time has a large probability of leaving actual faults undetected (See Chapter 3). The lack of reliability in the fault detection offered by limit checking was the motivation behind the work in this thesis.

The work presented in this thesis has been directed towards the design and software implementation of an improved fault detection system for the reverse pilot[1] of cable television networks. There were several objectives in developing this system:

- The fault detection properties of the system should be superior to those of the traditional limit checking system. The improvement must be measured in terms of false alarm rate and fault detection sensitivity.
- The system must be designed so that it is applicable in real-time and can be retrofitted into the existing monitoring system.
- The system must not rely on any *a priori* knowledge of the plant or on the type and number of fault modalities that may occur[2].

For the development of the fault detection system, a model-based approach was chosen. In this approach, a model of the reverse pilot is constructed, which is then used to detect any deviations of reverse pilot measurements from model predicted values. These deviations are deemed to signify changes in components' behaviour which may in turn be associated with the onset of fault events[3]. The modeling is implemented by using either a *neural network* or a *statistical modeling engine*.

---

1. The signal transmitted from the amplifier reporting its status to the head-end (See Chapter 3).
2. This information is generally not available and very difficult to obtain.

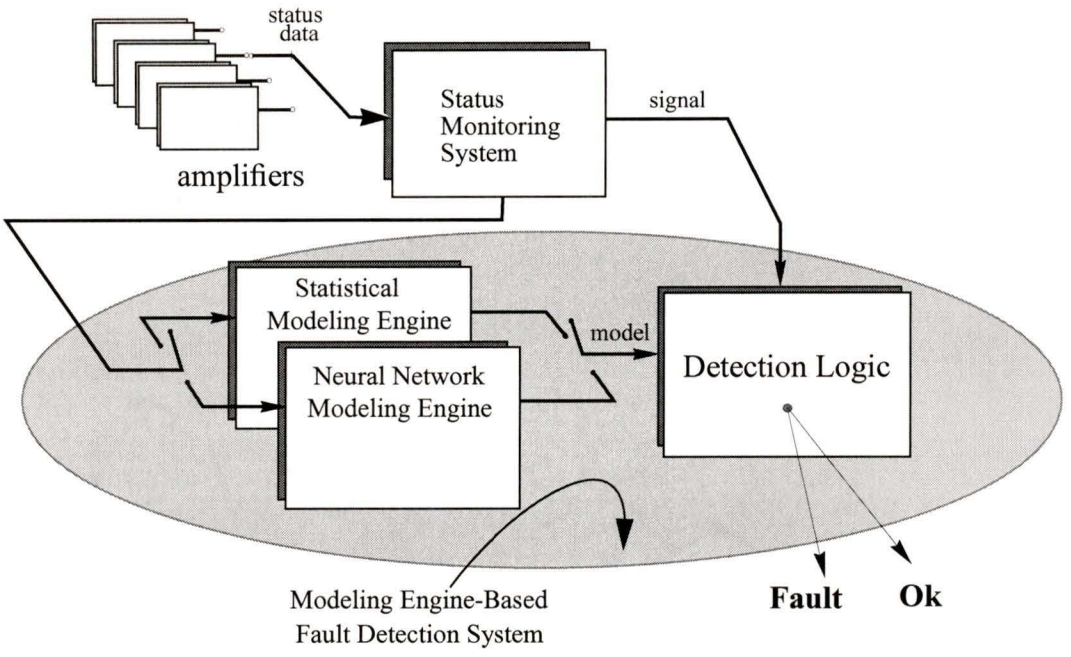

FIGURE 1-1.
The model-based fault detection system uses either a neural network or a statistical modeling engine to model specific monitored amplifier signals (such as the reverse pilot), and thereby, detect changes in these signals' behaviours.

The developed systems were calibrated and tested on an operational cable plant. To this end, data were acquired[4] and used from cable amplifiers of one of **Rogers Cablesystems Inc.** cable television networks. Figure 1-1 illustrates a general block diagram of the developed fault detection system.

---

3. Within this work a fault, and more precisely an observable fault is defined as an event which causes an observable deviation in the measured status data fields. A methodology which detects behavioural changes will therefore be able to detect the occurrence of these fault events. However, it should be noted that not all behavioural changes may be associated with a fault condition, hence some further processing of the detected behavioural changes is required. The nature of this processing though is outside of the scope of this work. Some discussion on how this processing is done for the domain of cable amplifier plants can be found in [30].

4. The amplifier monitoring and data acquisition is implemented by Rogers.

1.1 Thesis Outline

For the remainder of this thesis, the chapter by chapter outline is as follows:

Chapter 2 provides a brief introduction to artificial neural networks. Special attention is given to the characteristics of feedforward multilayer networks and to the backpropagation training algorithm. In addition, the concept of "initial weight space exploration" is introduced, and, finally several fields where artificial neural networks have been applied are outlined.

Chapter 3 presents a general description of the process of fault detection and some concepts associated with it. A short overview of the measurement-based and model-based fault detection techniques is also presented, After discussing their limitations to providing accurate fault detection in large scale engineering plants, a neural network fault detection method is presented.

Chapter 4 describes the basic structure of a cable television network and provides background information on the cable amplifiers used in the example cable distribution plant. Further, this chapter illustrates the monitoring and fault detection systems implemented in this plant, and describes in detail two of the monitored amplifier parameters, the reverse pilot and Temperature. The last sections of this chapter introduce preliminary concepts and techniques used for the fault detection of the reverse pilot.

Chapter 5 builds on the work and theory presented in the previous chapters, and describes in detail a novel model-based fault detection system that has been developed for the reverse pilot of cable television networks. This chapter also details the two modeling engines developed for the purpose of modeling the reverse pilot. Finally, the results of a large scale experiment testing the fault detection capabilities of the developed system under both modeling engines, are presented.

Chapter 6 presents the analysis of the results of the large scale experiment, and the evaluation of the modeling and fault detection properties of the systems developed in this work. In addition, it presents a comparison between the fault detection capabilities of the developed system and the limit checking detection system currently in use.

Chapter 7 presents a summary of this thesis and the conclusions of this work. As well, it provides a discussion of directions for future work.

1.2 Trademarks

The following is a list of trademarked words and terms used within this work:

UNIX is a trademark of American Telephone and Telegraph, inc.

Matlab is a trademark of MathWorks Inc.

Sun, Sparc, SparcStation, Sun OS are trademarks of Sun Inc.

HP, HP 715, HP 720 are trademarks of Hewlett-Packard Ltd.

C-Cor and C-Cor Electronics are trademarks of C-Cor Electronics Inc.

Rogers, Demon are trademarks of Rogers Cablesystems Ltd.

Chapter 2:

Neural Networks

2.0 Introduction

The following chapter will provide a brief introduction on neural networks. Artificial neural networks (ANNs) are loosely modeled after the neural structure of the mammalian brain. The basic processing element of an ANN is the artificial neuron, which abstracts the functions of a biological neuron. A brief description, therefore, of the biological neuron and its functions is presented. Several characteristics of an ANN will be discussed, such as activation functions, architecture and training algorithm. The basic backpropagation algorithm as well as some heuristics that improve training such as momentum, adaptive learning rate and initial weight space exploration are also briefly described. Finally, in section 2.7 some of the capabilities of ANNs are mentioned, as well as a several fields where ANNs have been applied.

2.1 Biological Neuron

The basic building block of the nervous system is the neuron, the cell that communicates information to and from the various parts of the body. All neurons are constructed from the same basic parts: a cell body called a soma which is the central part of the neuron, several spine like extensions of the cell body called dendrites, and a single nerve fiber called the axon that branches out from the soma and connects to many other neurons. The connection between the end of an axon branch and another neuron is called a synapse[37]. Figure 2-1 shows a simplified version of a biological neuron[46].

Active transport mechanisms move ions across the cell membrane to maintain a dynamical chemical equilibrium that constitutes the resting state of the neuron. When the soma's membrane is electrically stimulated the soma's internal state changes, causing a violent discharge which propagates along the axon away from the cell body to the synapses[46][9].

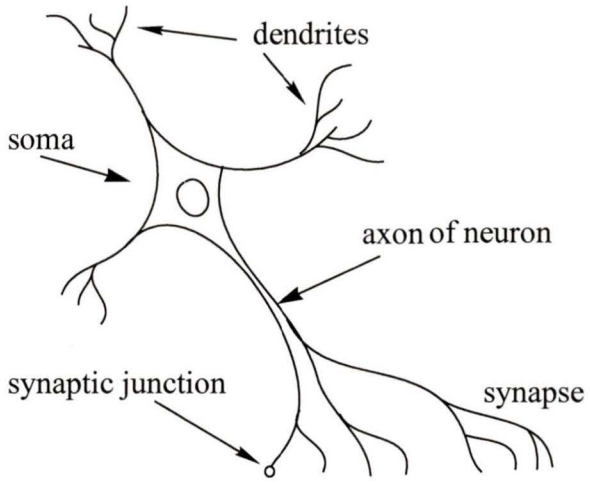

FIGURE 2-1.
Simplified version of the structure of a neuron

The body of a neuron collects the effects of its various input signals over its dendritic tree and over time. If these effects accumulate so that the resting potential exceeds a certain threshold, the neuron fires, initiating the whole sequence of events again. A synapse is termed *excitatory* if the total induced resting potential is positive and the influence of the synapse tends to activate the postsynaptic neuron. If the induced change of the resting potential is negative, the synapse is termed *inhibitory.* The influence of a synapse upon the cell body depends on its inherent strength. The strength of the synaptic connection can be adjusted [9].

The mammalian nervous system is constructed of billons of neurons with the axon from one neuron branching out and connecting with the dendrites of thousands other neurons. All the neurons interconnected by axons and dendrites create a neural network. The model neuron, illustrated in Figure 2-2(a), can be considered a threshold unit, a processing element that collects inputs and produces an output only if the sum of its input exceeds an internal threshold value. Artificial neural systems are created by interconnecting many of these simple "neurons" into a network.

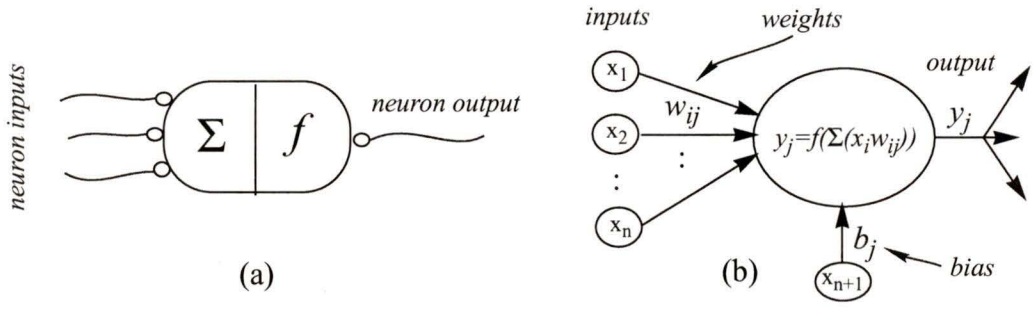

FIGURE 2-2.
(a) The model neuron is a threshold unit. (b) The output of the neuron is a function of the sum of its weighted inputs.The bias can be considered as the weight associated with an input (x_{n+1}) fixed at 1.

2.2 Artificial Neural Networks (ANN'S)

An Artificial Neural Network has a certain number of simple processing elements called neurons, which are connected to other neurons by means of directed communication links. Each link has an associated weight, which collectively encode information that is used by the network to solve a problem [34].

Just like the biological neuron, each "artificial" neuron collects the information that has been sent down its incoming connections and produces a single output value. This output is propagated through the neuron's outgoing weighted connections to other receiving neurons. The output value of the neuron shown in Figure 2-2(b), is a function of the outputs of the preceding layer, $F_x = (x_1, x_2, \ldots x_n)$ and the weights from layer F_x to neuron j, $W_j = (w_{1j}, w_{2j}, \ldots w_{nj})$ (see also Figure 2-4 in page 11). Mathematically the output of a neuron is a function of inputs and its weights:

$$y_j = f(F_x, W_j) = f\left(\sum_{i=0}^{n} x_i w_{ij} + b_j\right) \tag{2.1}$$

where f is a nonlinear function called *activation function* [36],
b_j is a input for neuron j called the *bias* [9].

The bias term can be considered as the weight of an input with its value clamped to one, $x_{n+1} = 1$ and $b_j = w_{(n+1)j}$. This way, Eq. 2.1 can be rewritten in a more general form:

$$y_j = f\left(\sum_{i=0}^{n+1} x_i w_{ij}\right) \tag{2.2}$$

2.3 Neural Network Characteristics

A network is characterized by its architecture, training algorithm and activation function. In the following sections these are briefly discussed.

2.3.1 Activation Functions

The activation function maps a neuron's input to a prespecified range. Some functions employed by the majority of neural networks are:

1. The *Linear* function:

$$f_l(x) = ax,\ x\varepsilon\Re \text{ and } a > 0 \tag{2.3}$$

2. The *Hard-limiter* function:

$$f_h(x) = \begin{cases} 0, x < \theta \\ 1, x \geq \theta \end{cases} \tag{2.4}$$

3. The nonlinear *Sigmoid* function:

$$f_s(x) = \frac{1}{1 + e^{-\sigma x}} \tag{2.5}$$

4. The *Hyperbolic Tangent* sigmoid function:

$$f_t = \frac{2}{\left(1 + e^{-2x}\right)} - 1 \tag{2.6}$$

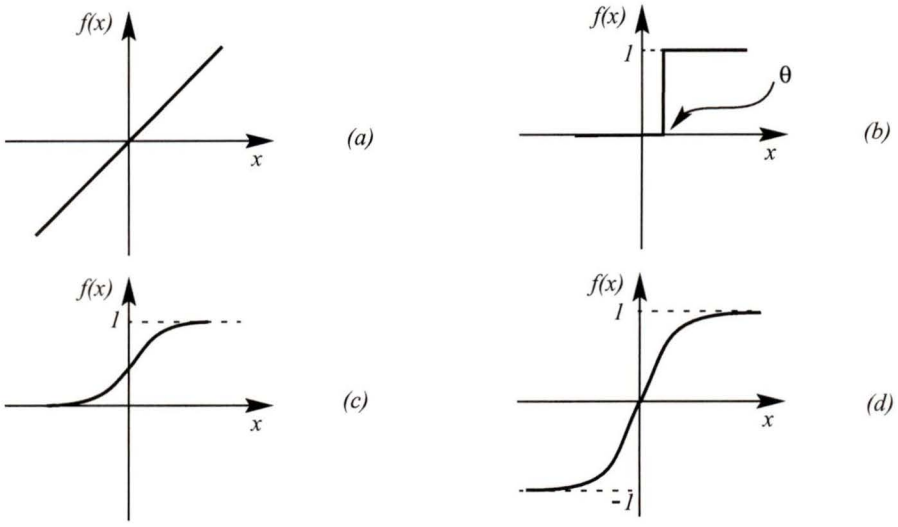

FIGURE 2-3.
Four functions widely used as activation functions: Linear function (a), Hard-limiter (b), Sigmoid (c), Hyperbolic Tangent (d).

2.3.2 Architecture

The architecture describes the pattern of connections between neurons. An ANN is termed "recurrent" if its architecture allows feedback from a neuron's output to its own input (self connection), or into the inputs of neurons within the same or previous layers. Figure 2-4(a) shows a possible connection scheme for a recurrent neural network. In contrast to recurrent neural architectures, "feedforward" networks can propagate activations only in the forward direction.

The main focus of this thesis is in the use of feedforward neural networks. Therefore, in the following sections feedforward networks are described in more detail, and aspects that improve their performance are discussed.

2.3.3 Feedforward Neural Networks

A typical feedforward ANN consists of at least three layers: an input layer F_x which copies the input values and associates them with weights, feeding the neurons in the hidden layer, one or more hidden layers F_y, which convert their receiving signal to an activa-

tion and pass it to the next layer, and an output layer F_z, that provides the outputs of the ANN. Figure 2-4(b) shows a typical connection scheme for a multi-layer feedforward network, where $\boldsymbol{W}$ are matrices representing the weights between layers.

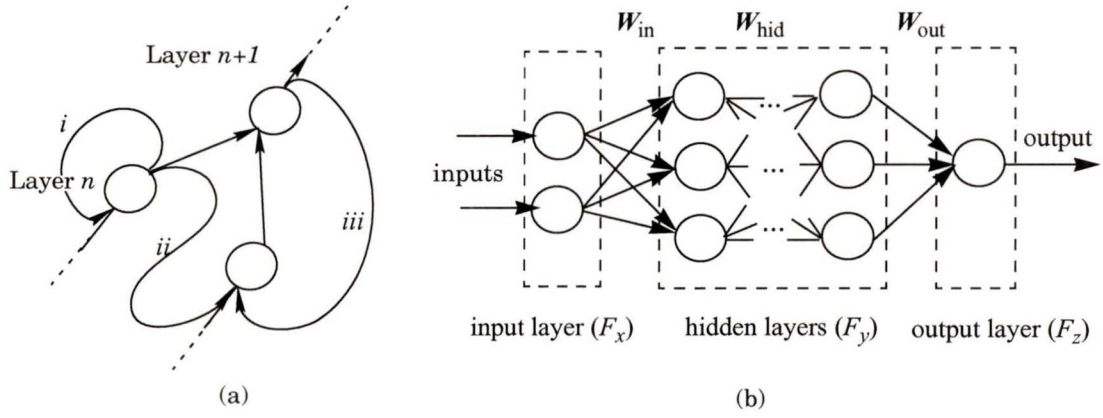

FIGURE 2-4.
a) A recurrent network may have interconnections between a neuron's output and: *i)* its own input, *ii)* neurons of the same layer, *iii)* neurons of previous layers. b) A feedforward network may only have interconnections in the forward direction.

For an ANN used in a certain application, the number of input and output neurons are set to match the number of the external inputs and outputs. The determination of how many hidden layers and neurons are required for each application is currently an open research problem. It is though well understood that the functionality of an ANN depends highly on the number of its hidden layers as well as on the number of neurons in those layers. In general, "large" neural networks with many hidden layers and neurons are more powerful and can be used to solve more complex problems. The trade-off in using large networks is the long training time required and the possibility of poorer error convergence, since the search error space is more complex. In addition, using more hidden units than required suppresses the generalization[1] capability of the network and often leads to poor interpolation (phenomenon of overfitting) [7]. Although some algorithms which help

1. By generalization we define the capability of the network to produce correct or nearly correct input-output relationships for input/output patterns that the network was never trained on [18].

determine appropriate structures for a given task exist, the most common way is to experimentally determine the optimal number of hidden layers and neurons through trial and error.

2.3.4 Training Algorithm

The most appealing quality of neural networks is their ability to learn. Learning is defined, in this context, by P. Simpson [36] as: *"...change in connection weight values that result in the capture of information that can later be recalled"*. There are several ways that multilayer neural networks can have their connection weights adjusted to learn an input - output space mapping. Basic backpropagation is currently the most popular learning rule used in supervised training [43]. In supervised training, an input/output set of pairs is presented to the network and the interconnection weights of the neural network are updated (by a recursive algorithm) to minimize the difference between the output predicted by the NN and that observed in the training set [43].

Typically, training a neural network involves several iterations called "epochs". Initially the weights of the network are set to small random values and an error threshold is chosen. During each epoch, a group of training patterns (training set) is presented to the network, and the error, e, between the outputs of the network, $\hat{y}$, and the desired response y_d, is calculated. The error can be calculated either after each epoch, or after each training pair is presented to the network. In the first case, a measure of the error e that is commonly used is the mean square error between all N neural network outputs $\hat{y}_n$ and the desired outputs y_{d_n} of the training set:

$$e = \frac{1}{2N} \sum_{n=1}^{N} (\hat{y}_n - y_{d_n})^2 \tag{2.7}$$

The error expression in the above Eq. 2.7 is general and can be used as well when the error is calculated after each pattern is presented. By setting therefore N=1 we get:

$$e = \frac{1}{2}(\hat{y}_n - y_{d_n})^2 \tag{2.8}$$

At each epoch, a recursive algorithm is used to adjust the weights iteratively starting at the output neurons and working back through the hidden layers to the input layer, so that the error will be smaller in future epochs. The new values of the weights $w_{ij}(t+1)$ at time $t+1$ are given by:

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x_i \tag{2.9}$$

Hence, the resulting weight adjustment is:

$$\Delta w_{ij} = w_{ij}(t+1) - w_{ij}(t) = \eta \delta_j x_i \tag{2.10}$$

where $w_{ij}(t)$ is the weight from neuron i to neuron j at time t,

Δw_{ij} is the weight "update" for connection ij at time t,

x_i is the input of neuron j from neuron i at time t,

η is a gain term called *learning rate*,

and δ_j is an error term for neuron j at time t.

Essentially δ_j represents the contribution of the j^{th} neuron to the total error.

For an output neuron, δ_j is given by:

$$\delta_j = f_j'(s_j)\, e_j \ , \tag{2.11}$$

where $s_j = \sum_i^p w_{ij} x_i$ is the sum of all p weighted inputs x_i of neuron j,

e_j is the error for neuron j as defined in Eq. 2.7 or Eq. 2.8,

and f_j' is the derivative of the activation function[2] for neuron j.

For a hidden layer neuron, δ_j is given by [9]:

$$\delta_j = f_j'(s_j) \sum_k \delta_k w_{jk} \tag{2.12}$$

where k is over all neurons in the layer above neuron j.

The term δ_k of Eq. 2.12, requires knowledge of the error e_k[3] for all the neurons that are directly connected to the hidden neuron j and lie to the immediate right of it. The term w_{jk}, consists of the weights associated with these connections [27].

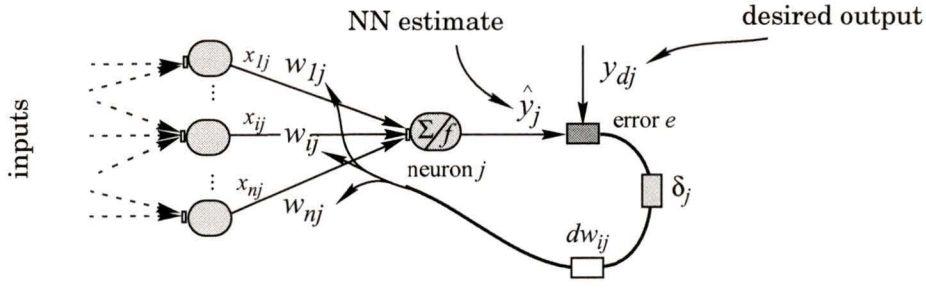

FIGURE 2-5.
Updating the input weights of a output neuron, using the backpropagation training algorithm.

A schematic representation of the above training algorithm for the case of an output layer neuron is shown in Figure 2-5. Typically, training terminates when the error is below the pre-specified threshold or the maximum number of epochs has been reached [27].

---

2. It should be noted that backpropagation requieres that semilinear (i.e non-decreasing and differentiable) activation functions be used [9].

3. Generally, for hidden layers, the error term e_k is replaced by the δ's of the previous layer.

The backpropagation learning algorithm performs essentially a gradient descent minimization process[4] within the error surface [7]. The global minimum is defined as the theoretical solution which yields the lowest possible error. The error surface in most problems is quite irregular, with numerous 'valleys' and 'hills'. This phenomenon is even stronger when non-linear activation functions are used in multilayered networks, which tend to introduce many local minima in the error surface [7]. Consequently, as gradient descent is performed, it is possible for the network solution to become trapped in one of these local minima which is not the best overall solution (Figure 2-6).

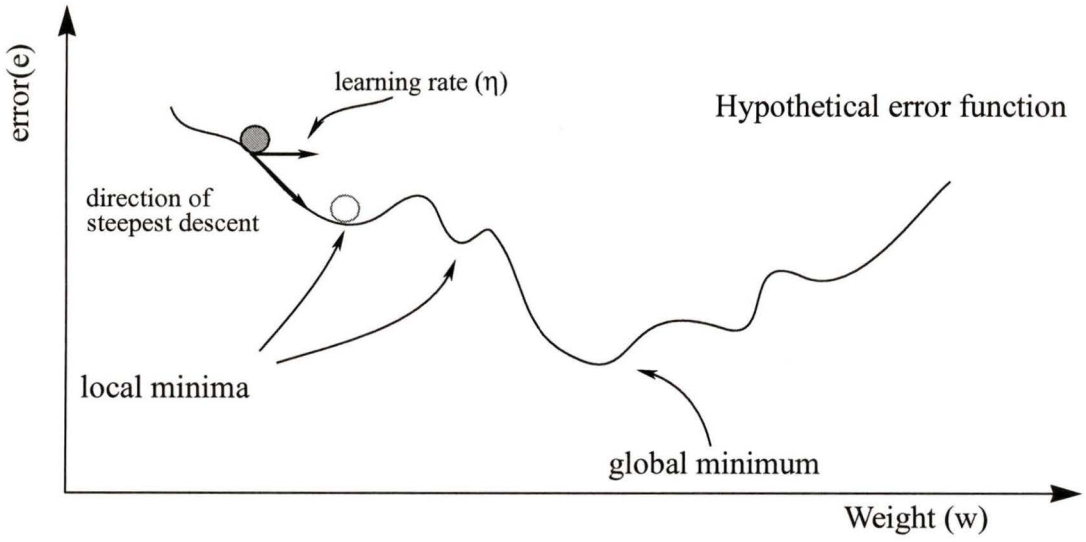

FIGURE 2-6.
Hypothetical two dimensional error function.

It should be noted though, that settling in a local minimum, is not necessarily an undesirable outcome of the search. Depending on how close the local minimum is to the global minimum and how low an error is required for the specific application the network is trained for, the solution may be , and often is, acceptable.

---

4. The gradient descent minimization process is defined as a search for the global minimum along the steepest vector of the error surface [9].

2.4 Improving Training

The basic backpropagation algorithm converges slowly and the learning process may stick on local minima [7]. Several heuristics, though, can be added to help to some extent in overcoming these problems. Adaptive learning rate and momentum, as well as initial weight space exploration help in most cases to improve the training process and are discussed in the following sections.

2.4.1 Adaptive Learning Rate

The speed of the descent is governed by the learning rate (η). If the learning rate is too high the network takes "steps" that are too large, resulting in an inefficient and possible less optimal search. In some cases, the solution search may oscillate over an error minimum without converging, as shown in Figure 2-7(a). In other cases, a large learning step might cause the search to "skip over" (overshoot) a low error minimum, resulting in higher error solutions (Figure 2-7b).

On the other hand, a learning rate that is too small will result in very long training periods (Figure 2-7c). There is no straight forward way of picking an optimal constant learning rate for non-linear networks. Choosing a learning rate large enough to ensure an acceptable training duration, and yet small enough to produce short steps when needed is not an easy task [7].

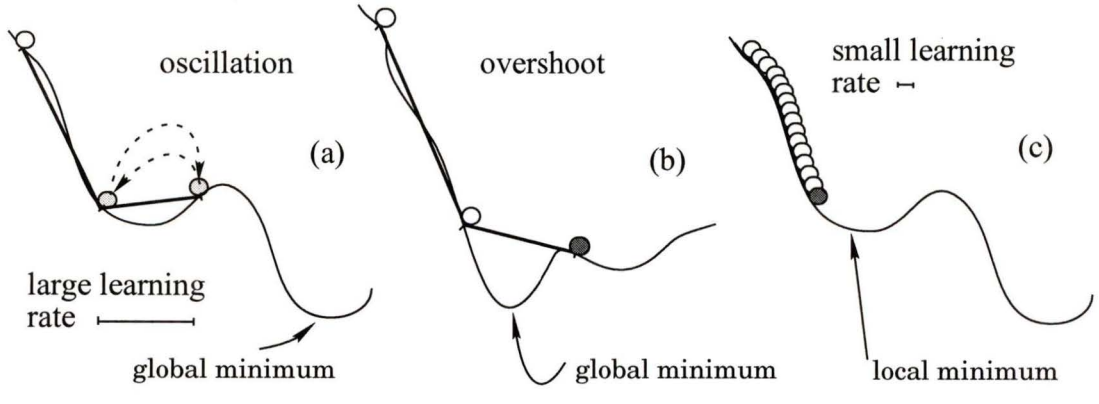

FIGURE 2-7.
Example of a two dimensional error surface search with: (a) and (b) large learning rate, (c) small learning rate

An approach that deals with the aforementioned difficulties associated with a constant learning rate, is to have a variable learning rate. A variable (adaptive) learning rate attempts to keep the descent step size as large as possible while assuring stable learning through adjusting the learning rate in response to the local error surface [7] (Figure 2-8). Using this technique, the learning rate increases linearly as the error decreases. If η becomes too large, and the error increases by more than a default error ratio, the adaptive learning rate suddenly decreases with an exponential rate [7].

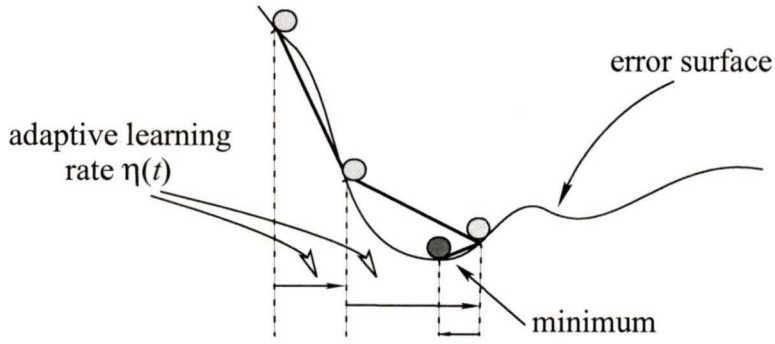

FIGURE 2-8.
Efficient gradient descent using adaptive learning rate. The search responds to the complexity of the error surface

The adaptive learning rate can be expressed by:

$$n(t+1) = \begin{cases} \eta(t) + \Delta_\eta & if \quad r < \theta \\ \eta(t) - \phi\eta(t) & if \quad r > \theta \end{cases} \tag{2.13}$$

where Δ_η is a small increment for the learning rate,

ϕ determines the step which the learning rate should decrease,

r is the ratio $r(t) = \dfrac{e(t)}{e(t-1)}$,

and θ is a pre-specified error ratio.

2.4.2 Momentum

Another way to improve the performance of the gradient descent search is to allow the network to respond not only to the local gradient but also to recent trends in the error surface. This can be done by altering the search algorithm to include a term called momentum which "carries" a proportion of the last weight change. The momentum term is applied in Eq. 2.10 as:

$$\Delta w_{ij} = \eta \delta_j x_{ij} + \Delta_m \tag{2.14}$$

where $\Delta_m = c_m (w_{ij}(t) - w_{ij}(t-1))$

and c_m is the momentum constant, $0<c_m<1$.

That is, the weight change carries along some momentum to the next iteration. The use of the momentum term tends to mitigate the effects of the fluctuations of the error surface, acting like a lowpass-filter [13]. With momentum, a network can slide through shallow local minima, continuing the search towards a lower error minimum Figure 2-9.

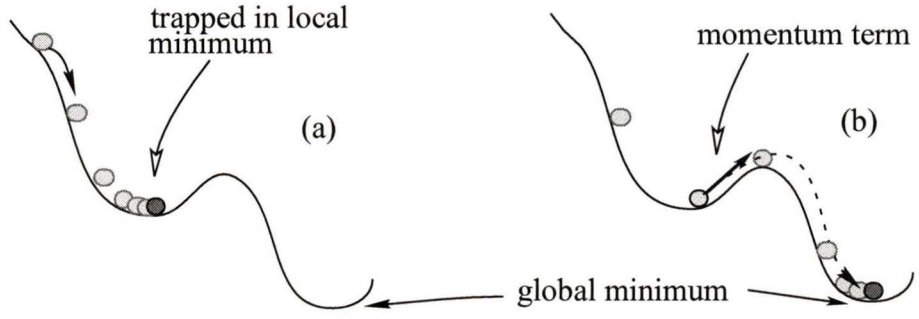

FIGURE 2-9.
(a) Without the momentum term, the optimization stops at a local minimum.(b) Momentum will allow the optimization to escape shallow local minima.

2.5 Additional Heuristics

Additional heuristics are available which improve gradient descent optimization. In some training algorithms an acceleration term is used, which is calculated from the second derivative of the activation function. The addition of the acceleration term in Eq. 2.14 has

been proven to speed the descent when a minimum is near [35]. Another approach is to alter the sigmoid activation function to ensure that values near 0 or 1 do not occur. This will speed training by eliminating extremely small weight updates [47]. Stochastic techniques which allow movement in directions other than steepest descent offer a solution when the network is trapped in undesired local minimuma [20]. Finally, perhaps the most desired improvement will come by selecting the initial weights so that they facilitate training[38].

2.5.1 Initial Weight Selection

As mentioned in Section 2.3.4, the first step in the process of training is to choose the initial training weights. It has been observed that the "success" of the training depends largely on the starting point of the usually complex multi-dimensional error surface on which the minimal error weight combination search (optimization) is performed. Multilayer ANNs, depending on the number of neurons and their structure, introduce tens to hundreds of weights with corresponding error surfaces which are multidimensional[5] and highly irregular. In addition, since $w \in \Re^6$, there is an infinite number of weight combinations, which makes the selection of a "good" set of initial weights, a very difficult task.

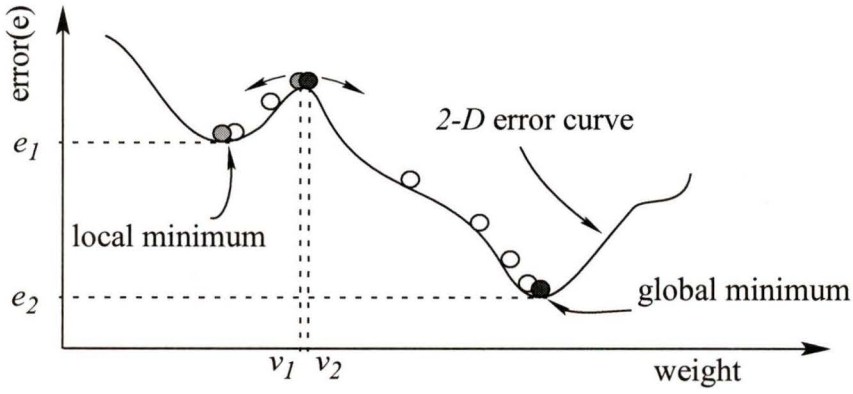

FIGURE 2-10.
Slightly different initial conditions (weight values) will cause the search to return different error solutions.

---

5. The error surface is $n+1$ - dimensional for a network with n weights and biases.
6. Domain of Real numbers.

Usually, trainings are very sensitive to the selection of the starting point (weight combinations) of the search. Slightly different initial weights may result in completely different solutions. This fact is demonstrated in Figure 2-10, by using as an example a *2-D*[7] error curve. Two neighboring starting points are selected, corresponding to weight values v_1 and v_2. The search initiated with weight value v_1 will cause the network to produce a solution with an error e_1 which is a high error local minimum solution, while weight value v_2 will allow the network to reach a lower error e_2 which is the global minimum in this case. In higher order error functions with more complex surfaces the search tends to be much more sensitive to the selection of the initial weights.

As noted in previous paragraphs, the dimensionality of the weight space is of the order of the number of the weights of the network, making the search for a "good" initial weight combination very complex and difficult. As such, most research efforts on initial weight selection have been focused on the distribution of the weights rather than their actual value. It has been proven that in most cases some randomness in the initial weights, which are typically set to the range $[-1, 1]$, improves final network performance [38].

2.5.2 Initial Weight Space Exploration

In order for neural networks to be used successfully in automated real time applications, trainings must be fast and reliable in the sense that they must be capable of exploring efficiently the given error surface, producing an acceptable error solution. By enhancing the back propagation training with modifications in the steepest descent algorithm presented in Sections 2.4.1 and 2.4.2, one can improve the training of the network for a given set of initial weights. There is no way, however, to know *a priori* which set of initial weights will produce the lowest error (best) solution.

Ideally, a network would be trained several times with several different sets of initial weights, and the weight combination of the training with the lowest error would be selected. Obviously, this approach is very time consuming and computationally expensive since several "full" trainings of usually thousands of epochs must be performed. An alter-

---

7. It should be noted that, for simplicity in this thesis, only *2-D* error functions are illustrated which correspond to single weight neural networks.

native to the above "error space exploration" would be a "partial" error space exploration, during which the network is trained only for a few epochs for each set of initial weights. The weights that produce the lowest error could be then used to further train the network for the whole number of epochs. This approach is much less time - and computationally - expensive. Although it is not guaranteed that initial weights that produce lower training errors after the first few epochs will eventually produce a better solution after the whole training is over, the partial search gives an indication of which weight combinations will result in poor training and therefore should be avoided.

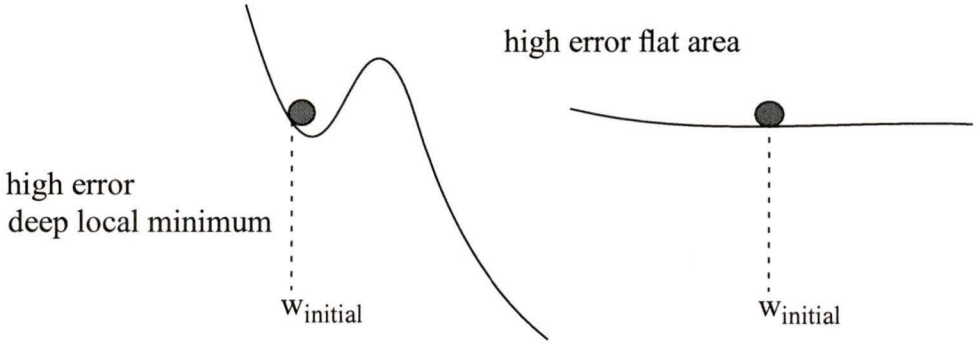

FIGURE 2-11.
Starting the search near a high error local minimum or a flat area will result in poor training.

Generally, there is no way to ensure that a multilayer NN which is trained with back-propagation will train to an acceptable error level and won't get trapped in some undesired (high error) local minimum. Using the partial error space exploration technique, we can ensure at least that the network training will not start (or get trapped early) in a high error local minimum, in a saddle point[8], or in very flat areas (Figure 2-11).

8. A phenomenon called "premature saturation" corresponds to a saddle point in the error surface, and refers to the situation where the training error remains almost constant for some period. It is different from a local minimum, since the error decreases after the end of this period [18].

2.6 Neural Networks and Fields of Application

Initially, Artificial Neural Networks were used as a research instrument for understanding and simulating biological neural networks. However, over the past few decades, their problem solving capabilities have been revealed and applied to a number of diverse fields. Today, ANNs are becoming increasingly important in areas such as, medical diagnosis, ecological modeling and currency trading.

Feedforward ANNs are *universal function approximators* [27]. They have been proven to be capable of emulating the input/output maps of arbitrary non-linear functions to a desired degree of accuracy, provided sufficiently many hidden layers and neurons are available and their activation functions have non-constant derivatives. The drawback of this architecture, which can be efficient on a variety of problems, is its difficulty to incorporate time domain information and thus deal with problems which temporal aspects are important (dynamic problems). Another type of neural networks, recurrent ANNs, due to their feedback interconnections, can store information about time, and therefore are more adept at performing dynamic mappings [31].

ANN's learn the interrelationships between the parameters of a problem by looking at examples of different situations. Hence, they are capable of offering a solution to the problems of modeling poorly specified processes. When using ANNs there is no requirement of making assumptions about the nature of the mapping to be performed. They only require behaviour history of the system as input.

ANNs are sometimes favored over other artificial intelligent or traditional analytic techniques because of their inherent parallelism and their proven ability to learn the essential underlying theme that is common to a group of problems. Obviously, the range of applications for which they could be considered is large. Some engineering areas for example, where neural networks are applied are:

- real-time optimization
- database analysis
- adaptive signal processing
- image data compression

- voice and handwritten numeral recognition
- process monitoring and process control
- process modeling and system identification
- fault detection and diagnosis

2.7 Summary

The structure of Artificial Neural Networks is based on the structure of the biological nervous system. A neural network is characterized by its architecture, activation function and training algorithm. It can be recurrent or feedforward. An ANN has to be trained in order to learn a certain input-output relationship. During training, the networks interconnection weights are adjusted according to a training rule. Back propagation is the most popular training algorithm for feedforward networks. When using this training method, the derivative of the activation function is calculated, so the activation functions must be differentiable. Commonly used activation functions are the semilinear sigmoid functions. To achieve better learning and speed up the process, we can use adaptive learning rate and momentum. In addition, initial weight space explorations can be performed to improve the training of the networks, by avoiding initial weights that "trap" the network into high error solutions.

Feedforward multilayered networks are suitable for use in many different fields, such as medicine, economics, ecology and engineering. In this thesis, a feedforward neural network is used to detect faulty components in a large communication network. More specifically, a Neural Network modeling engine (NNME) will be presented, which successfully models the behaviour of the reverse pilot of cable television amplifiers for the purpose of fault detection.

Chapter 3:

Fault Detection

3.0 Introduction

This chapter presents a general description of the process of fault detection as well as some basic concepts that are associated with it, and gives a short overview of two fault detection methods commonly used in large scale engineering plants, namely these of traditional model-based and measurement-based techniques. After discussing the limitations of these techniques, an alternative fault detection method is presented which can be used in poorly understood[1] plants where classical analytical modeling techniques cannot be applied. This method uses artificial neural networks and has successfully been implemented in engineering plants.

3.1 Fault detection (FD)

As modern automated engineering plants are becoming more and more complex, with highly sophisticated control algorithms, there is a growing demand for fail-safe operation. One way this can be achieved is by improving the individual reliability of the functional units by adding physical redundancy. Another way fail-safe operation of engineering plants can be accomplished, is by implementing efficient plant monitoring with early fault detection and diagnosis algorithms.

Large scale engineering plants are designed to operate with a minimum of human intervention, and implement highly automated supervisory systems. Consequently, there is a trend towards automating as well the operations of fault detection and diagnosis. These operations can be integrated with system design and implementation, or retro-fitted into the existing system. Automated fault detection has become a basic requirement in many domains where quality improvement, cost reduction, safety, reliability and longevity of the system are considered critical.

---

1. In this context, "poorly understood" is used to denote the lack of analytical information about the plant which can be used to construct an analytical model.

In accordance with [11][14][15][21] the concepts of fault detection, fault diagnosis, fault, and failure are defined below.

Fault detection is the task of detecting the existence of faults in a system as early as possible before the performance[2] of the system starts showing signs of degradation; the task of *fault diagnosis* is to find the root causes of the faults. In this context, a *fault* is defined as a non-permitted deviation of a characteristic property caused by any kind of malfunction in the plant, which leads to a reduction in the overall system performance [15] [21]. A *failure,* on the other hand, refers to the complete inability of the equipment or the process to fulfil their intended purpose [11]. Faults can occur in any component of the system, and if unchecked may endanger the integrity of the system and lead to failures [14][21]. Fault modes that can occur may be classified as *abrupt* faults, i.e. step-like changes, or as *incipient* (slowly developing) faults, i.e. bias or drift [14]. In order to avoid the consequences caused by malfunctioning components, which in some cases can be catastrophic, faults have to be detected early enough so that they can be mitigated by system reconfigurations or repairs.

3.2 Fault Detection Schemes

Fault detection approaches may be broadly classified as *measurement-based* or *model-based*. In measurement-based approaches, certain signal features such as magnitude or variation are monitored. Any significant deviation of these monitored features from a nominal level is marked as a fault. Although the measurement-based techniques are relatively easy to implement, their false alarm rate depends highly upon the choice of the threshold levels (discussed further in Section 3.2.1.1).

In model-based methods fault detection is performed by estimating the system's behaviour using measurements of the plant, and comparing these to a constructed model [10].

The methods used in Model-Based Fault Detection (MBFD) can be roughly divided into two categories [15]: classical analytical approaches, and artificial intelligence approaches. Analytical approaches are more appropriate for information[3] rich systems

---

2. A system's performance may be loosely defined as a measure of how well the system meets its intended design objectives [12].

which can be satisfactorily described by analytical mathematical models. In the case of poorly understood systems, where much less analytical information and poor analytical models are available, artificial intelligence approaches may be considered. In this case expert systems, neural networks, petri-nets and fuzzy logic can be applied [15].

3.2.1 Measurement Based Fault Detection - Trend and Limit checking

Some form of fault detection of technical processes can be achieved by implementing limit and trend checking techniques. The fault detection is done by comparing directly measurable signals against pre-set fixed thresholds (or trends) for upward or downward transgression. This can be easily done by using simple limit-value monitors [21]. Using this technique, various faults can be detected, but at a rather late stage, only after the monitored signal has been affected considerably[4]. The limitations of this technique are discussed extensively in Section 3.2.1.1

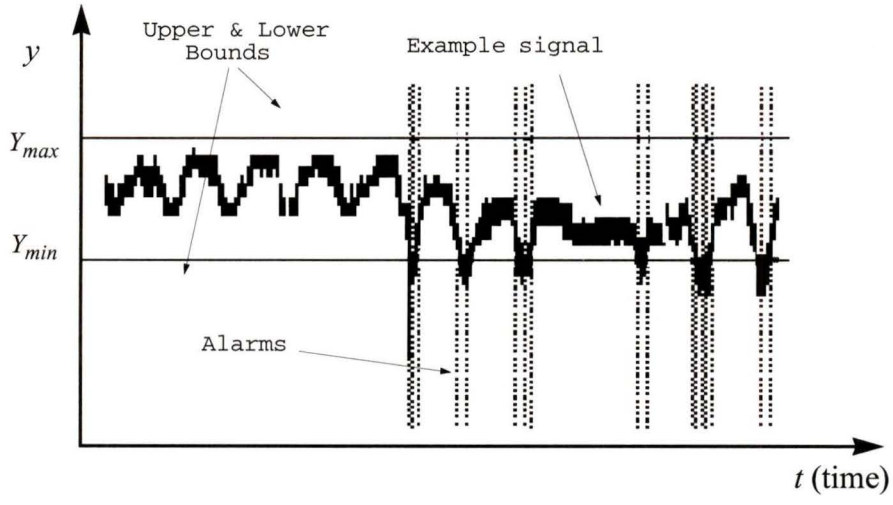

FIGURE 3-1.
Example of limit checking fault detection alarm generation. An alarm is generated (vertical lines) each time the signal transgresses either the upper or the lower threshold.

---

3. In this chapter, the utilization of the term "information" follows the work of [15]. In this context information is used to denote an analytical understanding of the plant's behaviour.

4. This holds especially for faults that result in behavioural changes without causing the monitored signal to exceed the thresholds [12].

When limit checking is used for signal *y(t)*, a fault alarm is produced as soon as the signal exceeds an adjustable maximum value Y_{max} or falls below a minimum value Y_{min}, as illustrated in Figure 3-1. The normal state for signal *y(t)* is then defined as:

$$Y_{min} < y(t) < Y_{max} \tag{3.1}$$

In the application of limit checking, the choice of the threshold values is crucial for satisfactory fault detection. The limits should be set tight enough so that there is enough time between the detection of a fault and the appearance of damage, but also wide enough so that the probability of false alarms is minimized. A quantitative method for describing such a balance which can provide design guidance for achieving a specified measure of fault detection performance is lacking [33].

Limit checking can also be applied to the trend $\dot{y}(t)$ of the signal $y(t)$. In this case the generation of alarms can take place earlier than in the case where limit checking is applied only on the signal $y(t)$, since the trend permits a certain prediction of the signal progression[5]. The normal state is then defined as [21]:

$$\dot{Y}_{min} < \dot{y}(t) < \dot{Y}_{max} \tag{3.2}$$

3.2.1.1 Limitations of Limit Checking Fault Detection

Due to its low computational cost and simplicity, limit checking fault detection is commonly employed in large engineering plants [12]. There are, however, several disadvantages and limitations associated with this technique. In limit checking, small values of the thresholds widths typically result in a high likelihood of false alarms which reduce the reliability of the fault detection system. On the other hand, large values of the thresholds widths result in an increased likelihood of leaving "real faults" undetected, resulting again in a fault detection system with decreased reliability [33].

---

5. A combination of absolute value and trend checking is also possible [21].

Obviously, the reliability of this method depends heavily on the accurate setting of the thresholds. Optimally, the threshold window should be centered around the nominal level of the signal and should be wide enough so that its "normal" variations do not exceed the limits and cause false alarms. This implies however, that the signal under consideration is wide-sense stationary and that there is some means of determining what constitutes "normal" variations [31]. For non-stationary signals this fault detection method can only be implemented, if the signal's lower order statistical moments change slowly [31]. Under this condition, manual adjustment of the limits would be possible, but clearly such a solution is less than ideal, particularly in terms of the goal of developing automated environments.

Even in cases where the bounds have been set in accordance with the principles outlined above, signal dependencies which cause significant perturbations in the monitored signal may produce false alarms. These types of threshold transgressions do not correspond to real faults and, therefore should not be reported. In addition, there is generally a lack of consistent flag generation throughout the duration of a fault, since, with the exception of extreme faults, the technique tends to produce sporadic error flags when the signal happens to exceed the thresholds [31]. This sporadic fault-flag generation can be clearly seen in Figure 3-1.

Finally, this technique is not able to detect faults that result in behavioural changes of the signal but do not cause the thresholds to be exceeded. As long as the signal remains between the bounds, any type of behaviour is deemed acceptable. This factor presents serious limitations to the technique's fault detection capabilities [24].

The difficulties and disadvantages of using limit checking fault detection can be mitigated if the affected signal $\boldsymbol{y}(t)$ can be in some way predicted. In order to do this, a mathematical model capable of capturing the behaviour of the system must be created, which will produce an estimate $\hat{\boldsymbol{y}}(t)$ of the signal. An absolute value check can then be employed on the error between the actual and predicted signal to detect faults:

$$|y(t) - \hat{y}(t)| < \varepsilon \tag{3.3}$$

where ε is an error threshold.

3.2.2 Model-Based Fault Detection (MBFD)

The most capable among the underlying strategies for fault detection is the model-based approach [15]. Model-based fault detection depends on the construction of an accurate mathematical model of the system. In order to construct such a mathematical model, the inherent mathematical redundancy contained in the static and dynamic relationships among the system's inputs and outputs must be exploited [14]. Clearly, if the dynamic processes arising in a physical plant could be understood exactly and every process variable could be precisely measured, then it would be possible to detect, locate and identify most faults in the plant [15]. This could be done by comparing the data collected from the actual plant with estimations obtained by the mathematical models. This is the fundamental concept of the model-based approach to fault detection. In the literature these techniques are referred to as "analytical redundancy" techniques since they utilize analytical models of the plant (and not physical redundancy) to track the changes in the plant's behaviour and to perform the fault detection task [31].

3.2.2.1 Residual Generation

When an accurate analytical model for an engineering plant is available, the detection process is a relatively straight forward matter of analyzing the residual generated from the "residual generator". *Residuals* are defined as the variation between measured plant variables and their estimates obtained by the mathematical model [14]. As such, MBFD can basically be described as a two step process: generation of residuals which should be sensitive to fault conditions, and the evaluation of the residuals with the use of a decision maker.

The task of MBFD is to detect process faults through the use of the measurable input and output variables $\boldsymbol{u}(t)$ and $\boldsymbol{y}(t)$. Mathematical models can be used to this end [21]:

$$\boldsymbol{y} = f(\boldsymbol{u}, \boldsymbol{N}, \theta, \boldsymbol{X}) \tag{3.4}$$

where $\boldsymbol{N}$ generally represents non-measurable process and measurement noise

θ non-measurable process parameters and

$\boldsymbol{X}$ partially measurable and non-measurable internal state variables.

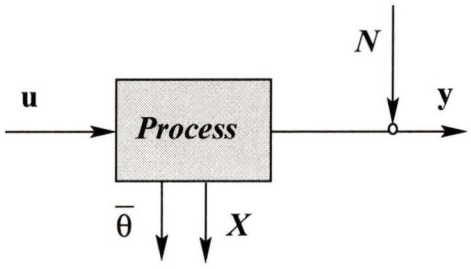

FIGURE 3-2.
Representation of a process with measurable input and output variables ($\boldsymbol{u}$ and $\boldsymbol{y}$), and non-measurable disturbance variables $\boldsymbol{N}$, process parameter θ and state variables $\boldsymbol{X}$.

A significant deviation of one of the quantities $\boldsymbol{y}$, θ, $\boldsymbol{X}$, from its expected value is characterized as a fault. Generally, the process parameters are constants or slowly time varying coefficients and the state variables are time-dependent. The non-measurable parameters of the model can be obtained by applying recursive parameter estimation methods [21]. A representation of a process is shown in Figure 3-2 [21].

Basically, the analytical redundancy approach utilizes a residual generator which validates the nominal relationships of the system using the actual input $\boldsymbol{u}$ and measured outputs. Whenever faults occur, the redundancy relations are no longer satisfied and a residual of $\boldsymbol{r} \neq \mathbf{0}$ is generated. The residual is then used to form appropriate decision functions. In order to achieve an accurate fault detection with low false alarm rate, the model should be frequently updated.

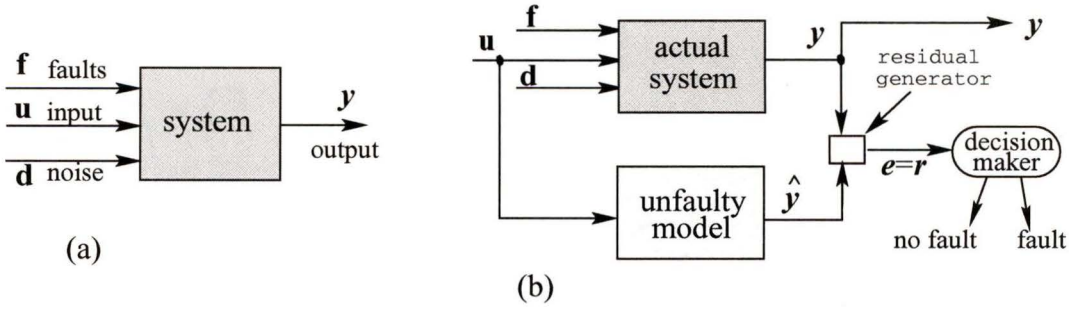

FIGURE 3-3.
System representation using classical model-based approach (a). General basic block diagram of model-based fault detection schemes (b).

Within classical signal processing and control theory, the basic system model that is generally under consideration is shown in Figure 3-3(a). In this model, the system is operated by a known input $\boldsymbol{u}(t)$ and its operation is observed through the available sensor output $\boldsymbol{y}(t)$. All the signals within the system are assumed to be affected by system noise ($\mathbf{d}$) and faults ($\mathbf{f}$) are assumed to occur throughout the system. Figure 3-3(b) illustrates a general block diagram of MBFD.

There is a great variety of proposed methods of generating residuals. These can be brought down to a few basic concepts, such as [14]:

1. the parity space approach
2. the dedicated observer approach
3. the fault detection filter approach, and
4. the parameter identification approach.

The first three approaches are closely related since they typically utilize state estimations: either dedicated observers or Kalman filters [14][31]. A detailed description of these techniques can be found in [31] and specific applications in [10][15][21].

3.2.2.2 Limitations of Analytical Redundancy Techniques

Unfortunately, the analytical redundancy approach invariently does not apply to operational plants. Neither can perfect models be created, nor are all necessary measurements for creating the models available. These limitations particularly hold for large scale plants that are typically complex, highly uncertain, time-varying and non-linear. For these plants the reliable measurement of the complete set of process variables is usually complicated, expensive or simply impossible.

Most of the above residual generation methods utilize linear models of the plants under consideration and, hence, are subject to modeling errors[6]. Within some of these approaches, non-linear models may be utilized which can capture some of the non-lineari-

---

6. Linear models are easyer to construct. Modeling errors are then introduced if the plant that is being modeled exhibits non-linearities.

ties exhibited by the system. This refinement may improve the model's overall accuracy, but these non-linear models have to be tailored to the specific plant under consideration [31]. Finally, most of these techniques require that the number of fault modalities be known *a priori,* information that is generally not available, especially in large scale engineering plants.

Nevertheless, even under such restricted conditions, MBFD remains basically practicable, and depending upon the special type of plant and the given restrictions, with proper modifications it can offer adequate fault detection. In cases where analytical models of the system are not available there is some potential for fault detection using artificial intelligence techniques such as Neural Networks.

3.2.3 Neural Networks and Fault Detection

In order to use the analytical modeling approach the understanding of the dynamics of the system under consideration must be accurate. However, it is possible to use ANNs to implement the modeling part of the fault detection system. This way no analytical modeling of the system is necessary. Neural networks can learn a desired mapping and "internal model" of the system. The training of the network can be done without having to know the exact mathematical relationship between the outputs and the inputs, but rather by presenting to the network input-output examples. Once appropriately trained, the adjusted weights of the networks will contain the relationships and non-linearities of the desired mapping. As a result, the complicated analytical modeling can be avoided.

From the system engineering point of view, it is simpler to visualize a neural network as a "black box" that is capable of modeling the output of a system. This property of neural networks can then be used in a model-based fault detection system.

Let a system have inputs $\boldsymbol{u}$ and corresponding outputs $\boldsymbol{y}$. The system can be represented as the mapping $\mathcal{M}$ from the input space $\mathcal{U}$ to the output space $\mathcal{Y}$, expressed in Eq. 3.5:

$$\mathcal{U}\text{--->}\mathcal{Y} \tag{3.5}$$

An artificial neural network can be used to learn this mapping $\mathcal{M}$. In accordance with Section 2.3.4, the output of the neural network is denoted by $\hat{\boldsymbol{y}}(\boldsymbol{u}, \boldsymbol{w})$, where $\boldsymbol{u}$ is the input of the actual system and input for the network, and $\boldsymbol{w}$ are the weights of the network. Generally, the mapping of the system $\mathcal{M}$ is not known. However, the set of input-output measurements $\{ (\boldsymbol{u}_1, \boldsymbol{y}_1), (\boldsymbol{u}_2, \boldsymbol{y}_2), \ldots (\boldsymbol{u}_p, \boldsymbol{y}_p) \}$ of $\boldsymbol{u}$ and $\boldsymbol{y}=\mathcal{M}(\boldsymbol{u})$ can usually be obtained accurately. In terms of neural network terminology, $\boldsymbol{u}_p$ is the p^{th} input pattern, $\boldsymbol{y}_p = \boldsymbol{y}(\boldsymbol{u}_p)$ is the corresponding output pattern, and $(\boldsymbol{u}_p, \boldsymbol{y}_p)$ is the p^{th} training pattern.

The training of the network can be thought of as training the network to represent the input-output mapping $\mathcal{M}$ as close as possible by adjusting the weights $\boldsymbol{w}$ so that the difference between the network output $\hat{\boldsymbol{y}}(\boldsymbol{u}, \boldsymbol{w})$ and the actual output $\boldsymbol{y}$ of the system with given input $\boldsymbol{u}$ will be lower than a preset error tolerance e_{tol}. The training process can be mathematically represented as:

$$\varepsilon_{min} = min \ \arg_w [\| \hat{y}(\boldsymbol{u}_p, \boldsymbol{w}) - \boldsymbol{y}(u_p) \|] \ p = 1, 2 \ldots P \tag{3.6}$$

such that $\varepsilon_{min} \leq e_{tol}$

where $\|.\|$ is some appropriate norm

A block-diagram representation of the training is shown in Figure 3-4.

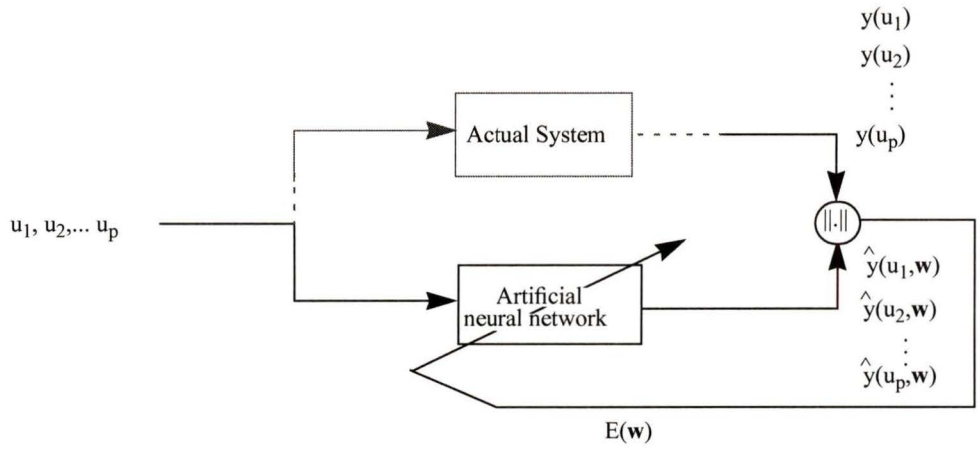

FIGURE 3-4.
Supervised training of an artificial neural network for system modeling.

After the neural network has learned the actual system input-output mapping sufficiently well (that is when the error (Eq. 3.6) is lower than the tolerance error e_{tol}), the network can be used to produce an estimate of the system's outputs given a set of inputs. Essentially, a model is constructed for each system, subsystem or component, which is capable of modeling its output. This model can be further used to detect any deviations of the measurements obtained by the actual system's output from these which the model indicates [24].

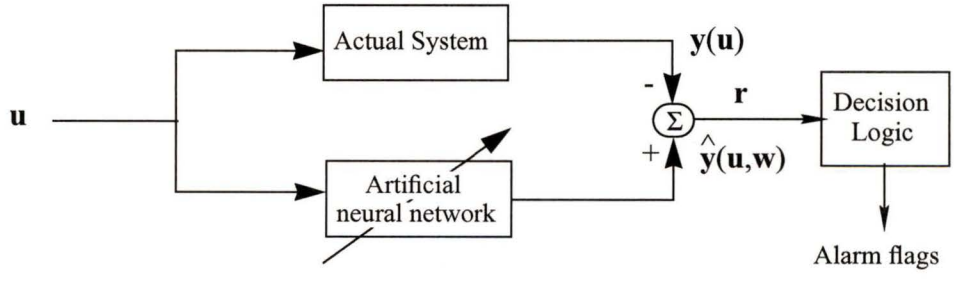

FIGURE 3-5.
Neural network fault detection block diagram

Any such deviations that are larger than a predetermined deviation tolerance are considered the results of faults. Hence, fault detection can be performed in a manner similar to analytical redundancy techniques, by generating a residual $\boldsymbol{r}$, a product of the comparison of the system's output $\boldsymbol{y}(\boldsymbol{u})$ and the neural network's estimate $\hat{\boldsymbol{y}}(\boldsymbol{u}, \boldsymbol{w})$. A general block diagram of the neural network fault detection is illustrated in Figure 3-5.

Unlike some of the classical modeling techniques where some information about the underlying plant operation can be extracted from the constructed plant model, no information about the plant is obtained when using neural networks. After the neural network has been trained, the plant behaviour, specified by the rules that govern the input-output mapping, are contained within its interconnection weights, its activation function and its architecture and cannot generally be extracted. This fact though, does not impose any limitations on the use of ANNs in a fault detection[7] system, since the sole information needed for this type of fault detection is the neural network's output, i.e. its estimate of the plant's output.

The most appropriate architecture of neural network to be used depends on the process to be modeled. Feedforward networks with sufficient numbers of hidden layers and neurons, can approximate arbitrarily well any non-linear function [9], and therefore are suitable for modeling non-linear static processes. On the other hand, recurrent neural networks, due to their feedback connections, are appropriate for modeling dynamic processes [13].

3.3 Summary

In modern automated engineering plants where quality improvement, cost reduction, safety, reliability and longevity of the plant are considered critical, fault detection is becoming a basic requirement. The task of fault detection is to determine the existence of faults in the plant's components at as early a stage as possible before the performance of the system is degraded. Fault detection techniques can be broadly classified as *measure-*

---

7. The neural network approach can only detect faults which cause changes in the behaviour of the plant's output. However, it cannot detect changes in the state variables or process parameters which generally offer information for the diagnosis of the fault.

ment-based or *model-based.* Further, model-based techniques can be roughly divided into two categories: analytical redundancy approaches, using analytical mathematical models, and artificial intelligence approaches, using various techniques such as neural networks.

Measurement-based techniques suffer from a high probability of false alarms and are not suitable for fault detection of non-stationary signals. In addition they are not capable of detecting faults that result in a change in the behaviour of the signal, but do not cause the signal to exceed the thresholds. Analytical redundancy techniques address these problems and can be used for fault detection in information rich systems where mathematical models of the system can be created. In the case of poorly understood systems where accurate mathematical models are not available, neural networks can be used. By training a neural network to learn the input-output relationship of the system in under consideration, the neural network creates a "black-box" model of the system which can then be used to produce an estimate of the systems output. To this extent fault detection via neural network techniques is performed in a way similar to that of analytical redundancy techniques.

Chapter 4:

Cable Television Amplifier Networks

4.0 Introduction

This chapter describes the basic structure of a cable amplifier network. Throughout this chapter some background information on the trunk amplifiers used in Rogers Newmarket cable distribution network is presented. For the purposes of performance monitoring and fault detection, several status fields are monitored by a status monitoring system. The temperature and reverse pilot fields are discussed in more detail in Sections 4.4.1 and 4.4.2, and the existing limit checking fault detection scheme is described in Section 4.5. Finally, two preliminary and incomplete fault detection methods are briefly described in the last sections of this Chapter. These methods were designed on a case-specific basis, and are not generally applicable to all reverse pilot signals. However, they are included because they present an introduction to concepts and techniques that are used in the complete fault detection systems presented in the next Chapter.

4.1 Cable Television Amplifier Networks

Cable television distribution networks are used to distribute cable signals from a centrally located injection site to subscribers' homes and businesses. This injection point is referred to as the "head-end". These networks, commonly termed cable plants, are responsible for providing consistent and high quality signals to all the cable subscribers. The basic structure of the network consists of several hundred to a few thousand cable television "trunk" amplifiers. These amplifiers are interconnected by coaxial cable in a tree type network as shown in Figure 4-1. The actual number of amplifiers in a network depends on the size of the area the network covers as well as the number of subscribers to be serviced. Smaller spans of distribution amplifiers exist at the leaf nodes of this tree; these propagate the cable signals from the main cable trunk network down to the subscribers. As a trunk amplifier may supply cable signals to up to twelve distribution amplifiers, and each of these typically provide cable service to two to three hundred subscribers, the failure of a single trunk amplifier may affect many hundreds of cable subscribers.

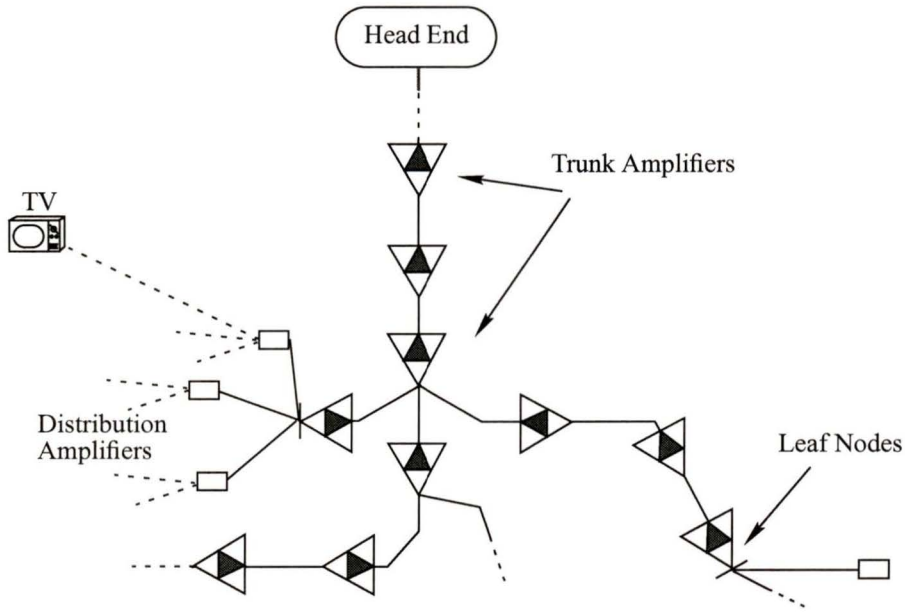

FIGURE 4-1.
Section of an amplifier network.

Typically, cable amplifier plants are two-way asymmetrical communication networks. The downstream path, from the head-end to the subscribers' homes, is a high bandwidth path which is used mainly for delivering cable television services. The upstream path, also referred to as the "reverse path", is a low bandwidth path that has been used traditionally to transmit the status data from the trunk amplifiers to the head-end. More recently, the reverse path has been used to provide a data path from the subscribers to the head-end for use in interactive services such as internet and video-game channels. Consequently, the requirement to provide a high quality reverse path has become increasingly important.

4.2 Cable Trunk Amplifiers

Mainly, the amplifiers used in Roger's cable television plant in Newmarket, Ontario are manufactured by C-COR Electronics Inc. and are specifically designed for use in broadband networks such as cable television networks. Their main purpose is to restore the strength of the signal, as it is attenuated by the spans of coax cable between the amplifiers [30]. These trunk amplifiers provide a two-way communication capability and can amplify signals over a range of a several hundred MHz. The amplifier's forward amplifi-

cation module amplifies the signals propagating downstream, called the forward pilot signal, while the reverse amplification module amplifies the signals propagating upstream, called the reverse pilot signal. Two types of amplifiers are required to maintain the desired strength for the forward signal: high pilot amplifiers with their forward module tuned to amplify the high frequencies of the signal, and low pilot amplifiers which amplify the low frequencies. Because of the relatively narrow bandwidth of the reverse path (5-33 MHz), unlike the case of the forward pilot signal, only one type of reverse amplification module is required to amplify the reverse signal.

4.3 Amplifier Monitoring

In order to perform any type of system monitoring, some information about the plant and its components must be obtained. This information is obtained by sensors which are designed to measure specific parameters of interest and are placed within the components. For practical and economic reasons not all components within the amplifiers are monitored. Typically, only those which are critical to the plant's operation and offer information about the amplifiers' behaviour are chosen to be monitored.

The data from the components' sensors are collected and transmitted to a Central Status Monitoring System (CSMS). This work is done by the Status Monitoring Transponder (SMT) which is present within each monitored amplifier. In addition, each SMT module is responsible for the sampling and quantization of the analog sensor signals. A unique address is assigned by the CSMS to each SMT module. The CSMS located at the head-end exists as a software suite on a personal computer (PC) and its task is to collect the data from all the status transponders. In order to do so, the status monitoring software (called Demon), uses a polling scheme in which all the trunk amplifiers are consecutively polled in the order of their SMT addresses. In the Newmarket plant, each amplifier SMT is polled once every 55 seconds. This results in approximately 1600 samples per day for each trunk amplifier.

The status data is sent to CSMS via the reverse path of the network, where it is time stamped in order to facilitate further processing of the data records. It is within the CSMS where the behaviour monitoring is implemented and analysis of the data for fault conditions is done. The whole data collection system when viewed in terms of a single status monitored variable, can be represented by the block diagram in Figure 4-2.

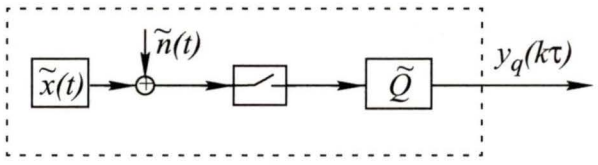

FIGURE 4-2.
Functional equivalent of a data collection system.

As per the notation and theory in [31], the output signal $y_q(k\tau)$ can be described by:

$$y_q(k\tau) = \tilde{Q}([\tilde{x}(t) + \tilde{n}(t)])|_{t=k\tau} \tag{4.1}$$

where τ is a constant sampling period, $k = 0, 1, 2, \ldots$

$\tilde{x}(t)$ represents the sensor output signal

$\tilde{n}(t)$ is the added noise (before sampling)

$\tilde{Q}(.)$ is the quantization function used.

4.4 Monitored Fields

The operational status of a cable television plant is tracked through the use of the Central Status Monitoring System. In total six status parameters are monitored for each amplifier. Specifically these parameters are:

1. The forward pilot level (a measure of the forward signal strength).
2. The reverse pilot level (a measure of the reverse signal strength).
3. The temperature (a measure of the temperature within the amplifier's housing).
4. The raw DC voltage supplied to the amplifier.
5. The current draw (a measure of the amount of current drawn by the amplifier).
6. The regulated voltage produced by the amplifier's regulated power supply.

The following sections 4.4.1 and 4.4.2, examine two of the monitored fields which are of interest in the work presented in this thesis: the temperature *(T)* and the reverse pilot *(R)*.

4.4.1 The Temperature

The temperature status signal *(T)* is a measure of the temperature within the amplifier's housing. A sensor located within the amplifier measures the temperature, while the SMT module quantizes and sends the information back to the CSMS each time it is polled (Section 4.3) [5].The quantization step for the temperature is 0.5 ^{o}C.

The measured temperature depends on the amount of heat produced by the electronic equipment of the amplifier, but also on the temperature of the amplifier's surrounding environment. As a consequence, the temperature field is affected by both daily and seasonal temperature variations. Typically, the amplifiers' temperatures range from 5 ^{o}C - 110 ^{o}C.

Each amplifier's temperature variation and resultant nominal setting are unique and depend highly on the cable amplifier's location. For example, amplifiers that are located underground or inside temperature regulated environments will not produce significant variations in their temperature, while amplifiers that are exposed to weather conditions

and sunlight will experience more extreme fluctuations following the temperature fluctuations of their external environment. Typical daily and monthly temperature measurements are shown in Figure 4-3.

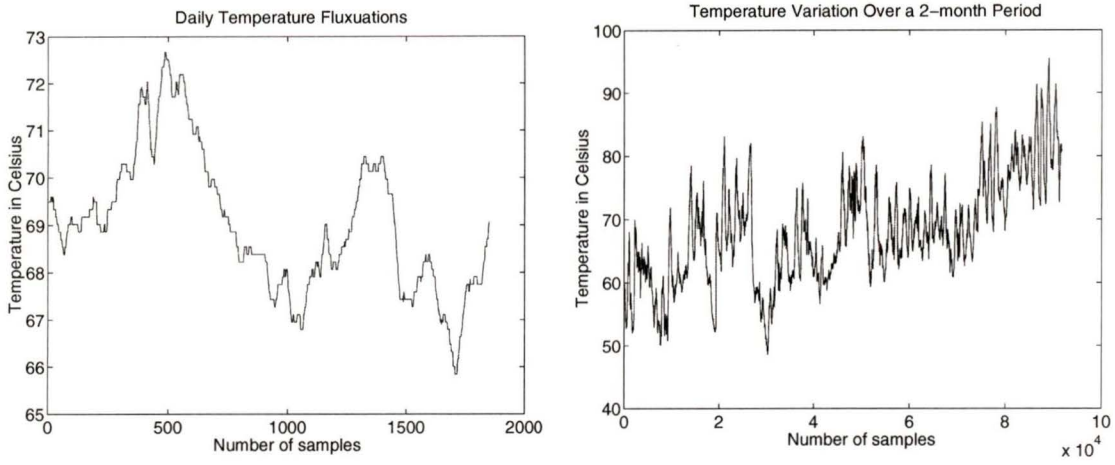

FIGURE 4-3.
Daily Temperature variations (a); Monthly Temperature variations (b).

It has been observed that other monitored signals such as the forward pilot signal *(P)*, the reverse pilot signal *(R)* and the DC current *(I)* follow in some way and extent the temperature fluctuations [24] [31].

4.4.2 The Reverse Pilot

The reverse pilot amplifier module is used to amplify the signals propagating upstream within the cable plant. The reverse pilot status measurement takes place at the head-end, and its level depends both on the amplifications performed by the cascade of reverse amplifiers and the losses that incur along the reverse path. Each polled amplifier sends an upstream carrier signal produced by the status monitoring transponder. The reverse amplifier module of each amplifier is adjusted [1] so that the losses that the signal incurs in the intervening cable span toward the next amplifier are compensated for, and the signal's level at the head-end is at a certain predetermined nominal value.

---

1. The amplifiers' modules adjustments are done during installation and also during subsequent maintenance visits to the amplifiers [30].

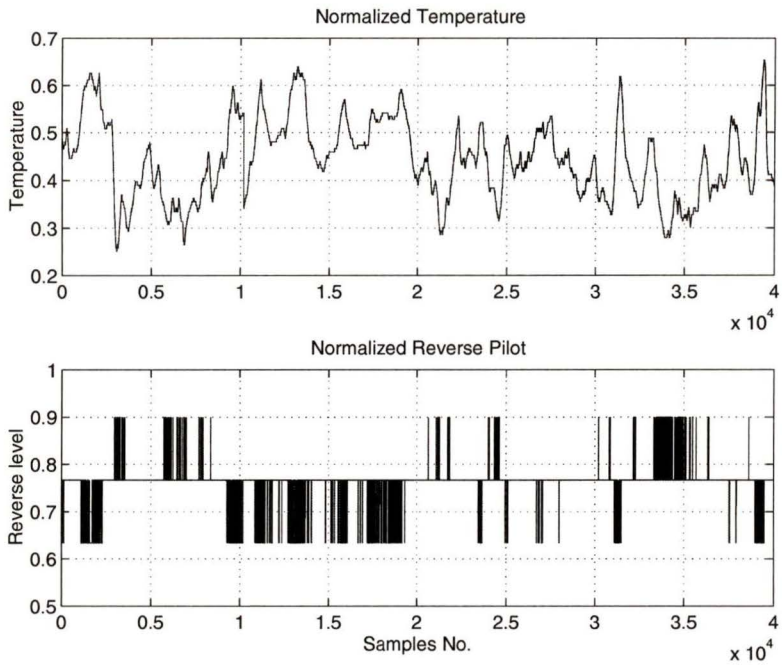

FIGURE 4-4.
The reverse pilot and the Temperature are negatively correlated

The reverse pilot level is quantized with a quantization step of 1 dBmV, resulting in a low number of quantization steps over the dynamic range of the signal. This quantization step is relatively large compared to those used for the quantization of the forward pilot or the temperature. This coarse quantization adds a significant amount of quantization noise to the signal, which complicates the problem of detecting faults in the reverse path of the trunk amplifier plant.

As noted in Section 4.3, the measurement of most monitored signals, such as that of the temperature and forward pilot, takes place in the amplifier and their measured values are sent via the reverse path to the head-end. In contrast, the reverse pilot level is measured (in dBmV) at the head-end. Assuming that the polled amplifier always emits an upstream signal at a given level, then the measurement at the head end will remain constant if the losses along the path do not change. The reality however, is that the reverse level as measured at the head-end changes. Part of this change may be attributed to noise,

but it has also been observed that these changes are negatively correlated[2] with the temperature: i.e. when the temperature increases, the reverse pilot level decreases and *vice versa*. This can be seen easily in Figure 4-4 and Figure 4-5.

In particular, we may assume that the reverse pilot-temperature (*R-T*) dependency is an unknown linear dependency, which can be expressed as:

$$R(t) = f(T(t)) \tag{4.2}$$

where $R(t)$ is the continuous time reverse pilot level,

$T(t)$ is the temperature of the enclosure of the amplifier,

and $f(.)$ is an unknown linear function[3].

This relationship is further obscured by the addition of noise on the reverse signal and the effects of quantization, which transform the dependency function into a non linear staircase. In terms of the reverse status data, as per the notation of Section 4.3, we can rewrite Eq. 4.2 as:

$$R_q(k\tau) = f_{RT}(T_q(k\tau)) \tag{4.3}$$

where $R_q(k\tau)$ and $T_q(k\tau)$ are the measured outputs of Eq. 4.1,

τ is a constant sampling period, $k = 0, 1, 2, \ldots$,

f_{RT} is the unknown non-linear *R-T* dependency function,

and the subscript q in $R_q(k\tau)$ and $T_q(k\tau)$ denotes the fact that these signals are quantized.

In the following sections, for simplicity, the measured temperature T_q and reverse level R_q will be denoted simply as T and R.

---

2. The influence of the temperature on the impedances of the amplification modules and the transmission medium (cable) along the reverse path are the main cause of this dependency.

3. We assume $f(.)$ to be a static linear function.

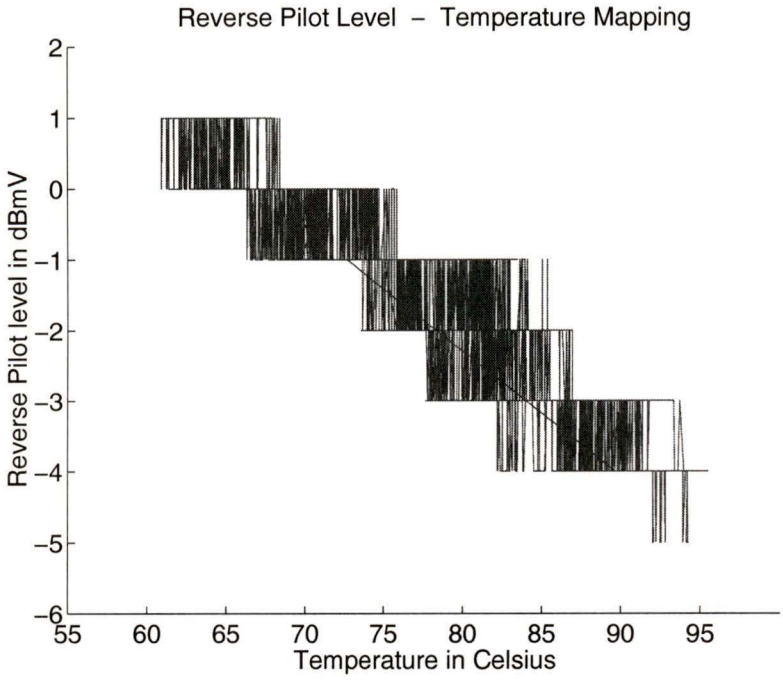

FIGURE 4-5.
Plot of a typical reverse-temperature relationship of a well behaving amplifier.

By plotting the reverse pilot level versus the temperature of a typical amplifier for a time period of half a month (Figure 4-5), it can be clearly seen that the reverse pilot-temperature (*R-T)* relationship, can be more accurately described as a staircase relationship with overlapping transition areas.

4.5 Reverse Pilot Fault Detection in Rogers Cable Amplifier Network

In Rogers cable network, a limit checking system is implemented in order to perform fault detection for the reverse pilot module. As described in Section 3.2.1, the general process in this fault detection technique involves placing fixed threshold bounds above and below the signal's "normal" range of variation. As long as the reverse pilot signal remains within these bounds, it is assumed to be fault-free. On the other hand, whenever the signal exceeds either the upper or lower bound, a fault alarm is produced, and the associated module is assumed to have experienced a fault (Figure 4-6).

The major drawbacks of limit checking, as outlined in Section 3.2.1.1, are twofold. First, a large number of false alarms are produced which as discussed in Chapter 3, are not easy to eliminate, and secondly, this technique is unable to detect faults that result in behavioural changes[4] in the amplifier without causing the signal to exceed the thresholds.

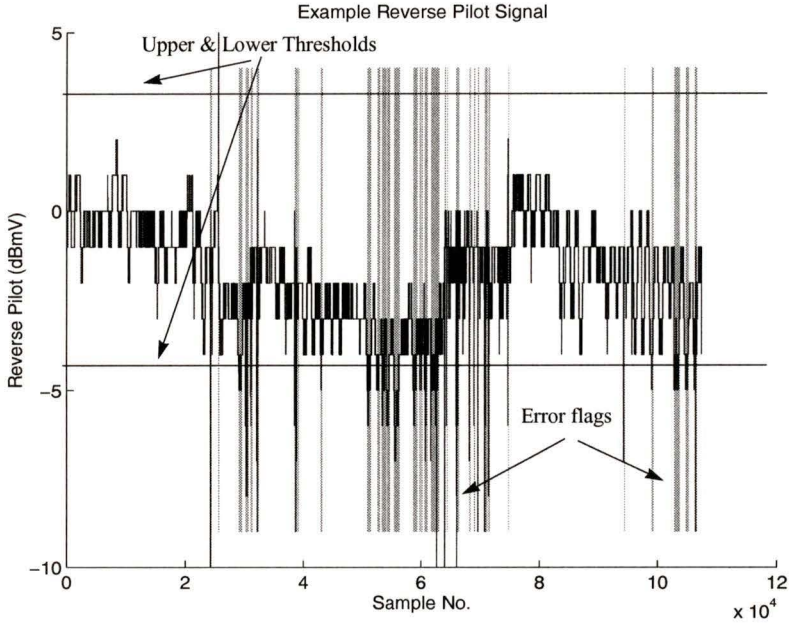

FIGURE 4-6.
Typical error flag generation by use of threshold fault detection. The vertical gray lines depict the occurrence of alarms. The two horizontal lines depict the position of the upper and lower bounds of the allowable variation of the signal.

When limit checking is used, signals which exhibit some kind of temperature dependency, such as the reverse pilot, will produce a high number of false alarms. Because of this dependency, seasonal and daily temperature variations will cause significant perturbations in the reverse pilot signal. As a result, the status signal may cross the threshold limits causing false alarms, even though these perturbations are considered as normal since they are not associated with real faults. The only way to deal with this problem when using fixed threshold bounds, is to use a threshold window adequately wide to allow the signal's

4. As change in the behaviour of the reverse pilot we define the change in the relationship (dependency) of the signal to the temperature - change in the dependency function $f_{RT}(.)$ (Eq. 4.3).

"fast" perturbations without causing it to exceed the thresholds. However, this procedure will increase the probability of real faults going undetected, and unfortunately offers a solution that deals only with the effects of daily (fast) temperature fluctuation of relatively small variation. The effects of "slow" temperature variations such as seasonal variations can be alleviated to some extend by periodically resetting the nominal level of the threshold window.

A general approach for fault detection of the reverse pilot that deals with the temperature dependency problem more efficiently than limit checking (as implemented in Rogers Cable Systems), involves modeling the temperature dependency of the reverse pilot signal.

4.6 Preliminary Modeling Approaches

The main part of the research done for this thesis has been directed towards the development of model based methods for fault detection for the reverse pilot of cable amplifiers. The underlying goal is for these methods to be generally applicable to all amplifiers without any exceptions and under minimum *a priori* assumptions. Furthermore, these methods should perform better[5] than the existing limit checking system. Throughout the course of this research several different approaches were tested and explored, with some of them returning promising results, and others leading to a dead end.

In the following Sections 4.6.1 and 4.6.2, two preliminary approaches are presented. These methods were developed on a case-specific basis, are not complete, and are generally not applicable to all reverse pilot signals. However, they are briefly described, because they constitute the starting point for the development of the complete and successful fault detection methods presented in Chapter 5.

---

5. In the context of this thesis, a fault detection system is "better" than another if: It discovers changes in behaviour, produces fewer false alarms and utilizes tighter bounds.

4.6.1 Basic Modeling Approach

One way to perform fault detection for signals such as the reverse pilot, with an unknown temperature dependency, is to use information available from the behavioural map of the signals' relationship. For a given set of reverse pilot R and temperature T measurements this map can be denoted as $\mathcal{M}(R, T) = 0$. The general form of this map is that shown in Figure 4-5. The modeling process can then be defined as the process of generating two bounding functions $f_l(T)$ and $f_u(T)$ such that for any temperature T and for all values of the reverse pilot R which satisfy the mapping $\mathcal{M}(R, T) = 0$ then $f_l(T) < R < f_u(T)$. These bounding functions should produce tight bounds on $\mathcal{M}(R, T) = 0$ and be easy to compute from a relatively limited number of observations.

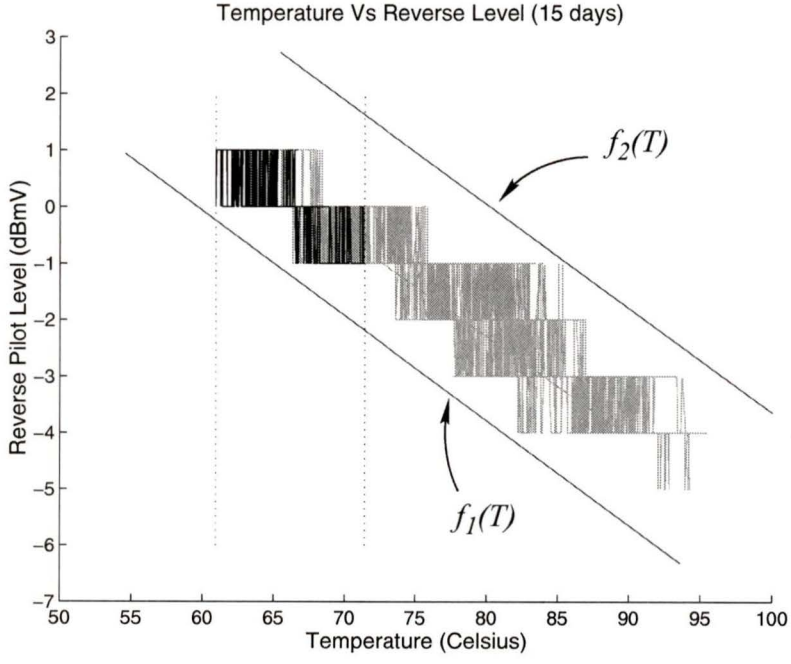

FIGURE 4-7.
Lower and upper thresholding functions $f_1(T)$ and $f_2(T)$. The bounds are set manually.

Using the above approach, the fault detection system consists of an upper and a lower bound, which form a diagonal band within which the *R-T* map is considered to correspond to normal behaviour. In other words, as long as the *R-T* mapping of the amplifier remains in the area between these bounds, the amplifier's reverse pilot is deemed to be normal. Outside of this area the reverse path is assumed to be operating under a fault condition.

An example of the bounding of the *R-T* mapping is shown in Figure 4-7, where the distribution of Figure 4-5 is reproduced and bounded by two linear functions, with $f_1(T)$ as a lower bound and $f_2(T)$ as an upper bound. The results of employing these bounding functions on the time domain reverse signal associated with Figure 4-7 are shown in Figure 4-8.

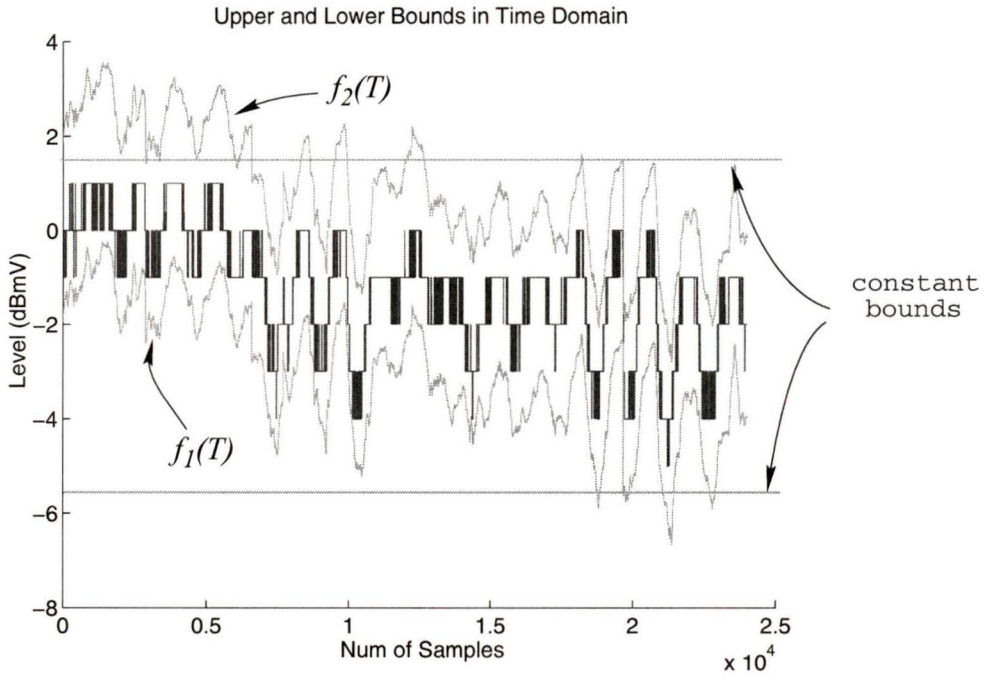

FIGURE 4-8.
Application of the lower and upper linear (temperature dependent) bounding functions, $f_1(T)$ and $f_2(T)$ to the time domain reverse signal associated with Figure 4-7. The tightest constant bounds that can be applied to this signal without causing any alarms are also shown. The plotted reverse pilot segment corresponds to 15 days of data.

The temperature dependent bounds shown in Figure 4-8 are set to an average distance of 3.9 dBmV apart. The bounding window generated by these two functions is quite narrow, even compared to the tightest temperature constant bounds that can be used for the example signal without causing any alarms. These constant bounds can be seen in Figure 4-8, and are 7 dBmV apart.

The two bounding functions f_l and f_h of this example, were selected to be adequately wide apart to include all the data points of the distribution[6]. In other words, the selection of the bounds was done without taking into account the statistical nature of the dependency map. As such, information on the likelihood of occurrence of the "extreme points" in the *R-T* map was not considered (Figure 4-9).

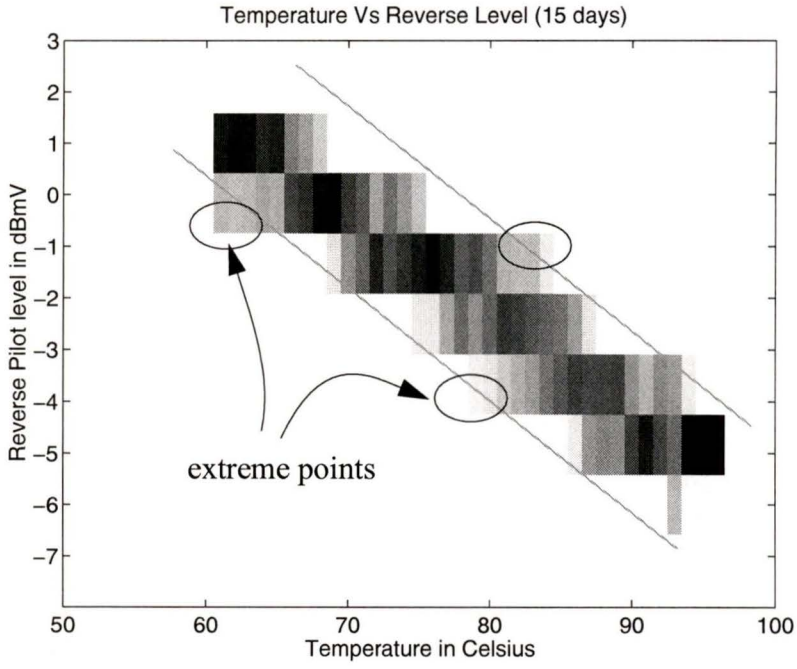

FIGURE 4-9.
The *R-T* distribution. Gray scale represents the density of occurrence of the reverse levels for each temperature. The bounds are set tighter than in Figure 4-7.

---

6. The bounds in this example were set manually. In general, the bounding functions are given by $f_l(T) = a_l \cdot T + b_l$ and $f_h(T) = a_h \cdot T + b_h$, and the coefficients a_l, b_l, a_h, b_h have to be found.

By utilizing statistical techniques, it is possible to obtain such information about the mapping and by using this information, the bounds can be set even tighter.[7] This is achieved by not allowing statistically rare events to influence the positioning of the bounds. In Figure 4-9, the *R-T* dependency map of Figure 4-7 is reproduced, using this time, statistical information, which reveals the percentage of occurrence of each reverse level with respect to the temperature. More specifically, the temperatures are rounded to an accuracy of 0.5 ^{o}C and the density of occurrence of each level for each temperature is represented in gray scale, with the darker scale corresponding to levels that occur more often (for the given behaviour).

Ideally, we would like to be able to estimate the mapping and set the upper and lower bounds tight enough by using only a limited number of observations. The restriction to this is that a large enough range of temperatures and reverse levels need to be contained in that small set in order for the slope of the distribution to be extrapolated. In Figure 4-7 the *R-T* distribution for one day's data is shown with darker color, and as it can be seen, it is not easy to estimate the bounds so that they fall close enough to $f_1(T)$ and $f_2(T)$, in order for them to be used for fault detection.

Unfortunately, the approach of bounding the reverse - temperature distribution, apparently applicable to the example presented, has in general serious limitations. First, the basic requirement of being able to construct a model from a limited number of observations, generally can not be met. Secondly, it is obvious that non-linear dependencies can not always be bounded effectively by linear functions. Moreover, detailed analysis of the signals of the cable amplifier plant shows that the monotonic staircase *R-T* relationship which was presented in the example, does not occur consistently for all the amplifiers, which makes the bounding process even more difficult. In any case, this approach seems not to be very promising for use as a fault detection method, for the reverse pilot at least.

---

7. Developing a methodology on how to set the bounds is outside the scope of this thesis.

4.6.2 Using Neural Networks - A First Approach

In this section, a first approach for modeling the reverse pilot using neural networks is presented. The first stage, involves the processing of the reverse and temperature data. By implementing statistical analysis on the data, we can find the single reverse pilot level that occurs the most for each temperature, obtaining a one-to-one temperature-reverse pilot map. This map will be referred to as "*R-T* dependency function".

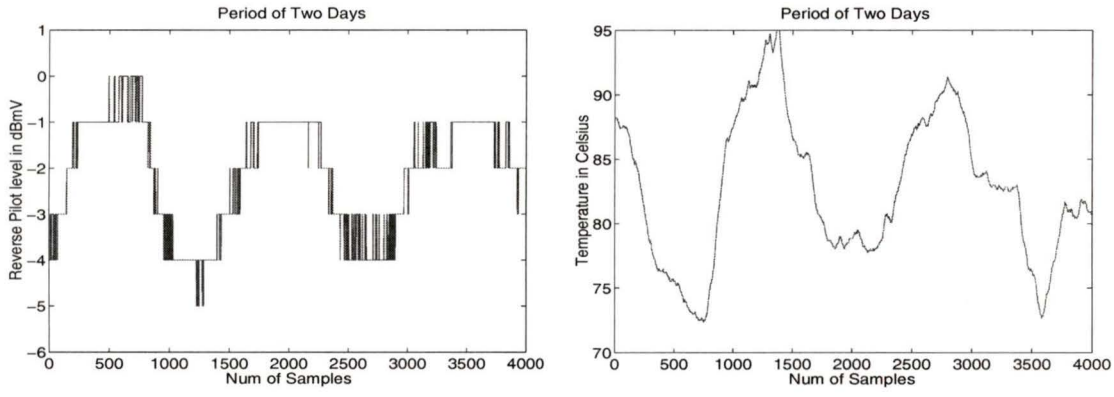

FIGURE 4-10.
The training set used to generate the dependency function of Figure 4-11(a).

The second stage involves the modeling of the reverse pilot, based on the dependency function generated in the first stage. Specifically, a neural network is used to approximate the dependency function, and then simulate the reverse signal. For the purpose of fault detection, we can then bound the error between the reverse signal measurements and the output of the neural network, producing an alarm each time the error threshold is exceeded.

Applying the statistical analysis on the small data set seen in Figure 4-11 (which correspond to the last segment of 4,000 samples shown in Figure 4-8), it was possible to produce the dependency function illustrated in Figure 4-11a (dark trace). A feedforward neural network with six neurons in each of its two hidden layers[8] was trained to approximate this function. The training was done with temperature data as inputs and reverse pilot data as the target, with the *T-R* pairs taken from the map of Figure 4-11a (dark trace). The results of the neural network's function approximation are plotted in Figure 4-11b.

8. This architecture was sufficient for the specific application.

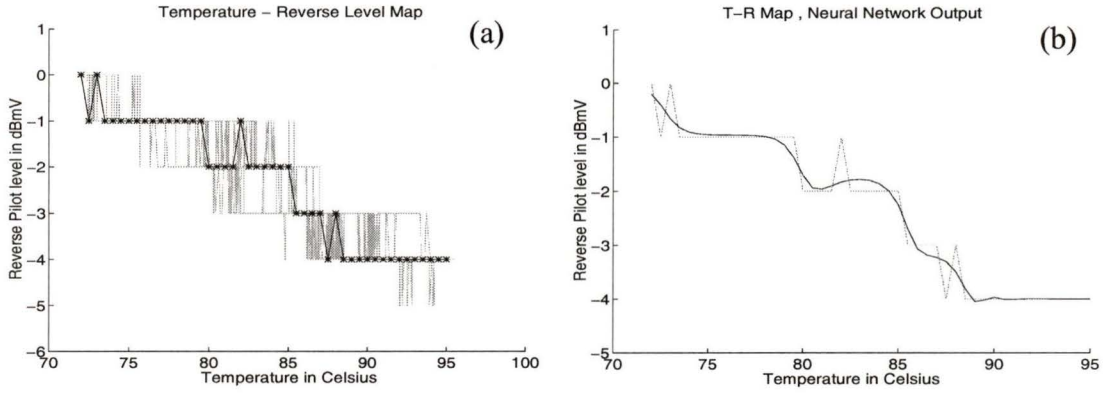

FIGURE 4-11.
(a) *R-T* map (gray trace) and generated dependency function (dark trace); (b) A replicate of the dependency function of (a) (light gray line), and the NN's approximation (dark line).

The trained network was used to model the reverse pilot over a period of several days. The results of modeling the signal of Figure 4-8 are shown in Figure 4-12. This approach yields a good prediction of the reverse pilot in the temperature range that the neural network had been trained on. The neural network however, was not able to simulate correctly[9] the reverse signal (see first 6,000 samples) for temperatures that exceed the temperatures it had been trained on.

The main disadvantage of the approach presented in this section is that the neural network is trained on the statistically processed data[10], data which is missing information which the neural network could process and learn from. For example, the neural network is never presented with the information of which is the second and third most occurring level for each temperature. In the extreme case where the most frequently occurring level for each temperature represents 34% of the occurrences (the second and third most occurring level represent each 33% of the occurrences), the "lost" information may be as much as 66%.

---

9. This topic is discussed in more detail in Chapter 5.

10. By pre-processing the data before training the neural network we can reduce by one decimal point the size of the training set, resulting in much faster trainings.

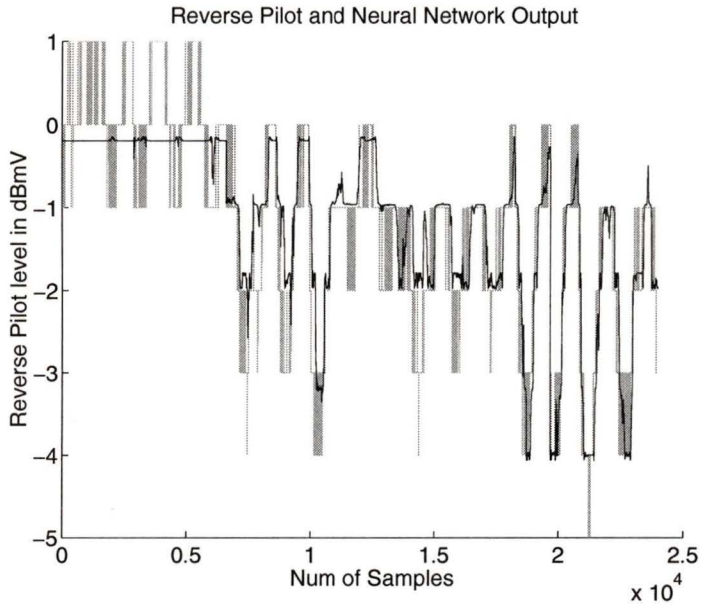

FIGURE 4-12.
The neural network (dark line) tracks the reverse pilot (gray trace) well for temperatures close to the temperatures it has been trained on (after sample No. 7,000).

A more general approach to the modeling of the (*T-R*) dependency using neural networks is to use directly the temperature and reverse pilot measurements to train the network. A feed-forward neural network can model even more accurately the relationship, without the statistical pre-processing of the data. This is precisely the strength of neural networks: their ability to capture the underlying relations and rules of a process without any assumptions, but rather by being presented with examples. In the following chapter, a full description of such a neural network modeling engine is presented.

4.7 Summary

Cable television amplifier networks distribute cable signals from a centrally located injection site (head-end) to subscribers' televisions. Typically, they are two-way asymmetrical communication networks, with the downstream (forward pilot) path, used mainly for delivering cable television services. The upstream (reverse pilot) path, is used to provide a data path from the subscribers to the head-end for use in interactive services. Consequently, the requirement to provide a high quality reverse path has become increasingly important.

One way of improving the reliability of the network is by implementing accurate fault detection techniques. Currently, in Rogers Cablesystems amplifier plants, limit checking is used to perform fault detection. There are, however, limitations and disadvantages in using limit checking. In particular, limit checking does not take into account the temperature driven variations of the reverse signal, which results in a less accurate fault detection. On the other hand, model based techniques can be used to overcome some of the limitation of the traditional limit checking.

In the last sections of this Chapter two introductory model-based approaches to fault detection for the reverse pilot were briefly described. These methods, although incomplete, offer a starting poind and present some of the fundamental concepts and techniques that are used in the development of the complete methods presented in the next chapter. As these methods are more easy and straight forward to implement, the further and possible improvement of the two incomplete methods is left as an open research topic.

Chapter 5:

Model-Based Fault Detection for the Reverse Pilot

5.0 Introduction

This chapter describes in detail a novel fault detection system that has been developed for the reverse pilot of cable television networks. As the developed system is based on the use of a modeling engine to model the reverse pilot dependence on the temperature, two modeling engines developed for this purpose are presented. The first modeling engine uses a back propagation neural network to learn the reverse-temperature dependency; the second modeling engine is not neural but captures the reverse pilot behaviour for a range of temperatures by implementing statistical analysis on the two signals. The details of the two modeling engines are discussed extensively in this chapter, and the results from a large scale experiment on which both the modeling engines were tested are presented.

5.1 Modeling of the Reverse Pilot

In the following sections a model-based technique is presented that addresses the problems associated with the use of the limit checking fault detection scheme that is currently used in Rogers cable plants. A general approach has been implemented based on the use of a modeling engine to capture the dynamics of the reverse pilot of cable television amplifiers. Using this technique, the process of fault detection becomes a process of detecting behavioural changes within the plant's components.

Any attempt to detect changes in behaviour has to account for the temperature driven variations of the reverse signal noted in Section 4.4.2. The approach presented here is based specifically on this observation. Namely, a model of the dependency of the reverse pilot measurements on the temperature is developed and then this model is used to detect any deviation of the measurement from that which the model indicates. Any such deviations are deemed to signify a change in behaviour which may be associated with the onset of a fault[1]. A block diagram representation of the above model-based approach is illustrated in Figure 5-1.

---

1. Detecting changes in behaviour is a sufficient condition for fault detection, since by definition a fault will cause a change in behaviour.

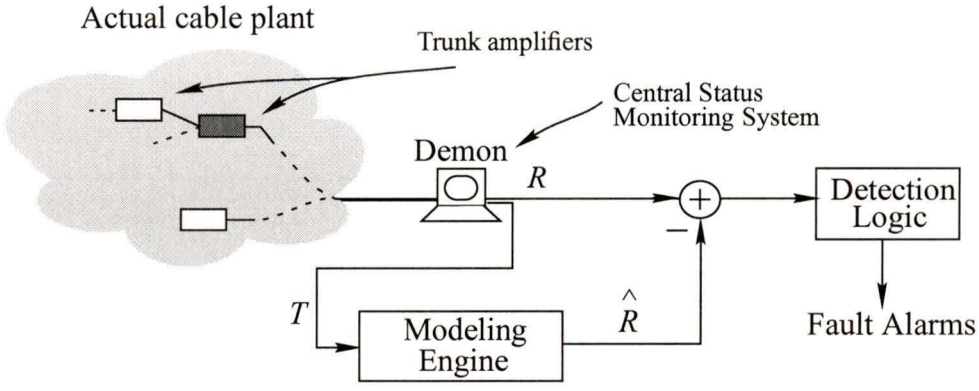

FIGURE 5-1.
Modeling of the reverse pilot - temperature dependency. Fault detection can be performed by comparing the modeling engine's estimate with the actual output of the amplifier.

For the work in this thesis, several modeling engines were developed and experimented with. Out of these, two have emerged as having superior modeling capabilities and are presented in the following sections. The first one is based on the use of neural networks (Neural Network Modeling Engine-NNME) and the second one is based on statistical processing of the reverse and temperature signals (Statistical Modeling Engine-SME).

5.2 Neural Network Modeling

Neural networks are a natural choice for black-box modeling of poorly specified processes, where no analytical models can be constructed and an abundance of sensor information is available. In the case of the reverse pilot, since no analytical model exists and a lot of sensor information is available, neural networks can be used to model it. In addition, the neural network modeling approach is flexible in the sense that the neural network can be re-trained with a relatively low overhead to update or create a new model, when for some reason the nature of the reverse pilot changes. Such changes may be attributed to network reconfigurations, repairs or replacement of components, software reconfigurations or to fault events. In the latter case of behavioural changes due to faults, it is desirable that the modeling engine be able to model the "faulty" behaviour for two reasons:

first, a further change of the "faulty" behaviour may signify either the occurrence of a second fault or the repair-mitigation of the fault and subsequent return to normal behaviour. Secondly, the analysis of the model's output during faulty periods may contribute to the diagnosis of the fault[2].

5.2.1 Feed Forward Neural Network Modeling

As mentioned in Section 4.4.2 the relation between the reverse pilot and the temperature seems to be non-linear and wide-sense stationary. As such, a feedforward neural network with the appropriate architecture and characteristics would be capable of learning this relationship and hence model the reverse pilot. An example of the capability of feedforward neural networks to model the reverse pilot is shown in Figure 5-2 and Figure 5-3.

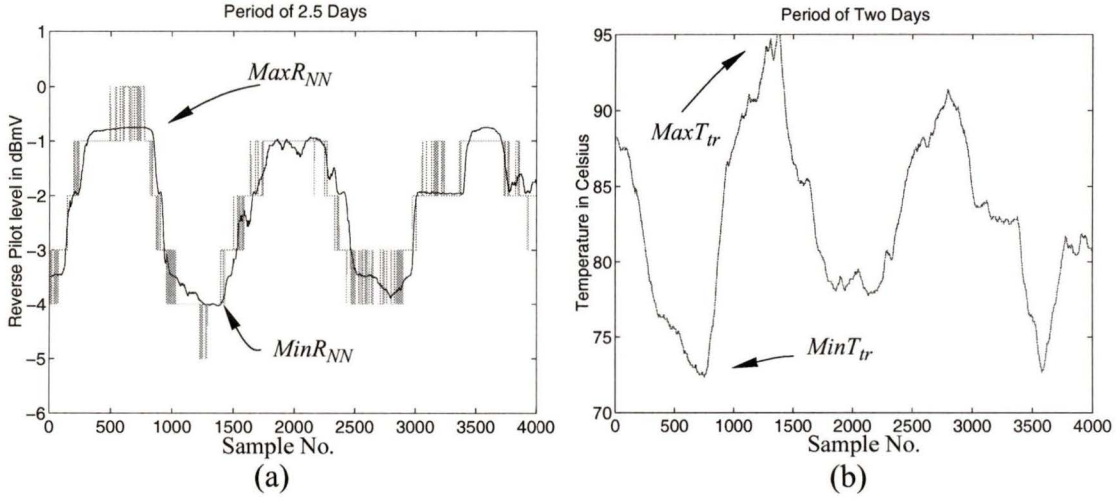

FIGURE 5-2.
The data set (reverse pilot (a) and temperature (b)) used to train the neural network corresponds to the last 4,000 data points of the modeling set of Figure 5-3. The dark line in plot (a) is the neural network approximation.

The neural network is trained using a small training set (Figure 5-2) of 4,000 reverse pilot and temperature pairs which correspond to 2.5 days of collected data, and models the reverse pilot using as input temperature that of a larger set of several days (25,000 samples). The input temperatures are shown in Figure 5-3(b).

2. The analysis of the difference between the model and the actual behaviour, could contribute in the classification of the faults, and hence help the diagnosis procedure.

This approach yields a good estimate of the reverse pilot in the temperature range that the neural network has been trained on. The neural network, however, is not able to model the reverse pilot signal for temperatures that exceed the training temperature range. For temperature inputs outside this range, the neural network produces as an output one of the "extreme"[3] levels $MinR_{NN}$ or $MaxR_{NN}$ (Figure 5-2a). That is, if the input temperature is higher than the maximum training temperature $MaxT_{tr}$, the network outputs $MaxR_{NN}$, and if it is lower than the minimum training temperature $MinT_{tr}$, the network outputs $MinR_{NN}$. This can clearly be seen in Figure 5-3(a), where for the first 6,000 samples, (which correspond to temperatures much lower than those the network was trained on-Figure 5-3b), the network fails to track the reverse pilot signal. For the remaining 20,000 samples the network tracks the reverse signal well, both in the training region (Figure 5-3) and in the "free running region"[4], where the NN interpolates the behaviour of the system based on the knowledge it has gained during training (Figure 5-3).

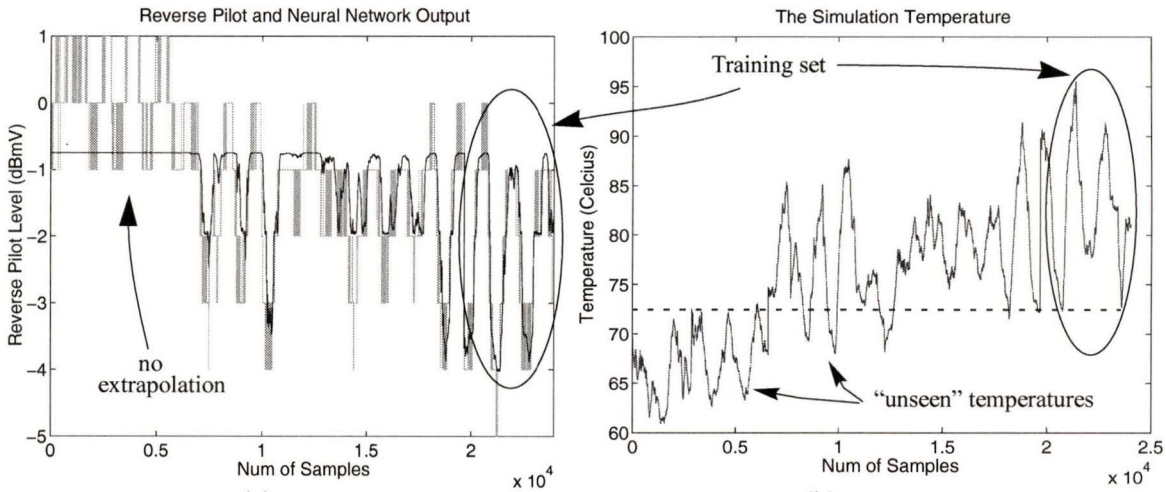

FIGURE 5-3.
a) The modeling set (gray trace) and the NN estimation (dark line); b) The training and the simulation temperatures. The data set corresponds to a period of approximately 15 days.

---

3. The extreme reverse levels $MinR_{NN}$ and $MaxR_{NN}$ are defined as the neural network's output for input temperatures $MinT_{tr}$ and $MinT_{tr}$ (i.e. $MinR_{NN} = NN(MinT_{tr})$, $MaxR_{NN} = NN(MaxT_{tr})$).

4. The region on which the neural network has not been trained.

The fact that the Feed Forward Neural Network (FFNN) is unable to extrapolate for temperatures that are outside the temperature range the network has been trained on, leads to the requirement of re-training the network each time the network has to estimate the reverse level for "unseen" temperatures. Therefore, in order for the network to model the reverse signal accurately, the new training sets have to include the "unseen" temperatures.

The preliminary results presented in this section show that feed-forward neural networks can model the reverse pilot. As such, further experiments were performed to determine the most appropriate neural network architecture, characteristics and training and re-training heuristics for the neural network modeling engine. These are presented in the following sections.

5.2.2 Architecture

Since the architecture and characteristics of the neural network that performs the best modeling depends on the process to be modeled, and as no theoretical guidelines to determine optimal architectures exist, the characteristics of the network have to be determined experimentally.

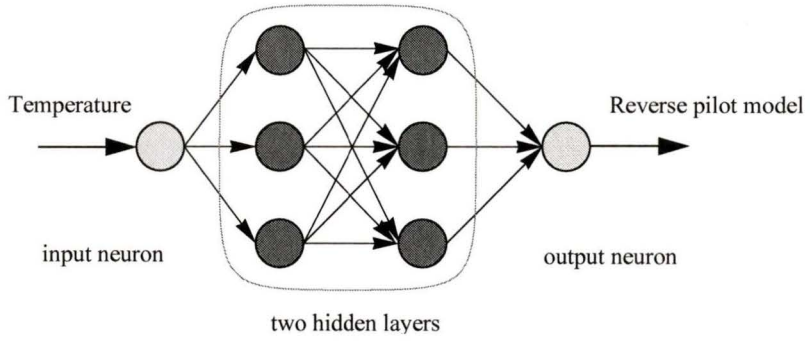

FIGURE 5-4. The architecture of the feed forward neural network used.

The number of input and output neurons is one of the quantities that are determined by the specific application for which the neural network is used. For the modeling of the reverse pilot, the temperature is used as the input of the neural network and the reverse pilot estimate is the output of the network. Consequently, a single input - single output NN is used.

As noted in Chapter 2, there is no way of knowing *a priori* the number of hidden layers and neurons that makes adequate the performance of a neural network. For each application, these factors have to be determined experimentally. Experiments show that two hidden layers were sufficient to learn the reverse-temperature mapping. As such, in all the experiments performed in this chapter, a neural network with 2 hidden layers is used (Figure 5-4).

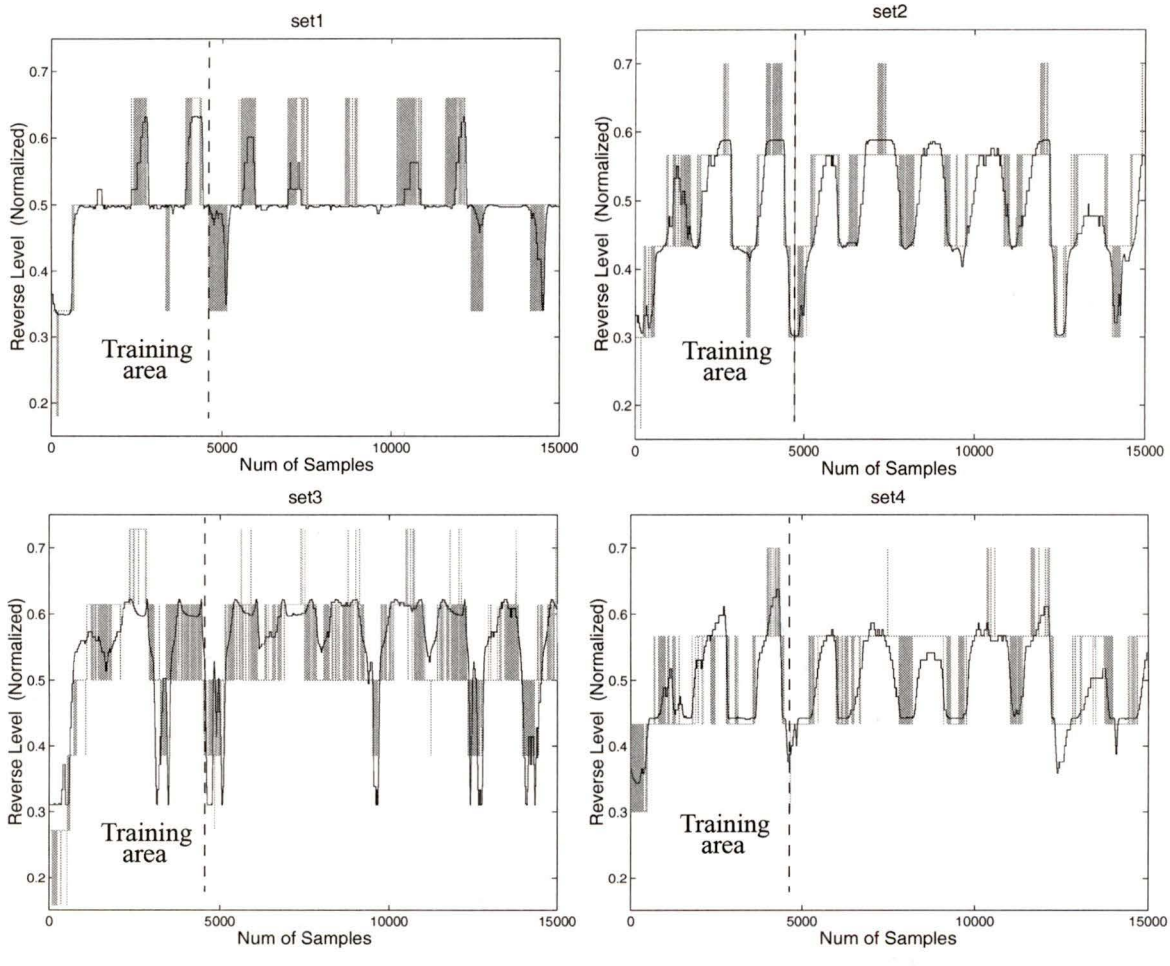

FIGURE 5-5.
Reverse pilot benchmark data sets (gray trace) used for training and modeling. The dark solid line is the output of an example neural network with 2 hidden layers composed of 3 neurons each.

5.2.3 Determination of the Number of Hidden Neurons

To determine the most appropriate number of neurons for the two hidden layers of the neural network, a simple experiment was set. Several trainings[5] were performed using the same training sets and network architecture. The rms error between the NN's output and the reverse pilot were calculated, both for the training set (training error) and for a larger simulation set (modeling error). The above procedure was repeated for four different data sets and different networks with increasing number of hidden neurons, with the smallest network having one neuron in each layer and the largest network having six neurons in each hidden layer. The benchmark data sets used in this experiment are illustrated in Figure 5-5. The average training and modeling error (normalized) over all four training sets and for all six networks are presented in Table 5.1.

Table 5.1: A comparison of the training duration and training and modeling errors for networks with 2 hidden layers and different number of hidden neurons.

Num of Neurons	Time (normalized)	Training rms error	Modeling rms error
1	1.000	1.392	1.509
2	2.005	1.169	1.221
3	2.834	1.054	1.174
4	3.687	1.017	1.174
5	4.579	1.028	1.190
6	5.550	1.000	1.183

Consistent with the discussion in Chapter 2, we see that networks with different numbers of hidden neurons result in different training durations as well as different training and modeling rms errors. The results of this experiment clearly show that, as the number of hidden neurons increases, the training durations are longer and the training rms generally decreases as a result of a better learning of the *T-R* mapping. On the other hand, the modeling rms error increases after reaching its minimum value using 3 neurons as a result of the poor generalization of networks with many hidden neurons. Figure 5-6 illustrates the results presented in Table 5.1.

5. The training heuristics used in this experiment are those presented in the following sections.

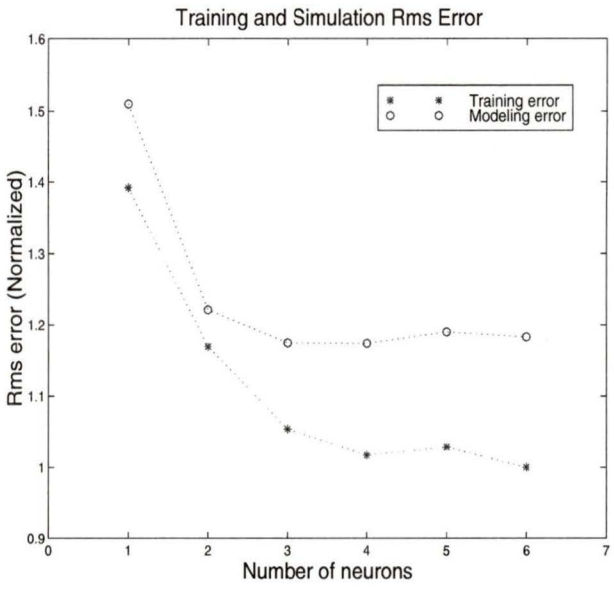

FIGURE 5-6.
The average training and modeling rms error for networks with various numbers of hidden neurons.

5.2.4 Activation Functions

A prerequisite for the use of back-propagation as the training algorithm is that the activation functions used are differentiable and non-decreasing. The family of sigmoid functions satisfy the above conditions and therefore are widely used as activation functions in feed-forward neural networks. The activation function used in the two hidden layers as well as in the output neuron of the neural network is the logistic sigmoid function. The logistic sigmoid (*logsig*) can be produced from the general sigmoid equation (Eq. 2.5) by setting the constant σ to one.

$$\log sig\,(x) \;=\; \frac{1}{1+e^{-x}} \tag{5.1}$$

There is no activation function used in the input layer (single input neuron in this case). The role of the input neuron is to simply copy the inputs and pass them to the first hidden layer neurons through its weighted interconnections.

5.2.5 Normalization Function

The input (temperature) as well as the target vector (reverse pilot level) of the neural network must be in the range $[0, 1]$ of the activation function[6]. Therefore, both signals have to be normalized to this range.

The temperature is normalized in that range by using a normalization function $n_T(.)$, which maps the two extreme temperatures of $-10\ ^oC$ and $110\ ^oC$ to zero and one respectively. All temperatures between these two temperatures are mapped uniformly between zero and one. The temperature normalization function is the same for all temperature signals and is given by:

$$n_T(T) = \frac{T + 10}{120} \tag{5.2}$$

In order to increase the dynamic range of the normalized reverse signal, a non-uniform normalization is applied. The normalization is applied to the signal such that 98% of the data around the nominal level, is normalized in the range $[0.1, 0.9]$. The remaining 2% of the signal, which typically consists of spikes and rarely occurring levels attributed mainly to noise bursts, is normalized in the ranges $[0, 0.1]$ and $[0.9, 1]$.

Initially, the lowest and highest reverse levels (R_l and R_h) are found, within which at least 98% of the signal is contained, which consist the $rangeR = [R_l, R_h]$ of the signal[7]. Also the minimum and maximum levels of the given reverse signal are found, which are denoted as $MinR$ and $MaxR$. The normalization is then performed in two steps: first, using a main normalization function $n_m(.)$, the whole data set is normalized such that R_l is mapped to 0.1 and R_h to 0.9, and all the levels in between are mapped linearly in the interval $[0.1, 0.9]$; in the second step, the levels that are outside the range $[R_l, R_h]$ are

---

6. The normalization is done for two reasons: first, it reduces the training time, and secondly, the non-uniform normalization diminishes the influence of outliers in the training process (See Appendix A).

7. The levels that belong to *rangeR* are referred to as "active" levels.

normalized in the ranges $[0, 0.1]$ and $[0.9, 1]$. For these levels, the linear functions n_l and n_h are used to map the low $(x < R_l)$ and high $(x > R_h)$ end of the signal in these ranges respectively if $n_m(MaxR) > 1$ and $n_m(MinR) \leq 0$. The normalization process can be expressed by the following mathematical function:

$$F_{norm}(x) = \begin{cases} n_h(x) & if \quad x > R_h \quad and \quad n_m(MaxR) > 1 \\ n_m(x) & if \quad x > R_h \quad and \quad n_m(MaxR) \leq 1 \\ n_m(x) & if \quad R_l \leq x \leq R_h \\ n_m(x) & if \quad x < R_l \quad and \quad n_m(MinR) \geq 0 \\ n_l(x) & if \quad x < R_l \quad and \quad n_m(MinR) \leq 0 \end{cases} \tag{5.3}$$

where $[R_l, R_h] \overset{n_m}{\rightarrow} [0.1, 0.9]$, $[MinR, R_l] \overset{n_l}{\rightarrow} [0, 0.1]$,

$[R_h, MaxR] \overset{n_h}{\rightarrow} [0.9, 1]$.

The functions n_m, n_h and n_l are linear functions with ranges and domains as above.

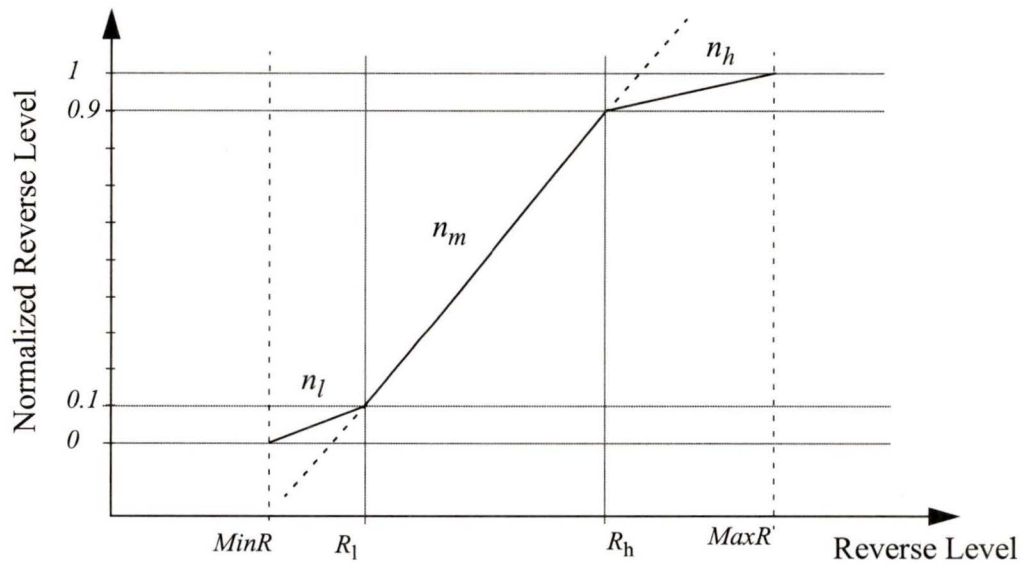

FIGURE 5-7.
The graphical representation of the normalization function. The function is different for each data set.

The normalization function $F_{norm}(.)$ can be seen in Figure 5-7. Obviously, the normalization function used for each reverse signal (amplifier) is different, and depends on the $rangeR$ and on the values of $MinR$ and $MaxR$.

5.2.6 Modeling Heuristics

The task of the modeling engine for the model-based fault detection system presented in this chapter, is to model the reverse pilot as accurately as possible by capturing its behaviour from a limited number of observations. Consequently, each time new input temperatures are encountered or changes in behaviour are discovered by the detection system, a new model needs to be created. For this purpose, new training sets must be constructed, and the modeling engine must re-trained to capture the updated or "new" behaviour.

The accurate modeling of the reverse pilot when using the neural networks modeling engine depends heavily on the implementation of heuristics that improve the trainings. By trainings we refer to the initial trainings, the temperature re-trainings (re-trainings that occur because of the "out of range temperatures") and the behavioural re-trainings (re-trainings triggered by detected changes in behaviour).

For a given architecture there are ways to improve the modeling capabilities of the neural network. This can be done not only by enhancing the training algorithm with additional heuristics, such as adaptive learning rate and momentum, but also by including algorithms that decide when to perform the re-trainings and what the new training set should be. In the following sections the modeling heuristics are discussed in more detail.

5.2.7 Training the Neural Network

An adaptive learning rate is used to update the network's interconnection weights during training, which as noted in Section 2.4.1, has significant advantages over a constant learning rate. The adaptive learning rate is implemented in the NNME as follows: while the "new" training error is lower than the previous training error the learning rate lr increases by a factor In_{lr}:

$$lr_{new} = lr_{old} \cdot In_{lr} \tag{5.4}$$

where lr_{old} represents the learning rate used in the last weight update,

lr_{new} is the learning rate to be used in the upcoming weight update,

and In_{lr} is the increment factor ($In_{lr} = 1.5$).

If the new training error gets higher than the old training error, such that the ratio of the new error over the old error is greater than a preset error ratio Er, ($er_{new}/er_{old} \geq Er$), then the learning rate is decreased:

$$lr_{new} = lr_{old} \cdot Dcr_{lr} \tag{5.5}$$

where Dcr_{lr} is the decreasing factor ($Dcr_{lr} = 0.7$)

and the preset error ratio $Er = 1.04$.

In addition a momentum term is added to the weight update (Eq. 2.14) to help the training escape shallow local minima (See "Momentum" on page 18.). The momentum constant c_m of Eq. 2.14 has a non-zero value only when the new error is lower than the old error:

$$c_m = \begin{cases} 0.9 \ if \ er_{new} \geq er_{old} \\ 0 \ if \ er_{new} < er_{old} \end{cases} \tag{5.6}$$

The effects of the momentum can clearly be seen in Figure 5-8, where at certain instances of the training the error increases as a result of the search moving opposite to the direction of steepest descent in order to overcome shallow local maxima.

5.2.7.1 Number of Training Epochs

Ideally, the neural network should be trained for as many epochs as it takes for it to learn the given *R-T* mapping sufficiently well, i.e. until a certain predetermined training error is reached. Unfortunately, this is not practical or even feasible, since some trainings may last several hours, or may never even reach the desired training error.

Alternatively, the trainings are implemented as such so that the neural network is trained for a preset number of epochs. Experiments show that, generally, 1,000 epochs are sufficient for the neural network to learn adequately well an *R-T* mapping. Figure 5-8 shows the training error as it decreases with the number of epochs.

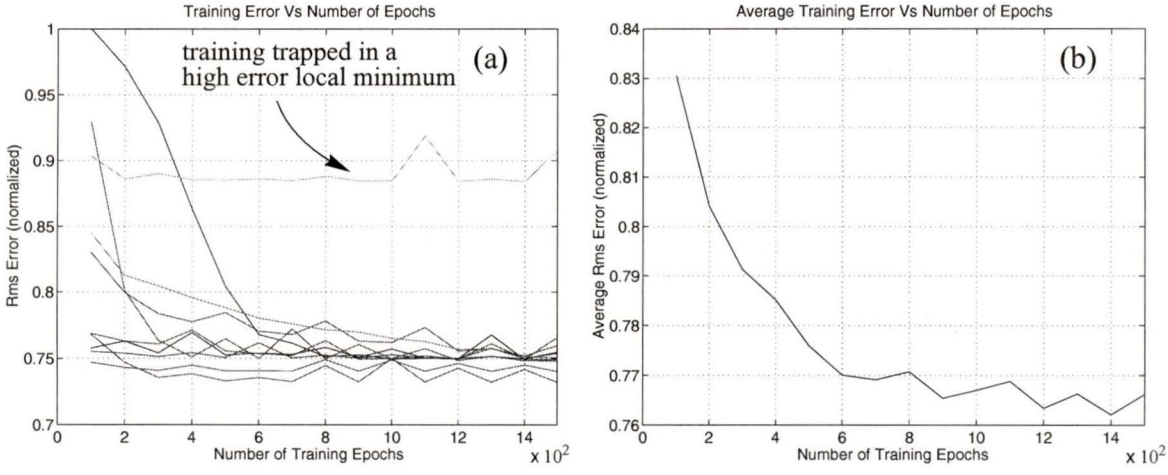

FIGURE 5-8.
(a) Training error versus the number of epochs for 10 different trainings (on reverse pilot data). Notice the training that was trapped early in a high error local minimum, and the effects of the momentum term. (b) Average training error versus number of epochs.

Figure 5-8(a) plots the training error for 10 different trainings versus the number of epochs, while Figure 5-8(b) shows their average training error for each epoch. It can be easily seen that the error improvement after 1,000 epoch is relatively small. As such, all

the trainings in the experiments of this chapter will be performed with 1,000 training epochs. In addition, as described in the next section, further training error improvement can be achieved by implementing the concept of initial weight space explorations.

5.2.7.2 Initial Weight Space Explorations

As noted in Section 2.5.2, initial weight space explorations[8] play a critical role in the success of the trainings. Through their implementation, trainings which start or get trapped early in high error local minima, such as the one shown in Figure 5-8(a), can be avoided. As shown in this section, the initial path searches are critical mostly for small networks with a low number of hidden neurons. Although it is not guaranteed that an initial training with the lowest error will produce the lowest final error after the full training is over, on the average they produce better final errors (see Figure 5-9). A simple experiment was performed to demonstrate this fact and determine the optimal number of initial trainings.

Several initial trainings[9] of 200 epochs each, and full trainings of 800 epochs[10] were executed on the four training sets used in the experiment of Section 5.2.3 (Figure 5-5). The final training rms error (rms of the full trainings), that correspond to the extension of the "best" initial trainings were calculated for 1 to 15 initial trainings, and for different networks with 1, 3 and 6 neurons in each of the two hidden layers. The results of this experiment are shown in Figure 5-9, where the average rms error is plotted for each number of initial re-trainings, and for all three networks.

The error, as expected, decreases for all three networks as the number of initial trainings increases. It is worth noting that, depending on the number of neurons, five initial searches are enough to produce more than 80% of the possible error improvement. This can be clearly inferred from Figure 5-10(a), which shows, for 1-15 initial searches, the average percentage of improvement of the final error over the training error that would

---

8. Referred to also in this thesis as "initial trainings".

9. Before each initial training the weights are reset to small random values (see Section 2.5.2).

10. Preliminary experiments show that 200 epochs are sufficient to give a good indication of the course of the training. These 200 epochs are added to the 800 epochs of the full training to give a complete training of 1,000 epochs.

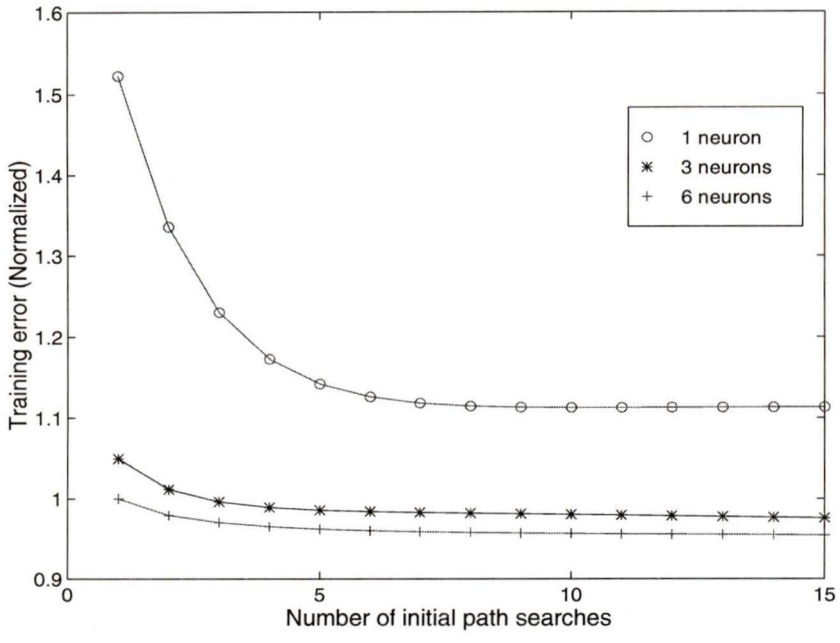

FIGURE 5-9.
The average rms training error for different number of initial trainings and for three networks with 1, 3 and 6 neurons in each hidden layer.

result in the absence of the initial searches. The improvement for all three networks is shown on the same plot, with 0% corresponding to trainings with "no initial path searches"[11].

From Figure 5-10(a) we can see that for networks with one neuron, initial path searches can improve on the average up to 27% the overall training performance (error). As the number of hidden neurons increases, the potential improvement decreases to 7% for a network with three neurons in each hidden layer, and to 4% for a network with six neurons. It can be seen that, in terms of training error improvement, initial path searches can offer more to small networks than to networks with a larger number of hidden neurons (for the reverse pilot data) . It should be noted, that the relative improvements for the three networks cannot be compared to each other, since they represent percentages of different error values. These error values correspond to the errors of the "no initial search" trainings of Figure 5-9.

[11] Trainings with "no initial path searches" are essentially trainings with a single initial path search, and that is why in the plots they are associated with the number one (1) for initial trainings.

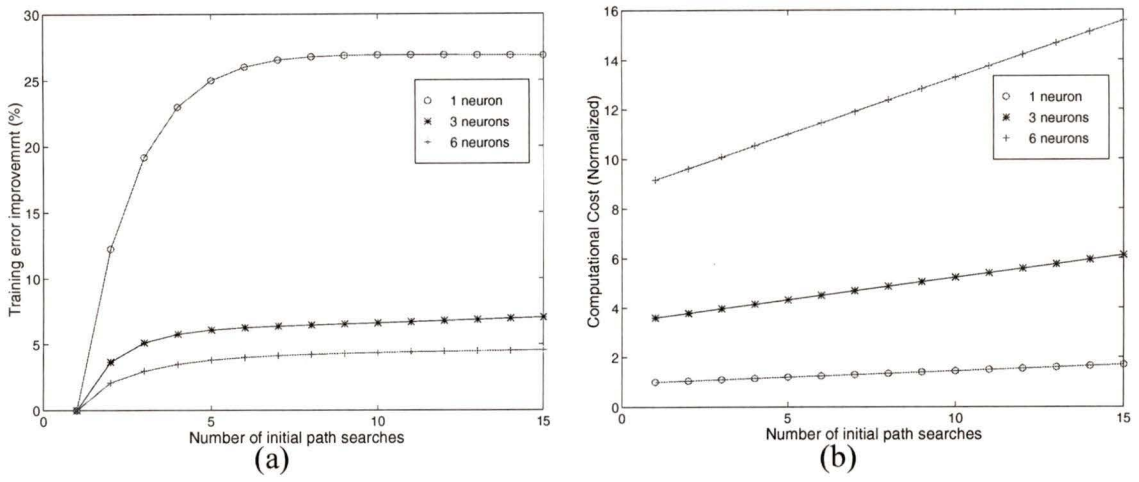

FIGURE 5-10.
(a) The percentage of improvement of the training error with 1-15 initial path searches with respect to the training error on the absence of initial searches (for all three networks). (b) The computational cost for the three networks and for 1- 15 initial path searches.

Consistent with the results from the experiment of Section 5.2.3, this experiment also suggests that larger networks on the average produce lower rms training errors. The trade-off for using large networks is the increased computational cost of the trainings. This fact is shown in Figure 5-10(b), where the normalized computational cost is plotted for all three networks and for different number of initial trainings. The error increases linearly as the number of initial trainings gets higher, with a slope that increases with the size (number of neurons) of the networks.

To summarize, the results of this experiment suggest that initial trainings improve on the average the training error. Independently of the number of neurons (1-6), more than 85% of the improvement that is attained with fifteen initial trainings, can be achieved with five initial trainings. Larger networks produce better training error, at the expense of a higher computational cost, but as their size increases a diminishing training error improvement is returned (Figure 5-9). As such, in the neural network used it the NNME, 3 neurons are present in each of the 2 hidden layers, and 5 initial trainings are performed for each training or re-training.

5.2.8 Temperature Re-trainings

Each training set consists of reverse pilot and temperature pairs. The minimum and maximum temperature values of a training set are denoted as $MinT_{tr}$ and $MaxT_{tr}$, and the range between these two temperatures constitutes the training temperature range $RangeT_{tr} = [MinT_{tr}, MaxT_{tr}]$. The neural network is trained to associate a reverse level for each temperature in the range $RangeT_{tr}$. A network with an appropriate architecture will interpolate for temperatures that it has not been trained on but, nevertheless, reside inside the range $RangeT_{tr}$ without *overfitting* or *underfitting*[12]. However, as shown in Section 5.2.1, a neural network will not extrapolate for temperatures outside the $RangeT_{tr}$ (unseen temperatures).

Training sets are normally a small subset of the modeling sets and they do not generally contain, within $RangeT_{tr}$, all the temperatures that may occur. Consequently, "unseen" temperatures are often inputs to the neural network. In order for the modeling engine to model the reverse pilot accurately, it must be capable of producing a valid reverse level estimate for all the possible amplifier enclosure temperatures. This means that the neural network must be re-trained (temperature re-trainings) with updated training sets which include the unseen temperatures.

All temperature re-trainings are done in the same manner, as follows: during the modeling phase a search is performed on the simulation set (in blocks of 1000 samples- half a day) for temperatures that exceed the training range $RangeT_{tr}$. If such temperatures exist, the modeling stops. The *T-R* pairs that correspond to the temperatures that are out of range are included in the "new" training set, and a temperature re-training occurs. Once the network is re-trained, it models for 1000 points and the "temperature check" is performed again on the subsequent 1000 temperatures. For each temperature re-training, the initial training weights are reset to random values and five initial path searches are performed.

---

12. Both *overfitting* and *underfitting* are results of bad interpolation due to the use of inappropriate neural network architectures. For definitions see [7].

Temperature re-trainings are necessary for the accurate modeling of the reverse pilot. Figure 5-11 shows two models of the same set: without temperature re-trainings and with temperature re-trainings. The resulting improvement is obvious.

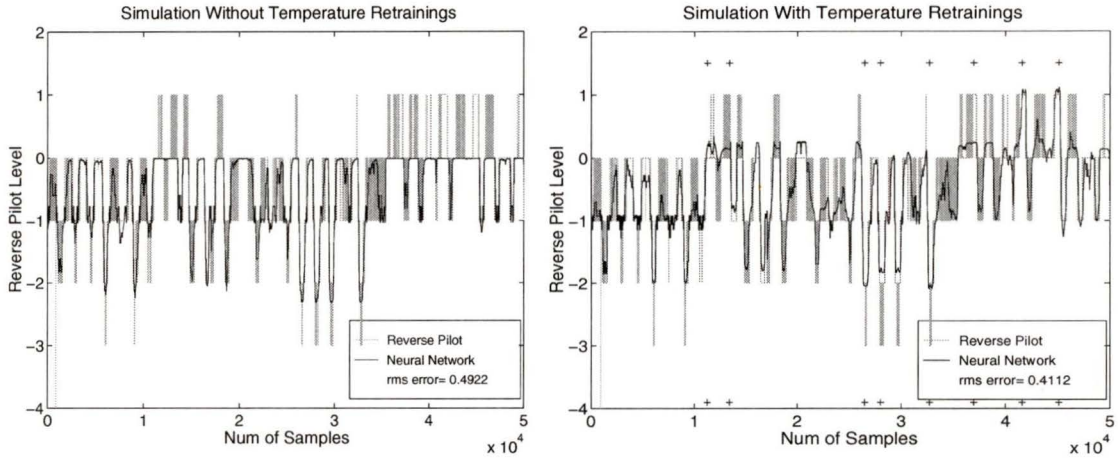

FIGURE 5-11.
Modeling without temperature re-training and with temperature re-trainings. The locations where temperature re-trainings occurred are marked with a cross. The data set corresponds to a duration of one month.

5.2.9 Behavioural Re-trainings

After the temperature check is done (and possibly a temperature re-training if necessary), a routine that "looks" for behavioural changes is invoked: every 1,000 data points (approximately 15 hours), a routine compares the NNME output to the reverse signal, and if a significant[13] error exists, a *reverse alarm* is generated[14] and a behavioural re-training is performed. For this kind of re-training, a new training set must be generated which captures to a satisfactory extent the new behaviour of the reverse signal, namely the new *T-R*

---

13. The error thresholds used are presented in the Section 5.2.9.3(See "Reverse Alarms" on page 77.)

14. The temperature re-trainings are performed before the "behavioural change" checks. Hence, if a change in behaviour is present in the set on which the temperature check is performed, it is possible that in the "new" training set *R-T* pairs that belong to the new behaviour will be included. However, this will result in a training set with mixed behaviour and cause the network to train poorly. As a result, an alarm will be produced in the subsequent behavioural change check, successfully identifying the change in behaviour.

dependency. As such, the old training set which represents the old behaviour is replaced entirely by a new training set, composed of the subsequent 4,000 data pairs. An example of reverse alarms and behavioural re-trainings is shown in Figure 5-12.

5.2.9.1 Changes in Behaviour [15]

Figure 5-12 presents the behaviour of the reverse pilot over a period of two months and, as can be seen, the NNME tracks the behaviour quite well. In addition, the fault detection algorithm "discovers" changes in behaviour indicated by the dark solid vertical lines. These alarms indicate a divergence of the model from the actual observations for an extended period of time.

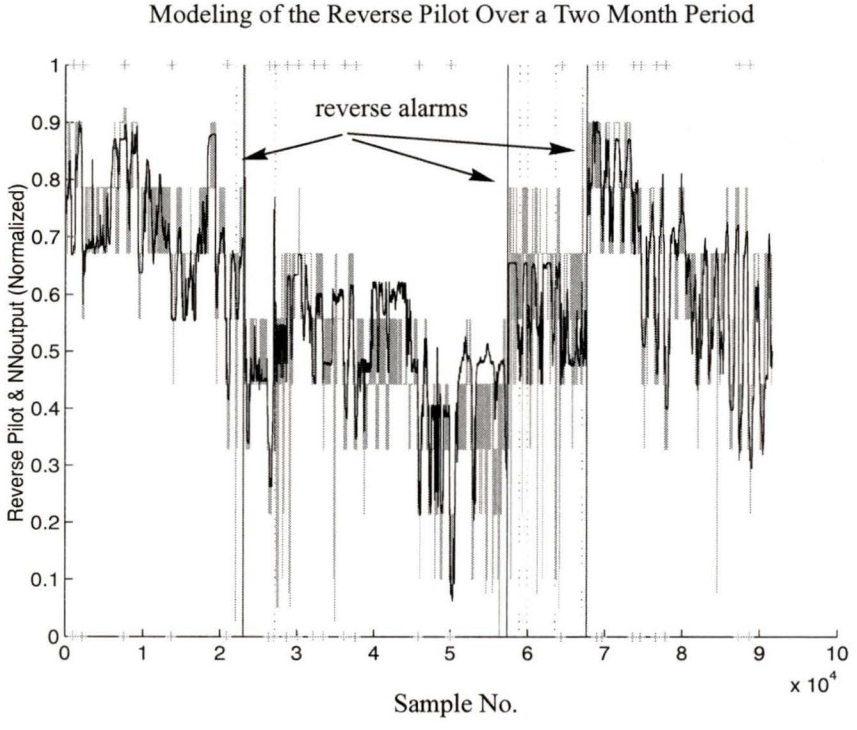

FIGURE 5-12.
Example of a modeling with three reverse alarms (behavioural re-trainings). The gray trace is the reverse pilot, and the dark trace is the neural network. The location of the alarms is marked with dark vertical lines. The temperature re-trainings are marked with crosses on the top and bottom of the plot.

---

15. The term *change in behaviour* follows the definition given in Chapter 4, wher s defined as the change in the relationship (dependency) of the signal to the temperature- change in the dependency function of Eq. 4.3.

It was also validated that the changes in behaviour which were discovered, did indeed correspond to different behaviours of the system. This was established by comparing the reverse pilot-temperature relation before and after the change of behaviour events. Figure 5-13 plots these relations and it is evident that they are markedly different.

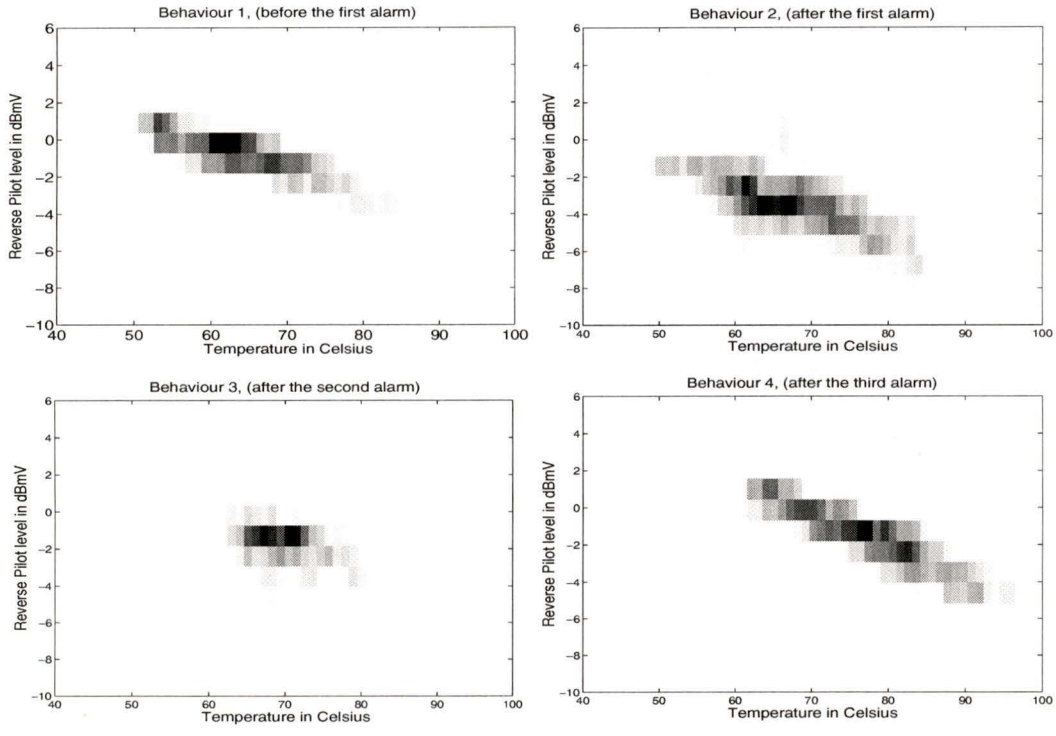

FIGURE 5-13.
Four different behaviours of the reverse pilot signal shown in Figure 5-12.

Figure 5-13 shows the reverse pilot levels that correspond to each temperature of each data subset, with the gray scale representing the density of occurrence of each level (dark corresponds to high density). The behavioural change before and after the alarms are revealed, in this particular case, by the drop in the reverse level that corresponds to the same temperature, and by the changes in the width and slope of the distributions.

5.2.9.2 Correlation Events

As in the example presented in the previous section, behavioural changes of the reverse pilot manifest themselves, most commonly, as sudden drops or rises of their measured level, which do not follow the corresponding temperature variations. Sometimes, behavioural changes are conveyed by changes in the correlation relationship between the reverse pilot and the temperature.

As discussed in Chapter 4, fluctuations in the reverse pilot signal level are normally negatively correlated with changes in the temperature signal. It has been observed, however, that in certain cases the reverse pilot variations "switch" from being negatively correlated to the temperature to being positively correlated or not-correlated.

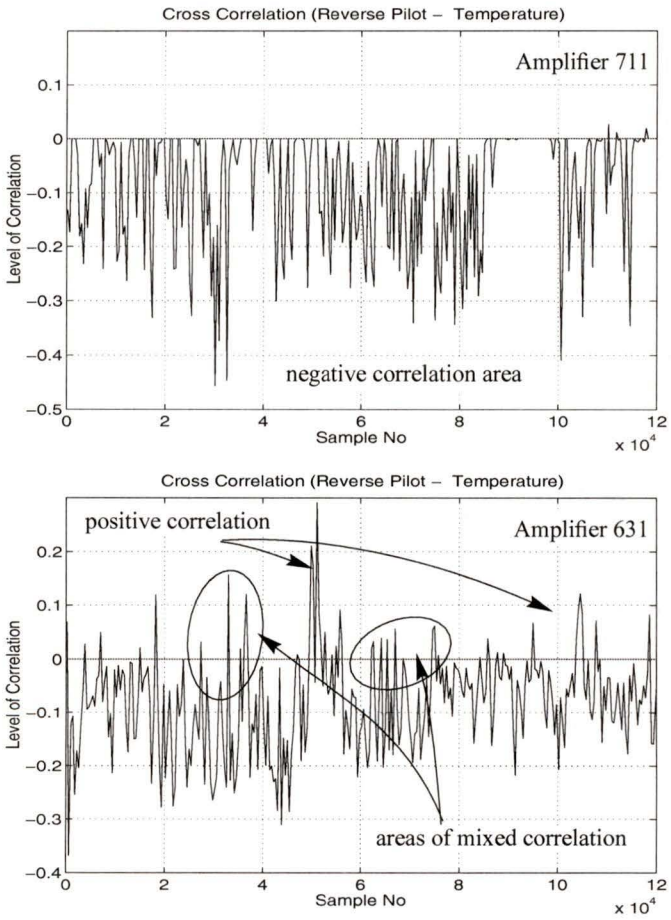

FIGURE 5-14.
The reverse pilot - Temperature cross-correlation for amplifiers 711 in subplot (a) and amplifier 631 in subplot (b). The correlation for amplifier 711 is consistently and strongly negative, while for amplifier 631 the correlation is "mixed".

As will be shown in Section 5.2.10, the model-based fault detection system presented in this chapter is able to identify these kind of behavioural changes and produce an alarm when they occur (see Figure 5-17 on page 82). In these cases, the presence of such "anomalies" in the correlation of the signal can be verified with the use of a correlation detector which was developed for this purpose. The correlation detector is presented in Appendix B.

In Figure 5-14, the reverse - temperature cross correlation for two amplifiers (711 and 631) is shown. The correlation is computed over a period of three months and with the use of the correlation detector. In the subplots (a) and (b) of Figure 5-14, the areas which correspond to negative values of the *Y*-axis denote negative cross-correlation of the two signals, while the areas with positive *Y*-axis values denote positive correlation. It can easily be seen that the reverse pilot signal of amplifier 711 is strongly negatively correlated to the temperature consistently throughout the whole three months. In contrast, for amplifier 631 the *R-T* cross-correlation is not consistently negative. As can be seen in Figure 5-14(b), during certain periods the correlation becomes strongly positive, while during other periods the correlation can more accurately be characterized as "mixed" (a mixture of both positive and negative correlation).

It is evident that the correlation detector potentially could be used by itself to identify behavioural changes[16] that result in alterations in the correlation of the reverse pilot signal. In this work, however, it is only used to investigate the presence of inconsistencies in the correlation, in the areas where alarms are produced by the model-based fault detection systems. The development of the framework within which the correlation detector can be used as a fault detection system is left as an open research area.

5.2.9.3 Reverse Alarms

A reverse alarm is produced whenever a certain number of reverse pilot (r) and NNME estimation ($\hat{r}$) pairs have an absolute error that exceeds an error threshold. The check is done as follows: a routine determines if, in a window of 30 points, more than 26

---

16. Obviously, the correlation detector cannot detect other kinds of behavioural changes (i.e. sudden drops or drifts) which can be detected by the modeling engine-based system. Also, in contrast with the modeling engine-based system, the correlation detector cannot be used in real time.

of the data pairs have an absolute error that is larger than an alarm threshold. This check is done for all the 970 windows that can be generated by sliding the window of 30 points by one to cover the whole span of the 1,000 checked data points. If more than a predetermined number of windows (46) satisfy the above check, then a reverse alarm is produced.

The above method of calculating the error term which determines if a change in behaviour has occurred, was selected because both the reverse signal and the neural network estimation are very noisy signals. Due to this fact, a more straight-forward method of calculating an error term, (e.g. rms error) would result in high errors in noisy periods and cause false alarms.

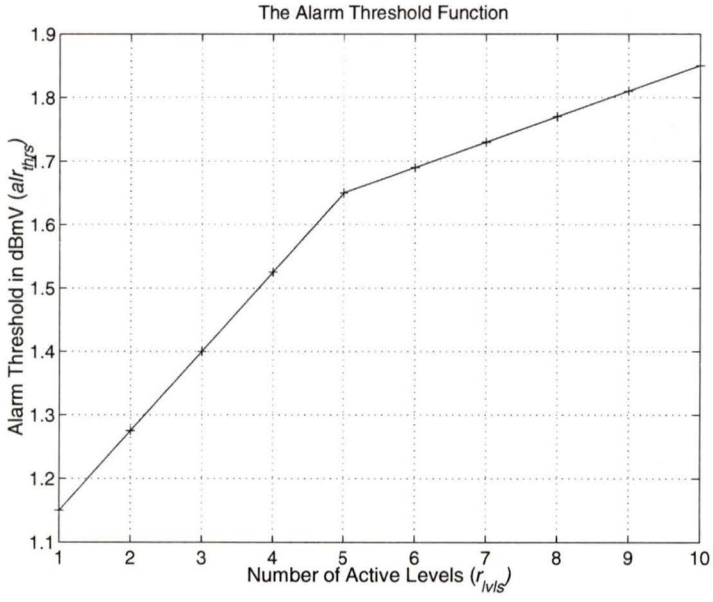

FIGURE 5-15.
The alarm threshold is a function of the number of "active" levels in the reverse signal.

The values of the sliding window length (30), the minimum number of data points that should exceed the alarm threshold (26), the minimum number of windows that should exceed the error to produce an alarm (46), as well as the alarm threshold (alr_{thrs}), were determined experimentally. These alarming parameters were calibrated by performing

several trials with data sets taken from amplifiers 117, 273 and 730, and were set so as to capture a set of "obvious" changes in behaviour, but also so as to avoid producing false alarms.[17]

The alarm threshold is set dynamically, and depends on the number of reverse levels that exist within the $RangeR$ of a set of 5,000 reverse data points. The alarm threshold is given by:

$$alr_{thrs} = \begin{cases} a \cdot r_{lvls} + b & if \quad r_{lvls} \leq 5 \\ a' \cdot r_{lvls} + b' & if \quad r_{lvls} > 5 \end{cases} \tag{5.7}$$

where $r_{lvls} = (R_h - R_l)/Q$ is the number of active levels; Q is the quantization step in dBmV, and R_h and R_l are the values of the maximum and minimum active level in dBmV.

$a = 0.133$, $b = 0.933$ and $a' = 0.04$, $b' = 1.4$.

The above function that gives the alarm thresholds is plotted in Figure 5-15.

The alarm threshold is set to 1.15 dBmV when one "active" levels exist in the subsequent data set (5,000 data points), and increases linearly to 1.65 dBmV for five levels. After that, the alarm threshold continues to increase linearly with a smaller slope, reaching a maximum value of 1.85 dBmV for ten active levels. The reason the alarm threshold was set to be a function of the number of active levels, resides in the fact that the modeling error of the NNME seems to increase slightly with the increase of the number of active levels.[18] As such, in order to avoid producing a large number of false alarms, the error tolerance between the NNME's output and the reverse signal is set to be larger when many active levels are present.

---

17. The optimization of the alarming parameters is left as an open research topic.

18. This observation seems to be related to the fact that the reverse pilot level is measured at the head-end and not at the amplifier. As such, the reverse pilot measurements are dependent on the temperatures along the cable span from the amplifier to the head-end, rather than only on the temperature of the amplifier's housing. This issue will be further discussed in the next Chapter.

5.2.9.4 Reverse Warnings

Warnings are issued by the fault detection system whenever the error conditions are met for a number of windows over the set under consideration, but the number of windows does not exceed the threshold number (46). This means that if an error condition is met for less than a predetermined time interval, a warning is generated and the reverse module is marked to have experienced an inconsistency of behaviour of short duration. After a warning is issued, no re-training is performed and the NNME continues the modeling. Often warnings precede reverse alarms, especially in the case of the occurrence of incipient faults. The location of warning flags are marked in figures as doted vertical lines (see Figure 5-17).

5.2.10 Neural Network Modeling Results

In this section results are presented which show the modeling and fault detection capabilities of the NNME. These results were obtained from a large scale experiment in which the neural network fault detection system was tested over a large set of reverse signal data. Specifically, data records were used from a randomly selected set of 50 amplifiers[19], each of which were 120,000 samples in length, corresponding to a time period of approximately three months[20]. For each of these data sets the neural network fault detection system was used in the same manner as described in the previous sections of this chapter.

Two typical cases of the modeling of the NNME are presented in this section while a detailed analysis of the results is presented in the next chapter. The first example shows a case where no alarms were generated, while the second example presents a case where three reverse alarms were produced.

The example modeling of the first case is illustrated in Figure 5-16, where the reverse signal from amplifier 730 and the model's output are plotted. The modeling engine initially used data from the first three days of the presented set to learn the reverse pilot - temperature dependency, and, as we can see, it has successfully modeled the reverse

---

19. All data used for the experiments in this thesis were taken from Rogers cable plant in Newmarket, Ontario. The data collection is done by Rogers.

20. The data corresponds to the period Aug 20 - Nov 15 1997.

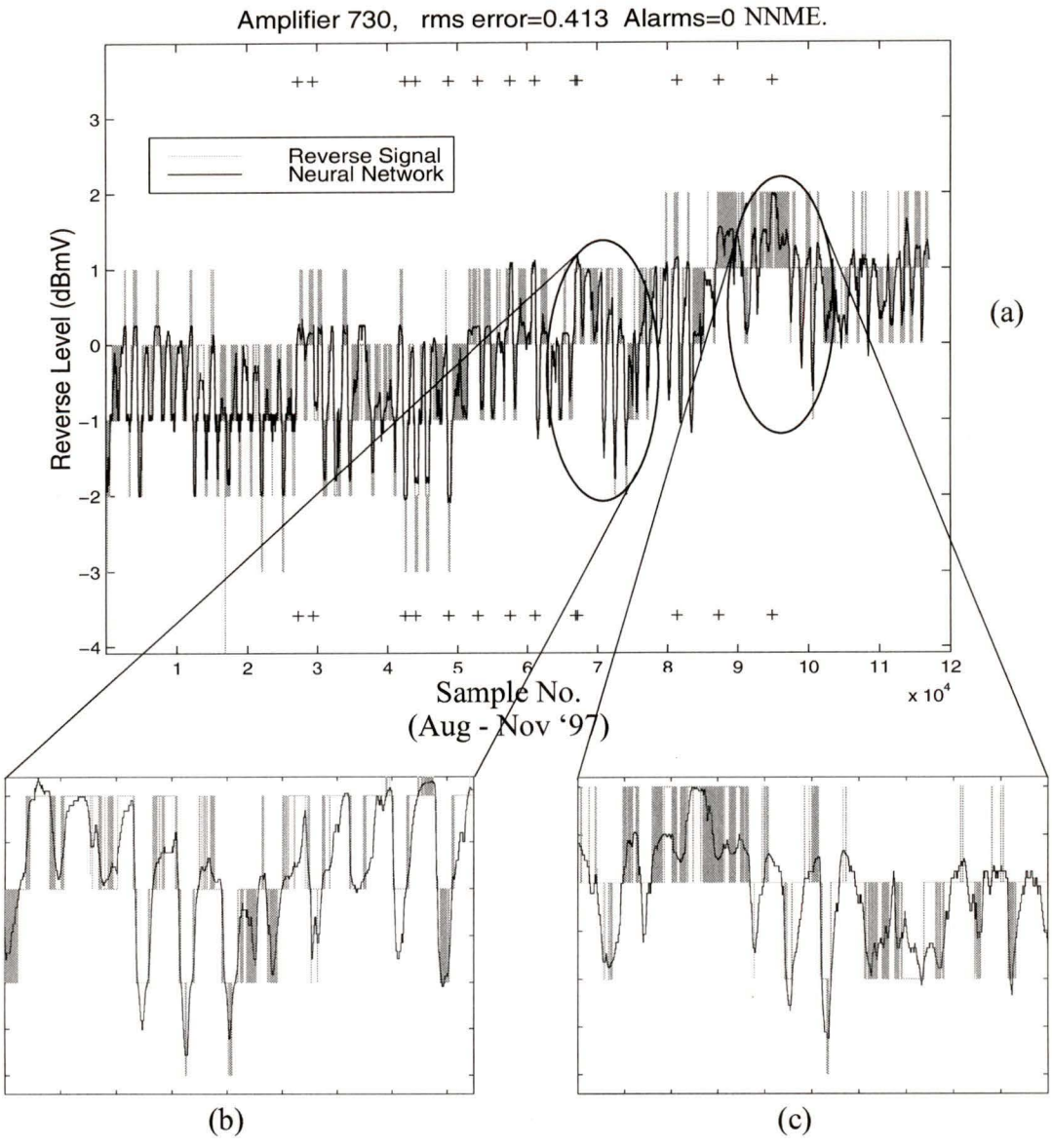

FIGURE 5-16.
Example of NNME modeling. The neural network fault detection system did not discover any significant changes in behaviour, and therefore no reverse alarms where produced. Subplots (b) and (c), show the magnification of the two regions circled in (a).

behaviour. During the modeling several temperature re-trainings occurred, but as the modeling engine tracked the reverse signal closely enough, no reverse alarms were generated. As such, the reverse pilot of amplifier 730 is marked as not exhibiting any significant changes in its behaviour during that period ("fault free" conditions).

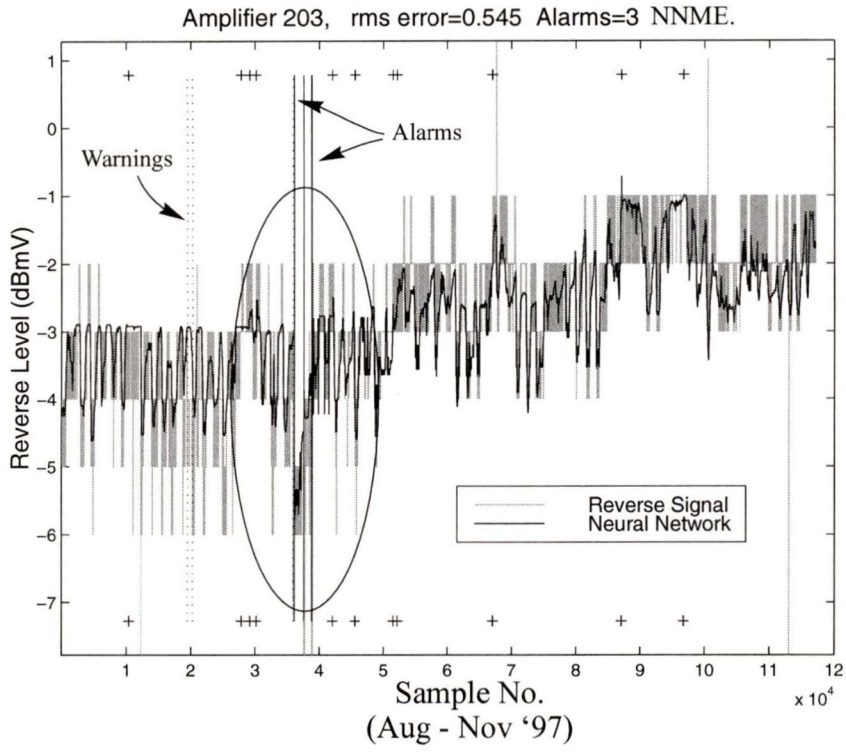

FIGURE 5-17.
An example of a reverse pilot modeling with warnings (doted vertical lines) and reverse alarms (dark solid vertical lines). Figure 5-18 shows a magnification of the area around the three alarms.

In contrast to amplifier 730, results from modeling amplifier 203 over the same period show that its reverse pilot signal experienced changes in its behaviour which caused the neural network fault detection system to generate three alarms. The results of the modeling are shown in Figure 5-17, with the locations of the three alarms marked by the dark vertical lines.

A magnification of the region where the three alarms occurred is plotted in Figure 5-18(a), while Figure 5-18(b) shows the temperature for the same region. Examining carefully that region, we can see the changes in the behaviour: near the point where the first alarm was generated, the reverse pilot suddenly drops to levels that are one to two levels lower than the levels normally associated with those temperatures. During that

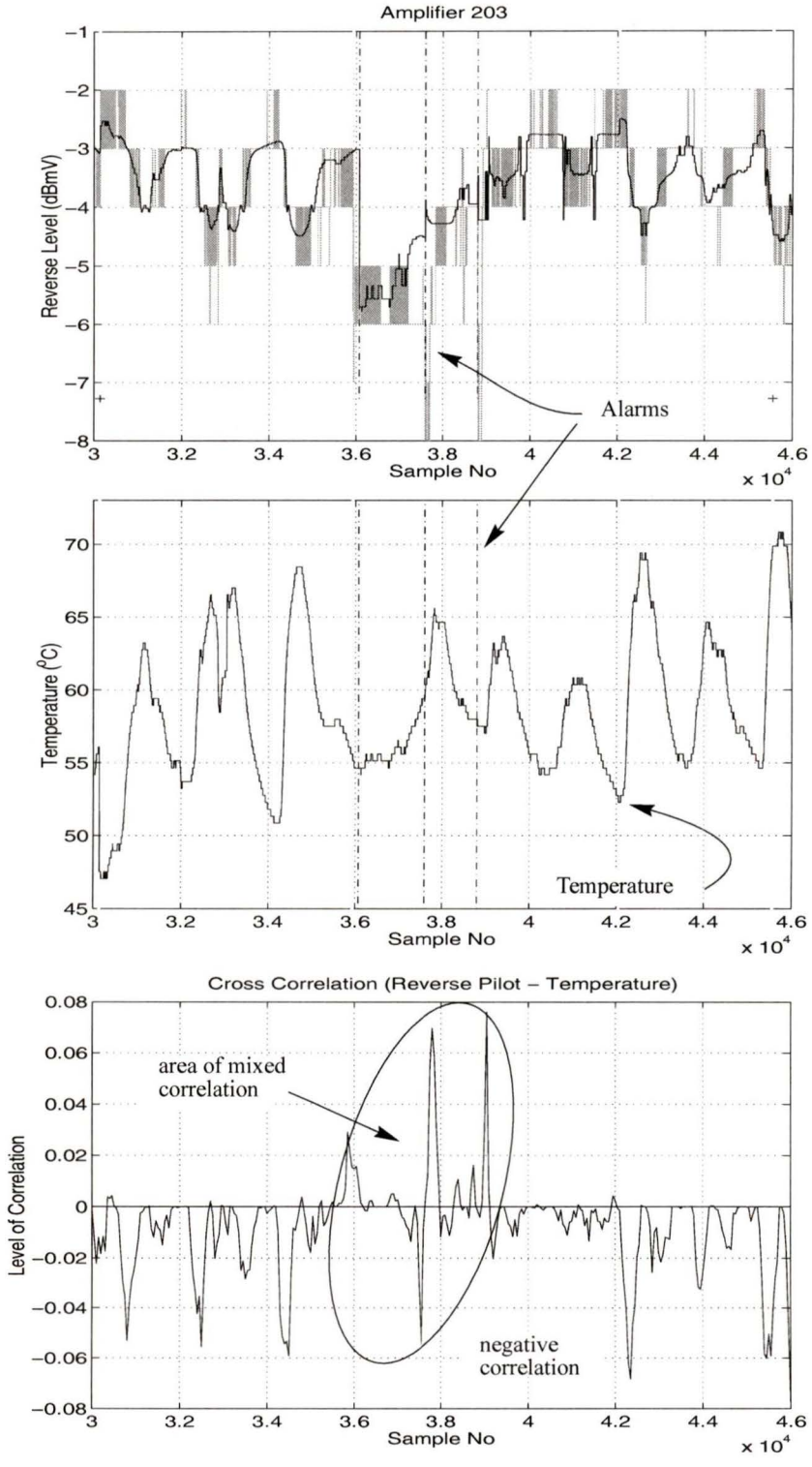

FIGURE 5-18.
Example of detection of behavioural changes. a) Reverse pilot signal (light trace) and NNME model (darker trace). The reverse alarms are denoted by the dashed vertical lines. b) The temperature signal and c) the cross correlation of the temperature and reverse signal.

period the temperature is also decreasing, which means that the reverse pilot becomes positively correlated to the temperature[21]. This fact can be confirmed by calculating the cross correlation between the reverse signal and the temperature.

In Figure 5-18(c), the cross correlation for that region is plotted, which shows that indeed the correlation is positive in the area of the first alarm and mixed (negative and positive) during the whole "faulty area" between the first and third alarm. Because of the inconsistent behaviour, the neural network modeling engine was not able to model during that period, which caused the second alarm. It appears that the third alarm denotes the last change in behaviour and a return to the normal negatively correlated reverse - temperature relationship.

The results of the large scale experiment are shown in Table 5.2. For each of the amplifiers that were used in the experiment, the rms error between the neural network model and the actual reverse signal is shown, as well as the total number of reverse alarms.

Table 5.2: Results from the Neural Network Modeling Engine.

Amplifier	**Rms error**	Number of alarms	**Amplifier**	**Rms error**	Number of alarms
102	0.24	0	**394**	0.56	0
110	0.23	0	**427**	0.68	5
117	0.59	2	**433**	0.62	4
124	0.6	0	**443**	0.79	6
138	0.39	0	**446**	0.89	8
141	0.45	0	**447**	0.86	9
203	0.54	3	**511**	0.35	0
206	0.50	2	**514**	0.37	0
207	0.50	0	**518**	0.44	0
223	0.59	3	**519**	0.40	0
232	0.50	0	**531**	0.35	0
236	0.56	0	**631**	0.75	1
240	0.53	0	**632**	0.67	0
241	0.60	3	**633**	0.66	4
246	0.62	0	**641**	0.58	4

21. As noted in previous chapters, the reverse and temperature signals are normally negatively correlated.

Table 5.2: Results from the Neural Network Modeling Engine.

Amplifier	Rms error	Number of alarms	Amplifier	Rms error	Number of alarms
273	0.62	1	**642**	0.60	4
278	0.60	1	**671**	0.86	8
283	0.61	3	**711**	0.26	0
292	0.54	0	**730**	0.41	0
296	0.81	6	**801**	0.34	0
300	0.80	4	**810**	0.47	1
313	0.46	0	**821**	0.36	0
321	0.42	0	**894**	0.45	0
353	0.52	0			
354	0.48	0			
361	0.43	0			
393	0.53	0			

From Table 5.5 we can see that, generally, modelings that have no alarms have a low rms error, while modelings with relatively many alarms have a higher rms error. This fact confirms the reliability of the fault detection system, and will be discussed further in the next chapter.

5.3 Statistical Modeling Engine (SME)

This section presents a second modeling engine also able to model the reverse pilot. The statistical modeling engine is able to model the reverse pilot based on the fact that it can capture the reverse pilot temperature dependency, by statistically processing a relatively small set of reverse - temperature data pairs. The statistical processing is done in a manner similar to the one presented in Section 4.6.2. This time, however, instead of finding the single reverse level that occurs most often for each temperature, the three[22] most frequently occurring levels are used. These levels are used to produce the average reverse

[22] Experiments show that generally *98-100%* of the reverse signal that corresponds to a temperature-bin of $1^{o}C$ is contained within three quantization levels. The possible presence of more than three levels is attributed mainly to noise. As such, these noise related levels are filtered out of the training set.

level associated with the given temperature. By doing this, a "look-up" table is generated, which contains all the temperatures that appear in the training[23] set, rounded to an accuracy of one degree ($1\ ^{o}C$), and a reverse level associated with each temperature.

Table 5.3: Amplifier 141. The table below contains the reverse level and its percentage of occurrence for each temperature (rounded to 1 ^{o}C accuracy).

Temperature (^{o}C)	1st level (dBmV)	%	2nd level (dBmV)	%	3rd level (dBmV)	%
52	1	69.8	2	30.2	-	-
53	1	81.5	2	18.6	-	-
54	1	81.1	2	18.9	-	-
55	1	93.7	2	6.3	-	-
56	1	100	-	-	-	-
57	1	98.0	0	2.0	-	-
58	1	73.7	0	26.3	-	-
59	1	72.5	0	27.5	-	-
60	1	65.7	0	34.3	-	-
61	0	65.1	1	34.9	-	-
62	1	51.2	0	48.8	-	-
63	0	60.0	1	40.0	-	-
64	0	88.4	1	11.6	-	-
65	0	90.0	1	10.0	-	-
66	0	89.4	1	10.6	-	-
67	0	68.4	-1	24.8	1	6.8
68	0	81.7	-1	18.3	-	-
69	0	54.0	-1	46.0	-	-
70	-1	51.4	0	48.6	-	-
71	0	71.6	-1	28.4	-	-

As an example, the results from the statistical analysis of a small set taken from amplifier 141 are presented in Table 5.3. The first column of the table contains the rounded temperatures of the training set; the second, fourth, and sixth column contain respectively the reverse level that occurred the most, the second most and third most frequently for each temperature, while the percentages of their occurrence are shown in the third, fifth and seventh column.

---

23. The set referred to as "training set" is basically the "look-up table generation set". We refer to it as training set using the same notation used for the NN modeling engine.

Using the results from the statistical analysis, it is easy to calculate the average of the three levels and associate a single (average) reverse level for each temperature. This average level appears in the generated look-up table shown in Table 5.4.

Table 5.4: The look-up table generated from the first 4,000 data points (3 days) for amplifier 141.

Temperature (°C)	Average R-level		Temperature (°C)	Average R-level
52	1.30		62	0.51
53	1.18		63	0.40
54	1.18		64	0.11
55	1.06		65	0.10
56	1.00		66	0.10
57	0.98		67	-0.18
58	0.73		68	-0.18
59	0.72		69	-0.46
60	0.65		70	-0.51
61	0.34		71	-0.28

In Figure 5-19 we can see the reverse - temperature map (light trace) as well as the dependency map (dark trace) that results from the above statistical analysis. Table 5.3. was generated by using approximately 3 days of data (4,000 data points). The SME can use the look-up table to model the reverse pilot: for each temperature that is presented to the modeling engine's input, a reverse level is produced as an output; the output level is the one that corresponds to the temperature present in the table that is closest to the input temperature. Obviously, and in contrast to the case of the NNME, the output set of the SME is not continuous. The results from the modeling of amplifier 141, as well as the results from a large experiment using the statistical model are presented in the next section.

5.3.1 Statistical Modeling Results

In order to evaluate the modeling and fault detection capabilities of the SME fault detection system, a large scale experiment was performed. This experiment was identical to the experiment of the NNME: the same 50 amplifiers through the same time period

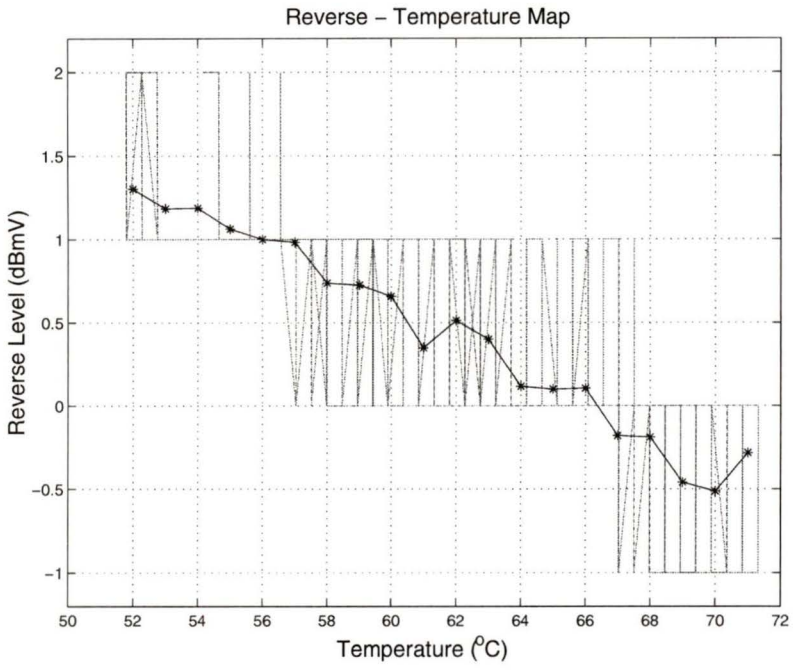

FIGURE 5-19.
The reverse pilot - temperature map (light trace) and the dependency function created by the look-up table (dark trace).

were modeled whith the statistical modeling engine, implementing the same heuristics for training, re-training and alarming. Some typical cases of this experiment are presented in this section and illustrated in the following figures.

First, the results of modeling amplifier 141 are presented. The initial look-up table for this modeling is the one shown in Table 5.4. Using this table the statistical modeling engine was able to model the reverse signal for 13 consecutive days without any re-trainings. After the 13th day, the SME performed several temperature re-trainings[24], during which the unseen temperatures were included in the initial training set, and the statistical analysis was repeated to generate "new" look-up tables. The results of the modeling are shown in Figure 5-20.

---

[24]. In the statistical modeling approach, we use the term "re-training" to refer to the generation of the new and updated look-up table.

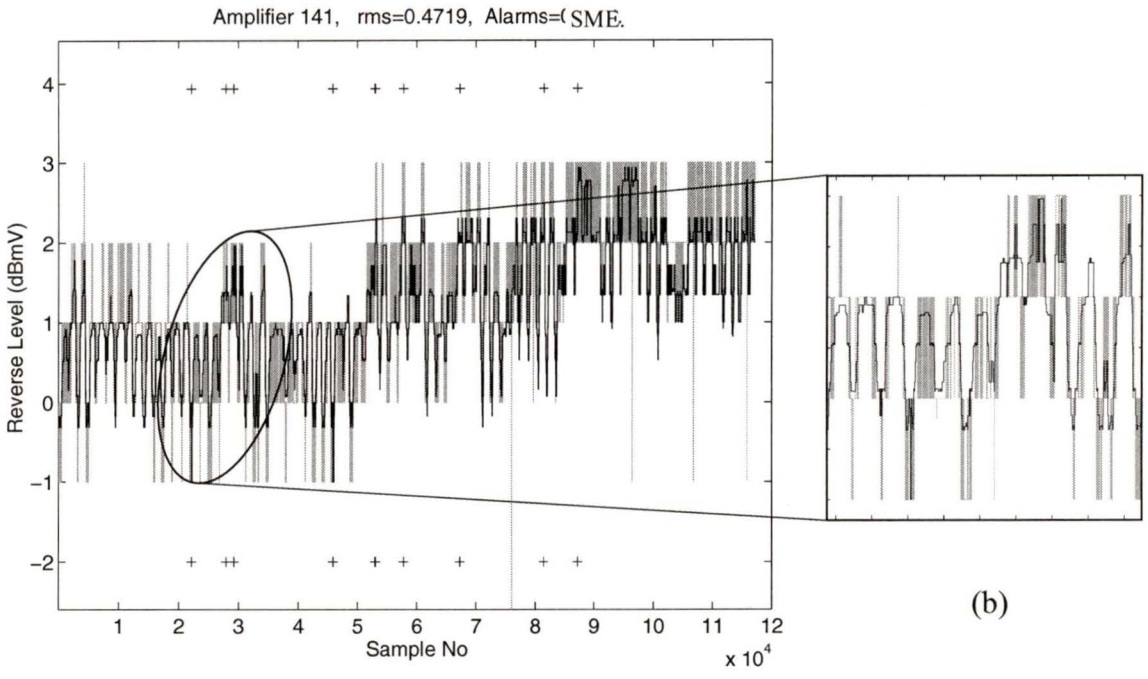

FIGURE 5-20.
An example of the modeling of amplifier 141 by the statistical modeling engine. A magnification of the circled region is shown in (b).

The statistical model has tracked the reverse signal fairly well, as can be seen in the magnification of Figure 5-20(b). The rms error (0.471) of this modeling is very close to that produce by the NNME (0.454), and with both modeling methods no alarms were produced.

In Figure 5-21, the modeling of amplifier 730 is also shown, with a magnification of the two regions shown also in the neural network approach. Observe that the models are very close, and that neither method produced any alarms. A more detailed comparison of the modeling results of the two engines is presented in the next chapter.

In Figure 5-22 the modeling results from amplifier 278 are illustrated. For this amplifier the SME fault detection system discovered a change in the behaviour of the reverse signal on the 30th of Sept. 1997, and consequently produced a reverse alarm. It has also generated three warnings which are not in the neighborhood of the fault alarm. A more

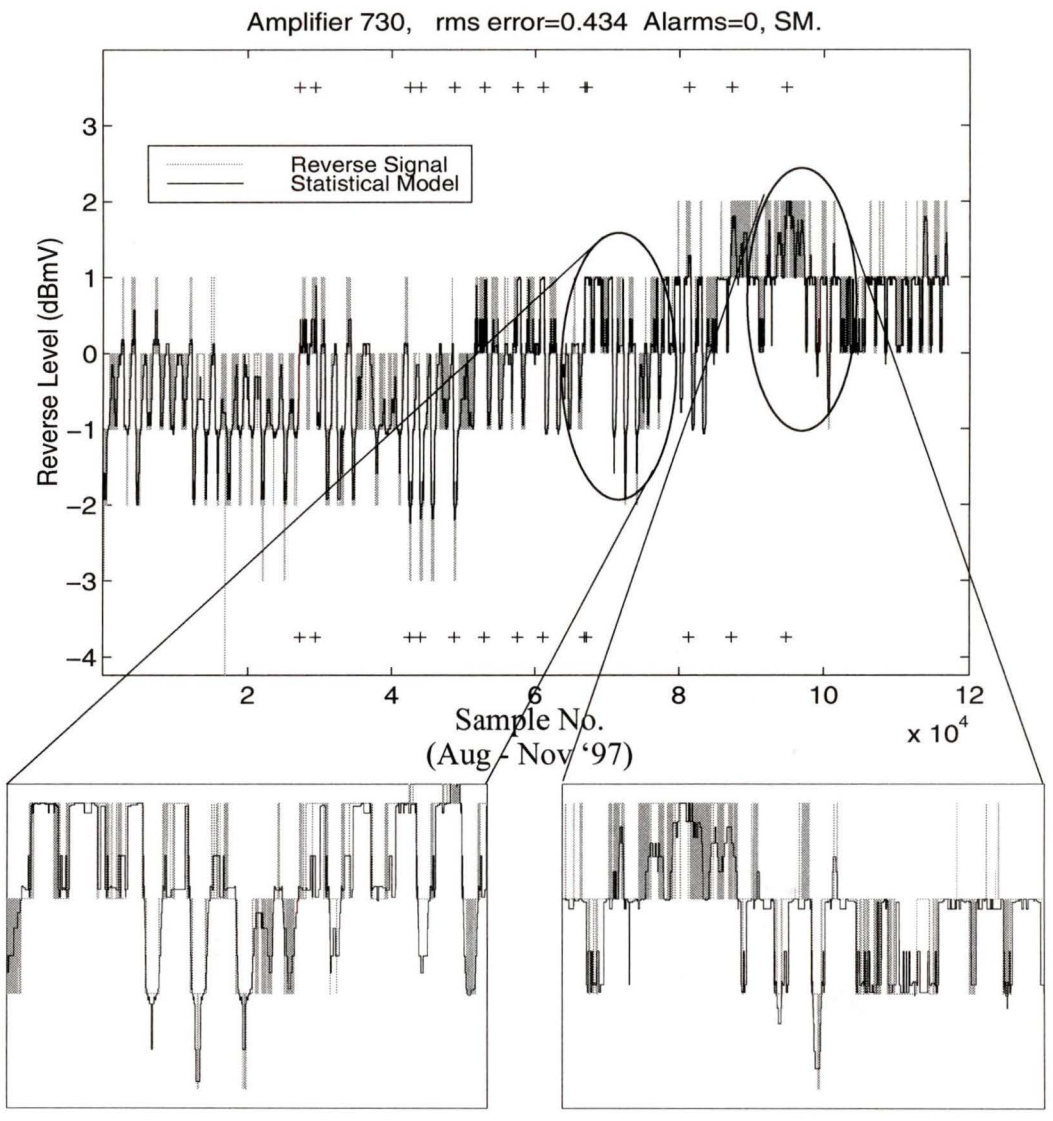

FIGURE 5-21.
(a) Modeling of the reverse pilot of amplifier 730 using the Statistical Modeling method. Also subplots (b) and (c) show a magnification of the same areas shown in Figure 5-16.

detailed analysis of the areas around the warnings indicate that these warnings are most likely not related to changes in behaviour, but rather to the effects caused by the large quantization step of the reverse signal.

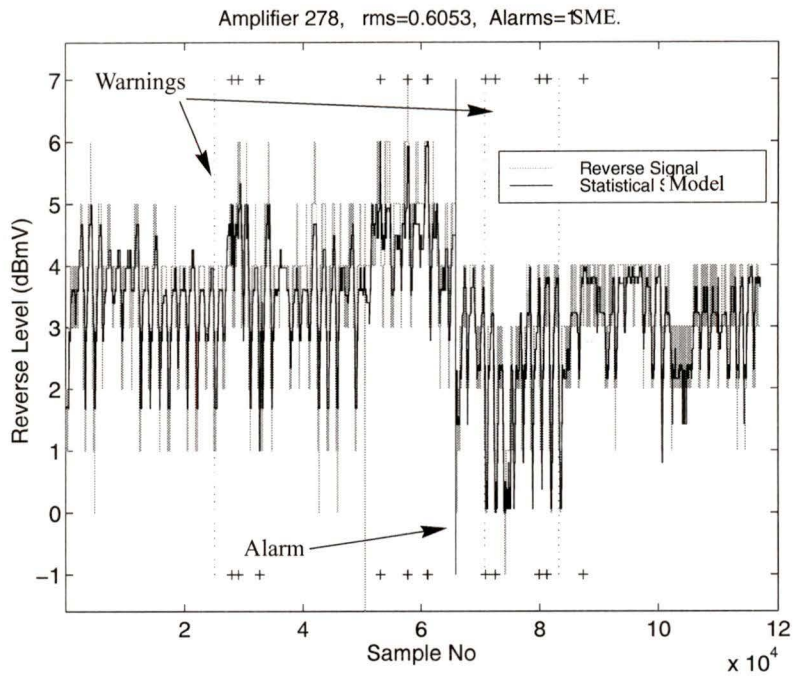

FIGURE 5-22.
The modeling of amplifier 278. The SME fault detection system discovers one change in behaviour marked by the dark solid vertical line. The doted lines indicate the position of warnings.

A magnification of the area around the alarm is shown in Figure 5-23. The reverse signal and the modeling engine's output as well as the temperature, can be seen in the subplots (a) and (b) respectively, while in subplot (c), the temperature-reverse level distributions are illustrated. These two distributions correspond to the different behaviours of the reverse pilot before and after the reverse alarm. The change in behaviour is revealed by the obvious drop in the reverse level (Figure 5-23c).

Another case where the statistical modeling fault detection method was able to detect changes in behaviour is illustrated in Figure 5-24. In this figure, the statistical modeling of amplifier 203 is shown. Notice the similarity of the modeling with that of the NN modeling shown in Figure 5-17. The two systems discover the same changes in behaviour.

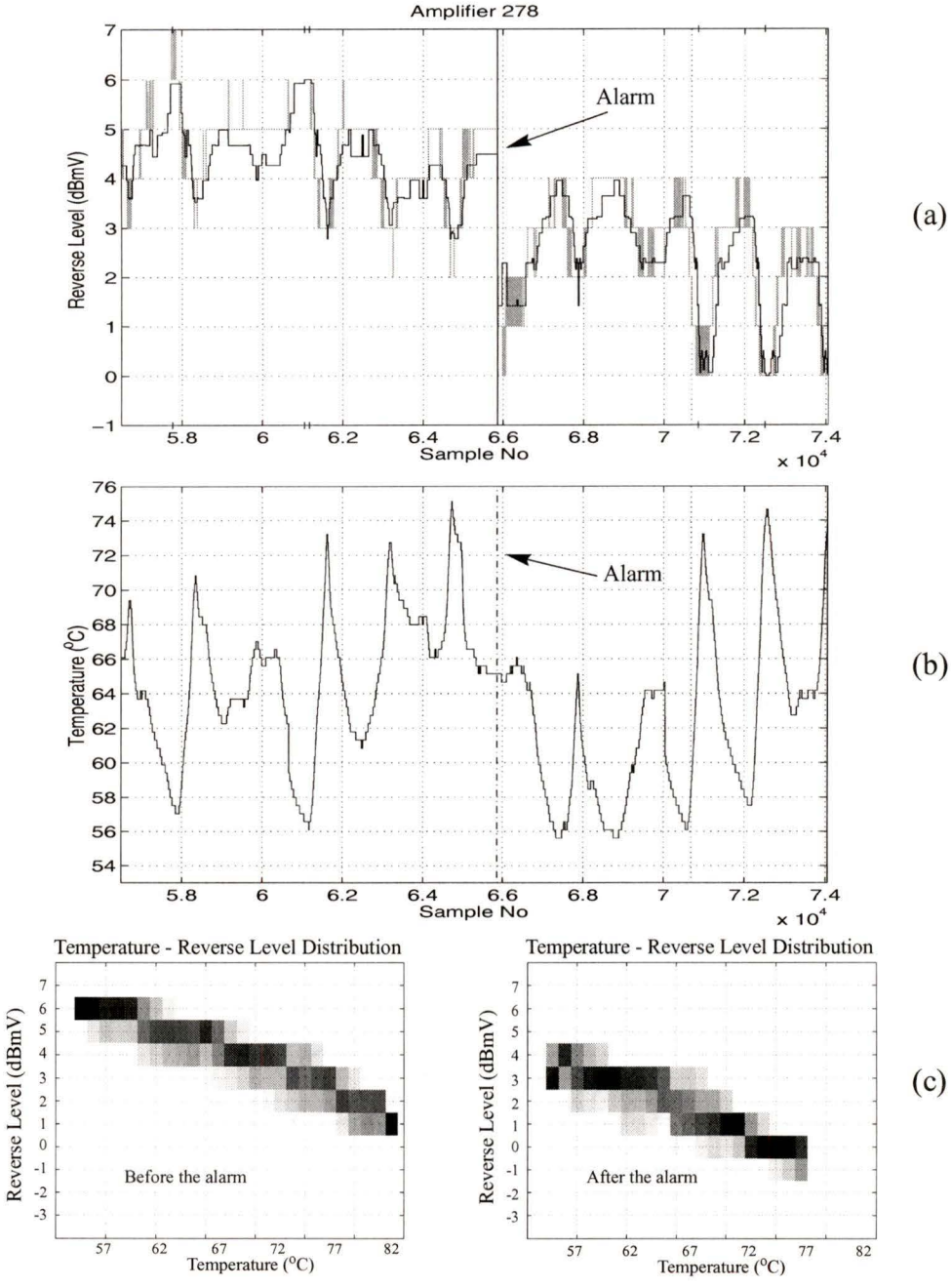

FIGURE 5-23.
a)The reverse Pilot and model, (b) the corresponding Temperature, (c) the two behaviours

The results of the large scale experiment using the statistical model are shown in Table 5.5. As in Table 5.2, for each amplifier, the rms error between the reverse signal and the model's output is shown in the second and fifth column, while the number of alarms for

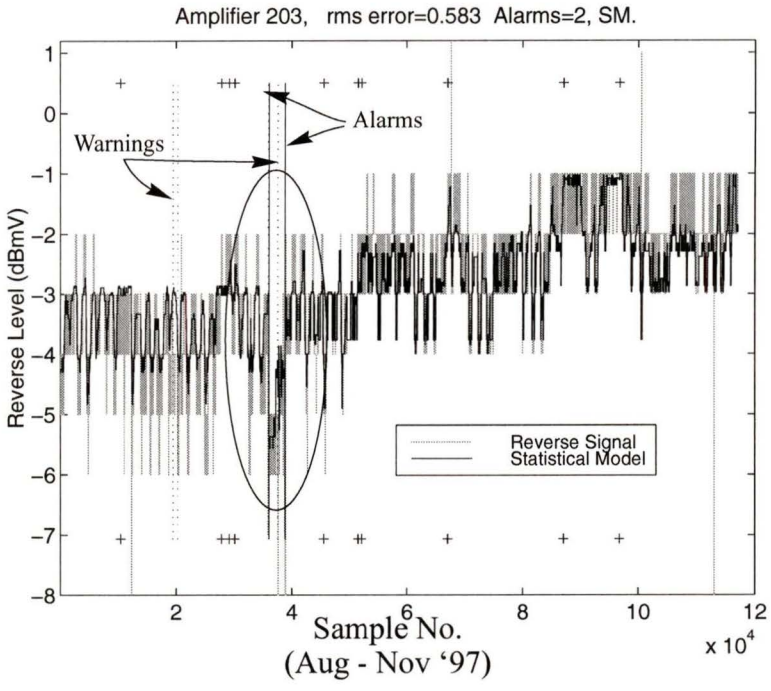

FIGURE 5-24.
The modeling of amplifier 203 using the statistical modeling method.

each amplifier is contained in the third and sixth column.

Table 5.5: Results from the Statistical Model fault detection system.

Amplifier	**Rms error**	Number of alarms	**Amplifier**	**Rms error**	Number of alarms
102	0.26	0	**394**	0.62	0
110	0.23	0	**427**	0.66	4
117	0.63	2	**433**	0.62	4
124	0.60	0	**443**	0.80	6
138	0.40	0	**446**	0.79	9
141	0.47	0	**447**	0.88	10
203	0.58	2	**511**	0.35	0
206	0.50	3	**514**	0.37	0
207	0.52	0	**518**	0.45	0
223	0.59	4	**519**	0.41	0
232	0.50	1	**531**	0.36	0
236	0.59	1	**631**	0.56	4
240	0.54	0	**632**	0.61	1
241	0.60	2	**633**	0.67	4
246	0.64	0	**641**	0.59	4
273	0.62	3	**642**	0.60	4
278	0.60	1	**671**	0.88	7

Table 5.5: Results from the Statistical Model fault detection system.

Amplifier	Rms error	Number of alarms	Amplifier	Rms error	Number of alarms
283	0.61	4	**711**	0.27	0
292	0.56	0	**730**	0.43	0
296	0.83	7	**801**	0.35	0
300	0.80	5	**810**	0.45	1
313	0.46	0	**821**	0.38	0
321	0.42	0	**894**	0.45	0
353	0.48	1			
354	0.46	1			
361	0.45	0			
393	0.52	0			

The results of the large scale experiments from both the SME-based and NNME-based fault detection systems are analyzed in more detail in the following chapter.

5.4 Summary

A basic requirement for implementing model-based fault detection for the reverse pilot of cable television networks, is to create a model of the reverse signal. Based on the observation that the reverse pilot level varies with the temperature, a model of the dependency of the reverse pilot on the temperature was created.

Two different modeling engines which are capable of capturing the reverse - temperature dependency were developed. The first modeling engine is based on the use of neural networks (Neural Network Modeling Engine). The neural network used is a single input-output feed forward NN, with 2 hidden layers and 3 neurons in each of its hidden layers. The second modeling engine (Statistical Modeling Engine), is based on using statistical analysis techniques to process the reverse and temperature signals and "extract" their dependency.

The results from a large scale experiment show that both modeling engines are capable of "capturing" the dependency by using information from a small period of time (2.5 days worth of observations), and then modeling the reverse pilot for an extended period of time (months). In addition, the developed fault detection system, using either model, is able to detect changes in the behaviour of the reverse signal.

Chapter 6:

Result Analysis and Discussion

6.0 Introduction

In this section the modeling capabilities of the two modeling engines (NNME and SME) are compared. The comparison is based on the results of the large scale experiments presented in Chapter 5. Additionally, the fault detection properties of the two modeling engine-based fault detection systems are discussed and further compared to those of the limit checking system currently used in Rogers cable plants. Finally, an approach which seems to increase the modeling accuracy of the reverse pilot is presented. This approach is based on modeling the reverse pilot by using an alternative measure of temperature, different from that of the enclosure of the amplifier.

6.1 Comparison of the Two Modeling Engine-Based Fault Detection Systems

In the following sections a comparison of the two modeling engine-based fault detection systems (NNME-based and SME-based systems) is presented. The comparison is done on the basis of their modeling accuracy, computational complexity, and fault detection properties.

The modeling capabilities of the two systems are compared using appropriate error norms which reflect the accuracy of the reverse pilot modeling. Their fault detection properties are compared on the basis of the number of alarms produced and their relative temporal location.

6.1.1 Modeling Capability

The accurate modeling of the reverse signal is critical for the development of a reliable model-based fault detection system. A model that is not capable of tracking the reverse pilot closely will be prone to producing false alarms. In order to avoid the false alarms under these circumstances larger error thresholds must be used. This will mitigate the effects of the poor modeling, but have the consequent result of a fault detection system

with decreased sensitivity[1]. On the other hand, if the modeling is accurate, the fault detection sensitivity can be increased by using tight alarm thresholds. This will also increase the reliability of the system, since it will minimize the probability of false alarms without sacrificing any fault detection sensitivity.

In this section, the modeling capabilities of the two modeling engines are compared. The comparison is done by using only those modelings of the large scale experiment that produced no alarms. To evaluate the two modeling engines we use the values of the $\|\mathcal{A}\|_\infty$ and $\|\mathcal{A}\|_2$ norms, of each modeling[2]. The values of these norms are shown in Table 6.1, with their averages contained in the last row.

Table 6.1: For the amplifiers that did not produce any alarms, the $\|\mathcal{A}\|_\infty$ *and* $\|\mathcal{A}\|_2$ norms, and their average values for both models.

Amplifier	Neural Network Modeling Engine		Statistical Modeling Engine	
	$\|\mathcal{A}\|_2$	$\|\mathcal{A}\|_\infty$	$\|\mathcal{A}\|_2$	$\|\mathcal{A}\|_\infty$
102	0.25	1.34	0.26	1.00
110	0.24	7.03	0.24	7.00
138	0.60	6.49	0.40	6.74
141	0.39	6.42	0.47	5.34
207	0.45	5.16	0.53	3.21
240	0.51	3.21	0.54	2.99
246	0.53	2.83	0.64	9.35
278	0.61	5.84	0.61	5.97
292	0.55	7.02	0.57	7.02
313	0.46	3.05	0.47	3.02
361	0.44	5.75	0.46	5.45
393	0.54	7.92	0.53	7.99
511	0.35	5.64	0.35	5.75
514	0.37	5.91	0.38	5.92
518	0.45	5.71	0.45	5.56
519	0.40	5.98	0.42	6.00

1. A fault detection system with low sensitivity may result in undetected faults, an effect as undesirable as the generation of numerous false alarms (large false-alarm rate).

2. The $\|\mathcal{A}\|_2$ (or else rms error), and $\|\mathcal{A}\|_\infty$ are given by:

$\|\mathcal{A}\|_2 = \sum_N \frac{(r-\hat{r})^2}{N}$, $\|\mathcal{A}\|_\infty = max(|r-\hat{r}|)$, where r is the reverse pilot and $\hat{r}$ is the model.

Table 6.1: For the amplifiers that did not produce any alarms, the $\|f\|_\infty$ *and* $\|f\|_2$ norms, and their average values for both models.

Amplifier	Neural Network Modeling Engine		Statistical Modeling Engine	
	$\|f\|_2$	$\|f\|_\infty$	$\|f\|_2$	$\|f\|_\infty$
531	0.36	6.33	0.36	6.38
711	0.27	5.64	0.27	5.74
730	0.41	6.02	0.43	6.05
801	0.35	6.86	0.35	6.92
821	0.37	6.02	0.38	6.00
894	0.46	6.91	0.46	7.00
Average Values:	0.43	5.75	0.44	5.78

By examining the results, we can see that both models track the amplifiers' reverse pilots well and that the two models are very close to each other. As an example, the modeling of amplifier 730 is presented in Figure 6-1. From this figure it can be seen how close the two models are, making it hard to distinguish the two traces from each other.

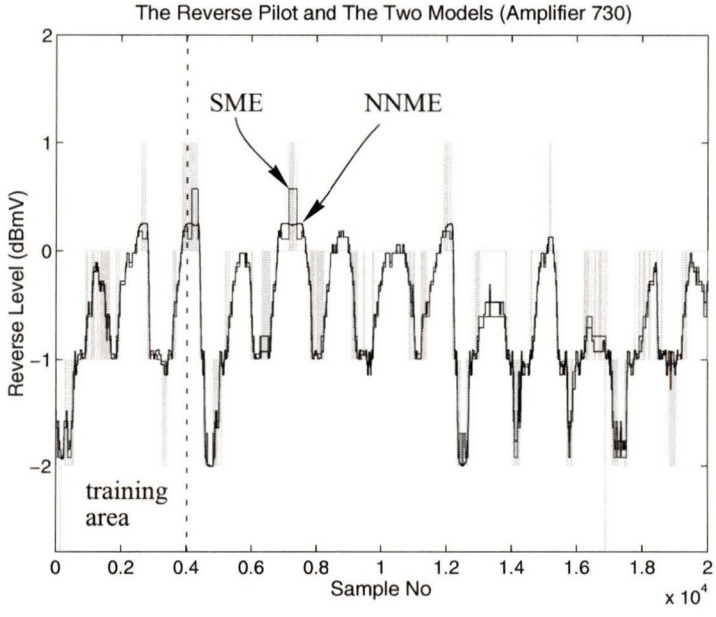

FIGURE 6-1.
An example of the modeling of the two modeling engines on a section of amplifier 730. The light gray trace is the reverse pilot, and the two models are shown in darker trace. The models are so close that it is difficult to distinguish the two traces.

Another way we can validate the accuracy and proximity of the two models, is by plotting the *T-R* dependency map of the training set, as well as the maps produced by the two modeling engines. These maps are illustrated in Figure 6-2.

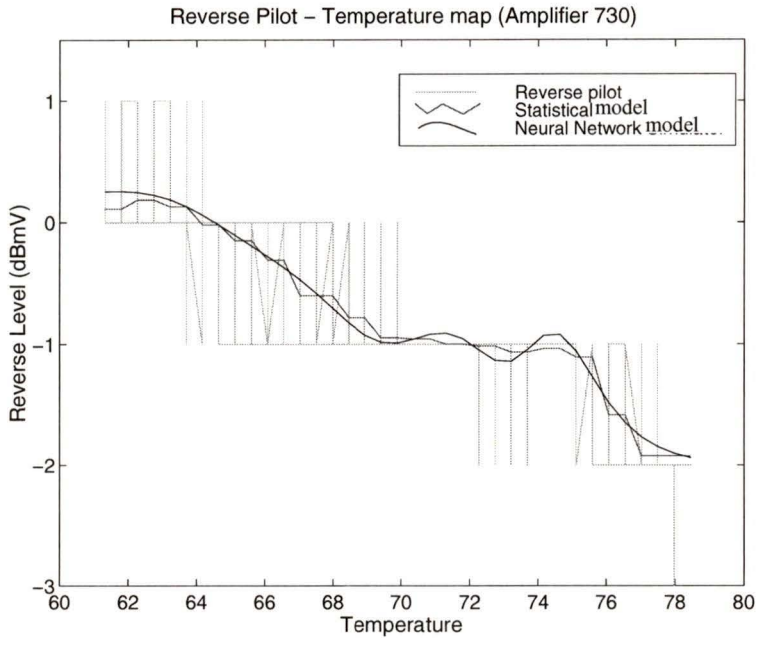

FIGURE 6-2.
For the first 4,000 samples of amplifier 730 (initial training set) shown in Figure 6-1, the reverse - temperature map is plotted (gray trace). Also shown in darker traces, are the statistical and neural network models.

In Figure 6-3 the $\|f\|_2$ norm is plotted versus the $\|f\|_\infty$ norm. Observe how close the error pairs fall to each other. By comparing the norms of the two modelings for each amplifier, as well as their average values, we can conclude that the neural network models the reverse pilot slightly better (produces lower error-norms) than the statistical modeling engine.

6.1.2 Computational Cost

Although the neural network generally models the reverse pilot better than the statistical model, there is a large computational cost involved in using a neural network. For both engines, the most computationally intensive parts of the modeling are the trainings, with

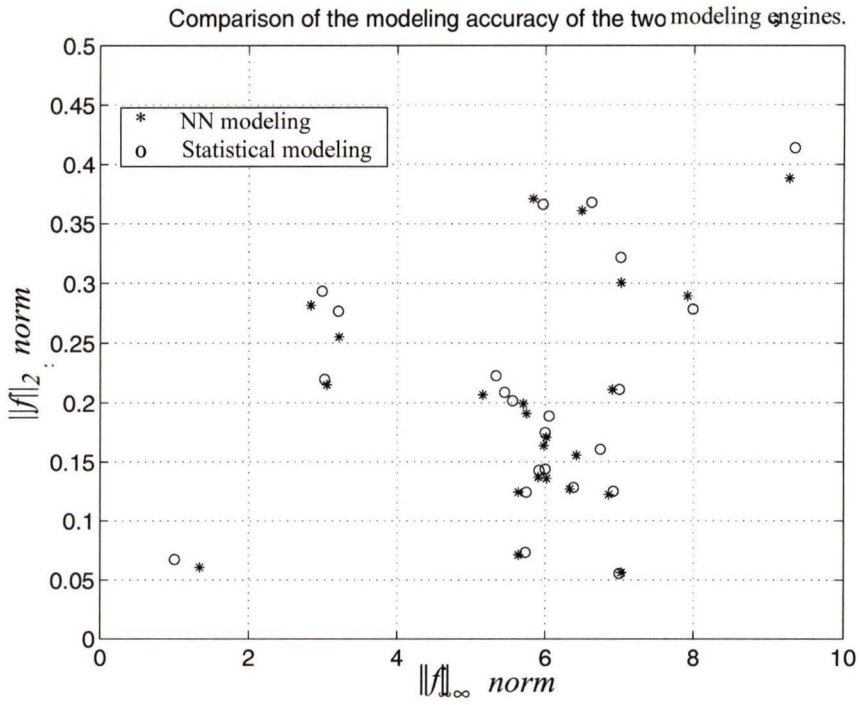

FIGURE 6-3.
For the amplifiers of Table 6.1, this figure plots the $\|f\|_\infty$ *norm* versus the $\|f\|_2$ *norm* for the two modeling engines. Observe that most of the neural network errors (stars) tend to be closer to the bottom-left corner than the statistical engine errors (circles).

the neural network retrainings, on the average[3], being 9×10^3 times more "expensive" than the statistical engine's re-trainings. This fact, along with the relative simplicity of the SME, constitutes the main advantages of the statistical modeling approach over the neural network modeling.

However, the computational complexity of the NNME, does not impose any limitations to its application in real-time, since the data acquisition process (one data record per min) is significantly slower than the modeling process. The issues related to the real-time application of the fault detection systems are further discussed in Section 6.5, Chapter 7, and Appendix A.

---

3. The computational cost of a training is not always the same, since temperature re-trainings use training sets of various lengths (recall that the training sets are enlarged with "unseen" temperatures). As such, the computational cost was calculated as an average over all re-trainings (temperature and behavioural).

6.2 Fault detection Properties

In this section, the fault detection properties of the developed systems are discussed. As noted in Chapter 5, the underlying rules for detecting changes in behaviour of the reverse pilot for both the NNME and SME based fault detection systems are essentially the same. This means that any difference in the outcomes of the two systems is a result of the difference in the modeling. In this section, the clustering of the experimental results is presented, while in Section 6.2.1 the location of the alarms generated by the two systems are compared. Finally, in Section 6.3, the fault detection capabilities of the modeling engine-based system and the limit checking system are compared.

In Figure 6-4 the rms error versus the number of alarms for each amplifier and for both models is plotted. From these plots it is easy to visualize the clustering of the results, which is very similar for both systems. Cluster (A) contains the amplifiers which did not cause any alarms and have relatively low rms error, cluster (C) comprises amplifiers which have produced relatively many alarms and have high rms errors, while cluster (B) is formed by amplifiers with a number of alarms and rms errors which lies in between (A) and (C).

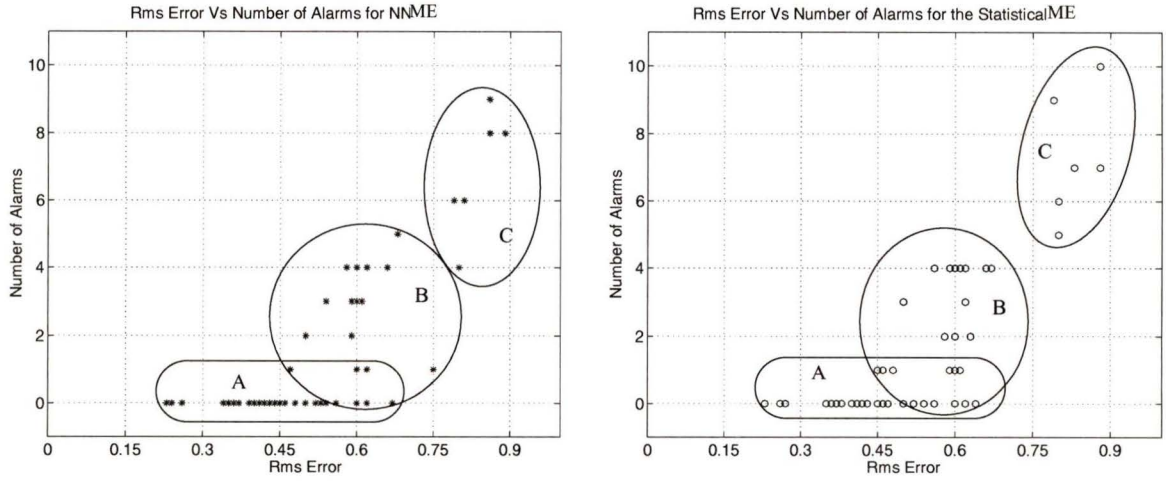

FIGURE 6-4.
The rms error versus the number of alarms for both models and for all fifty amplifiers. Notice the clustering of the results: cluster (A) -low rms and no alarms; cluster (C) -high rms and many alarms; cluster (B) in-between.

The results from both models show that most of the amplifiers which did not cause any alarms have relatively low rms error, while amplifiers which caused many alarms have generally a higher rms error. This fact confirms to some extent the reliability of the fault detection system: if a reverse signal has a consistent behaviour, it can be accurately modeled (low rms error); on the other hand, reverse pilots that exhibit many changes in their behaviour (inconsistent behaviour) cannot be modeled as closely (high rms error).

The results also indicate that the error bounds are neither excessively wide nor excessively tight[4]. If the bounds were excessively tight, there would be a large number of alarms (false) with a corresponding rms error that would be relatively low- a result of the good modeling of an amplifier which does not exhibit significant inconsistency in its behaviour. In contrast, if the bounds were excessively wide, then we would have a significant number of amplifiers with no alarms and a high rms error. It should be noted that the width of the bounds is not optimized, but rather set experimentally.

6.2.1 Alarms

In this section we will discuss and compare the number of alarms produced by the two model-based systems as well as their relative location. Although a direct comparison of the two modeling engines is not our goal, by comparing the alarms, and in particular the location of their occurrence, we can gain important information that could confirm or contradict the reliability of the fault detection system. If, for example, the two models produce a different number of alarms and at different locations (different events) for the majority of the tested amplifiers, then we could easily assume that the alarming events are subjective and hence might not correspond to real changes in behaviour. On the other hand, if the number of alarms produced by the two systems is comparable, with the corresponding alarms being temporally close, it would mean that both systems discover real changes in behaviour.

---

4. The bounds are not set optimally.The optimization of the bound width is beyond the scope of this thesis. A detailed analysis of a methodology for optimizing error bounds can be found in [31].

For the set of the fifty amplifiers, the two systems produced a comparable number of alarms, with the SME based fault detection system producing a total of 94 alarms, and the NNME based system producing in total 82 alarms. For each amplifier the time interval between the corresponding alarms was calculated[5]. Ideally, if the modeling of the two engines were identical, the two systems would produce alarms at exactly the same instants of time, and their calculated relative time interval would be zero. But as this is not the case[6], the corresponding alarms do not occur always at the same time. Instead, they may be close or even infinitely far[7].

Table 6.2: The number and percentage of alarm pairs with various distances.

Time interval between corresponding alarms	Number of Alarms	Percentage of occurrence
0	36	45%
0-2 hours	16	19%
2 hours - 1/2 days	6	8%
1/2- 3 days	16	19%
3-5 days	7	9%
0-5 days	81	100%

In Figure 6-5 a histogram of the time intervals between the corresponding alarms and their percentage of occurrence is shown. Their distance is marked in days in plot (a), and in hours in plot (b). For a total of 81 alarm pairs, 74 of them have less than 3 days distance, with the majority of these having a distance less than half a day (58). It is worth noting that from these 58 alarm pairs, 52 are less than 2 hours apart, while 36 occur exactly at the same time.

In terms of percentages we can say that 64% of the alarms occur with a distance of less than 2 hours, 72% occur within a distance of half a day, and 91% with a distance less than 3 days. Table 6.2 contains the number of alarms that occur in various distance intervals, as well as their percentage of occurrence, which is plotted in Figure 6-5(a).

---

5. The distances between each alarm of one system and all alarms of the alternate systems are calculated; the alarm pairs that have minimum distance constitute a "corresponding alarm" pair.
6. The two models are very close in the majority of the cases, but are still different (two different models of the same process), and consequently model the reverse pilot differently.
7. When one system produces an alarm, while no corresponding alarm is produced by the other system, we refer to the hypothetical alarm pair as being infinitely far. The alarms that do not have a pair are referred to as "orphan alarms".

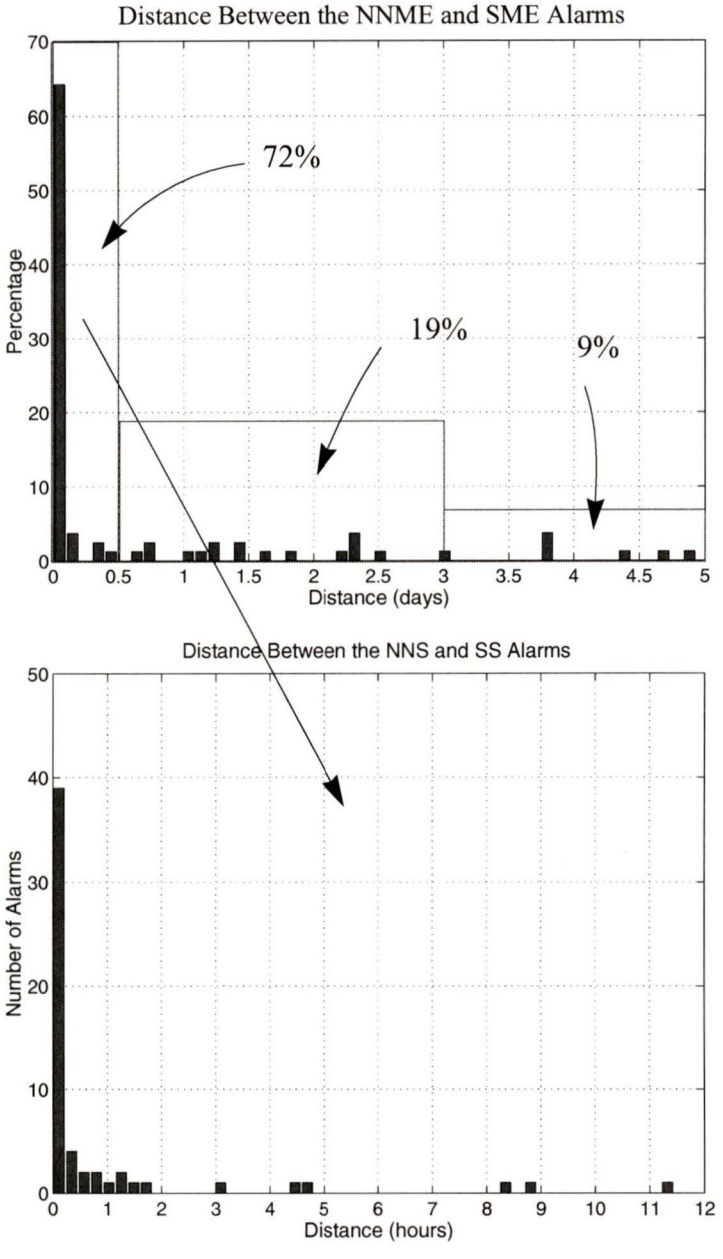

FIGURE 6-5.
(a) The distance of the alarms in days and their percentage of occurrence (dark columns). The percentage of the alarms with distance in the intervals 0-1/2 days, 1/2 -3 days and 3 - 5 days is also shown. For the alarms that are less than half a day apart, plot (b) shows their distance in hours versus their number.

It should be noted, however, that 18 "orphan" alarms were also produced by the two systems. The presence of these alarms implies that in certain instances, due to the differ-

ence of the two models, one of the systems alarms, while the other does not. It is most likely, then, that further improvement of the modeling accuracy of the reverse pilot and refinement of the alarming heuristics will eliminate these alarms.

In conclusion, the result analysis of this section shows that in the majority of the cases the two model-based systems alarm at the same, or very close, time. This fact indicates that the alarms refer to the same behavioural change events, and corroborates that indeed the two systems are discovering real changes in behaviour, and therefore are suitable to be applied as fault detection systems.

6.3 Comparison of Limit Checking and Model-Based Fault Detection Systems

In this section, a comparison of the model-based fault detection systems and the traditional limit checking technique is presented. The comparison will be done on the basis of the width of the bounds used by each system, and the number of alarms produced. The width of the bounds determines the sensitivity of the fault detection system, while the number of alarms is directly linked with its false alarm rate. The comparison is done over the whole set of the 50 amplifiers.

6.3.1 Bound Widths and Number of Alarms

Since there is no direct information on the actual limit checking thresholds used for the 3 month data sets, the bound widths were estimated by analyzing the fault detection flags generated by the limit checking system. Through this analysis it was possible to estimate the positions of the lower and upper thresholds: the lower threshold was estimated to be the mid-level between the lowest non alarming level and the next lower, while the upper threshold was estimated to be the mid-level between the highest non alarming level and the next highest[8]. Although the real threshold might have been different from the ones determined by this method, this process generates a lower bound estimate of the threshold widths for each amplifier.

---

8. A detailed description of the threshold estimation method used can be found in [31].

Unlike limit checking, the developed model-based fault detection system does not utilize a constant threshold bound width for the whole duration of the 3 months. Instead, as noted in Section 5.2.9.3, the thresholds are determined as a function of the number of the active levels that are present in a data set of 5,000 points (Eq. 5.7).[9] Consequently, the bound width for each amplifier was calculated as the average width for that period. The bound widths for both fault detection systems and for all 50 amplifiers are shown in Table 6.3.

Table 6.3: For each one of the 50 amplifiers: The bound width used by Demon and by the Model-Based system, and the produced number of alarms.

No.	Amplifier	Demon		NNME and SME		
		Bound Width	Alarms	Bound Width	Alarms NNME	Alarms SME
1	**102**	3	0	2.51	0	0
2	**110**	2	3643	2.53	0	0
3	**117**	5	742	2.57	2	2
4	**124**	8	5	2.59	0	0
5	**138**	8	2	2.63	0	0
6	**141**	8	0	2.65	0	0
7	**203**	5	6488	2.65	3	2
8	**206**	4	20229	2.68	2	3
9	**207**	3	22804	2.72	0	0
10	**223**	7	7	2.74	3	4
11	**232**	7	11469	2.76	0	1
12	**236**	8	1	2.76	0	1
13	**240**	7	60483	2.76	0	0
14	**241**	7	11	2.76	3	2
15	**246**	9	37574	2.78	0	0
16	**273**	9	5	2.78	1	3
17	**278**	9	14	2.82	1	1
18	**283**	9	13	2.82	3	4
19	**292**	9	3083	2.86	0	0
20	**296**	9	12541	2.88	6	7
21	**300**	9	19357	2.90	4	5
22	**313**	4	16299	2.93	0	0
23	**321**	9	1	2.97	0	0

9. The threshold calculated from Eq. 5.7 is utilized as an absolute error threshold between the reverse pilot signal and the model's output. Hence, this threshold must be doubled to represent the bound width used.

Table 6.3: For each one of the 50 amplifiers: The bound width used by Demon and by the Model-Based system, and the produced number of alarms.

No.	Amplifier	Demon		NNME and SME		
		Bound Width	Alarms	Bound Width	Alarms NNME	Alarms SME
24	**353**	6	0	2.97	0	1
25	**354**	6	1	2.99	0	1
26	**361**	5	2	3.03	0	0
27	**393**	9	1197	3.05	0	0
28	**394**	9	11428	3.03	0	0
29	**427**	6	8	3.01	5	4
30	**433**	6	96	2.99	4	4
31	**443**	8	443	2.97	6	6
32	**446**	8	741	2.93	8	9
33	**447**	8	725	2.93	9	10
34	**511**	4	6286	2.92	0	0
35	**514**	4	4	2.88	0	0
36	**518**	6	12	2.84	0	0
37	**519**	5	2	2.82	0	0
38	**531**	5	4	2.80	0	0
39	**631**	7	1	2.76	1	4
40	**632**	7	4	2.76	0	1
41	**633**	7	1	2.78	4	4
42	**641**	5	238	2.80	4	4
43	**642**	7	0	2.80	4	4
44	**671**	8	550	2.82	8	7
45	**711**	2	4712	2.82	0	0
46	**730**	5	7540	2.84	0	0
47	**801**	3	6766	2.86	0	0
48	**810**	7	321	2.88	1	1
49	**821**	5	35	2.88	0	0
50	**894**	5	13	2.90	0	0
Average:		6.42	5,118	2.94	1.64	1.88

Figure 6-6 illustrates the lower estimate bound widths for the limit checking system, and the average widths for each amplifier for the model-based system. The average standard deviation of the bound widths for the latter case is 0.538 dBmV.

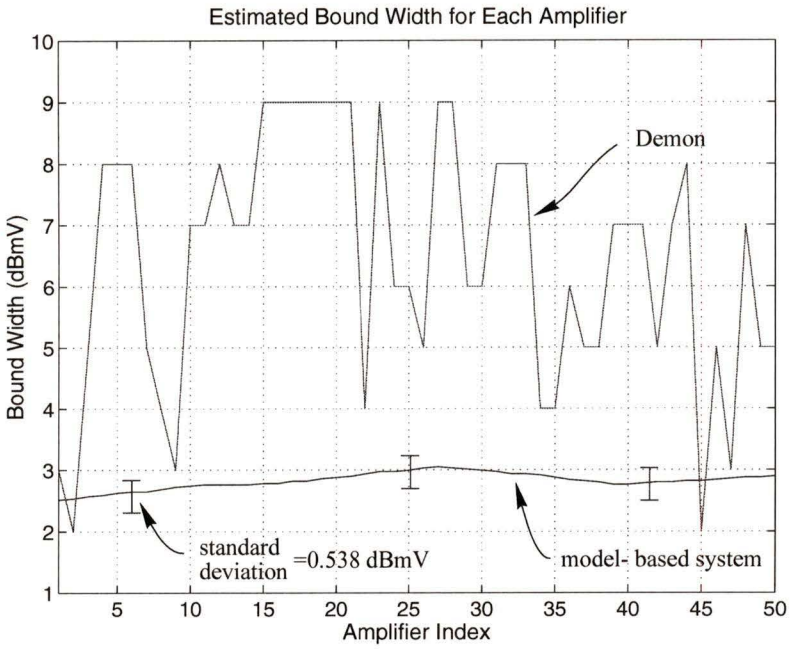

FIGURE 6-6.
The bound width for both the limit checking fault detection (Demon), and for the model-based fault detection. The bound width is plotted for all 50 amplifiers. (the X- axis values correspond to the indices of Table 6.3)

In Figure 6-7 a histogram of the same data plotted in Figure 6-6 is presented, with the light columns representing the percentage of occurrence of various bound widths utilized by the limit checking system, and the dark columns representing the percentage of occurance of all the bound widths used by the model-based systems.

It can clearly be seen from the results of Table 6.3 (illustrated in Figure 6-6 and Figure 6-7), that the model-based system utilizes tighter bounds than the limit checking system. On the average, these bound widths are approximately 2 times tighter than the average bound widths employed by the limit checking system.

Clearly, simply by comparing the bound widths used by the two systems it is not possible to determine whether one system is comparably better than the alternate. It is also necessary to compare the number of alarms produced by the two systems. For each amplifier, Table 6.3 shows the number of alarms produced by each system over the 3 month period. The last row of the table contains their average values. It can be seen that, both the

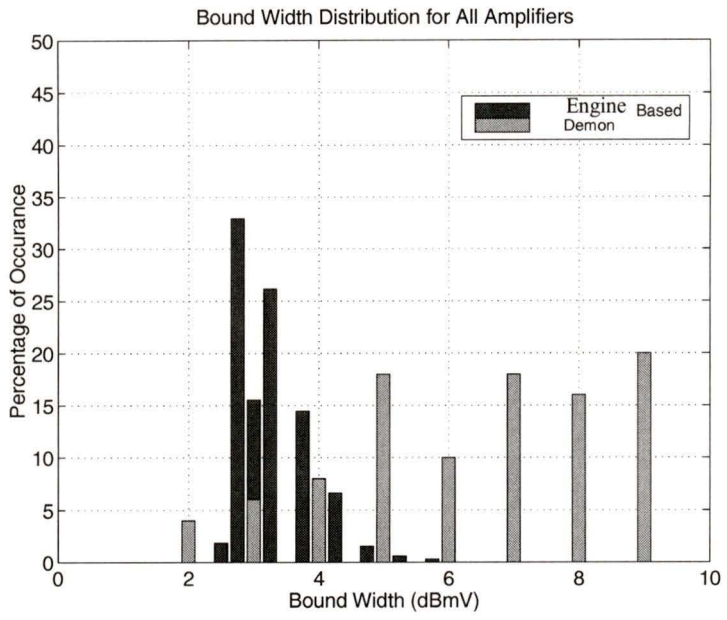

FIGURE 6-7.
(a) The percentage of occurrence of the bound widths used by the limit checking system (light columns). The dark columns represent the percentage of occurrence of all the widths (not the average) used by the model-based systems.

model-based systems (NNME and SME) produce many fewer alarms than the limit checking system. On the average, limit checking produces a number of alarms that is more than three decimal orders larger than the number of alarms produced by either the NNME or SME based systems. Figure 6-8 illustrates the number of alarms for all 50 amplifiers, for the Demon and NNME based fault detection systems.

6.3.2 Discussion

It is important to remember that the threshold utilized by the model-based system is a threshold on the absolute difference between the model and the reverse pilot signal. Its threshold bounds are not located at fixed levels in their time domain representation, but rather follow the model's fluctuations with the temperature. This allows the system to increase its sensitivity by utilizing tight bounds, while avoiding the generation of false alarms caused by the temperature dependency of the signal. In contrast, the limit checking

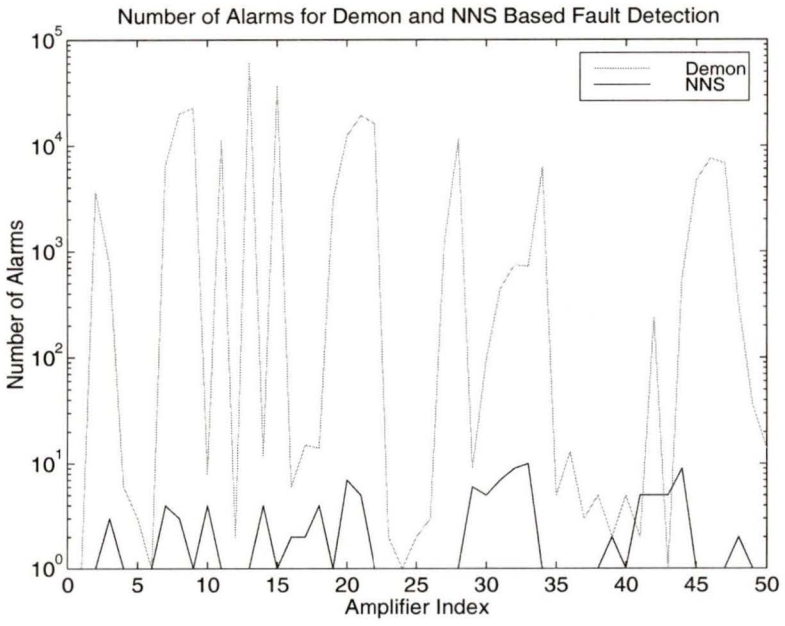

FIGURE 6-8.
Number of alarms for Demon and NNME based fault detection systems.

bounds are fixed in the time domain, and therefore can not follow the temperature dependency of the signal. This lowers the sensitivity of the system to behavioural changes and results in an increased false alarm rate.

As noted earlier, the limit checking bounds used for the comparison represent a lower estimate of the amplifier threshold widths. Most likely, the utilized bound widths were larger than those estimated. Even as such, for the amplifier set used in this experiment and for the 3 month period, generally, the developed model-based fault detection system utilized tighter bounds (is more sensitive) than the limit checking system. In addition, the model-based system produced significantly less alarms than the limit checking system.

However, there were two cases (amplifiers 110 and 711) where the limit checking bound widths were tighter than the average bound width utilized by the model-based system. Observe, however, that limit checking (Demon) produced 3,643 alarms for amplifier 110 and 4,712 alarms for amplifier 711, while the model-based system did not produce any alarms for these two amplifiers. It should be noted that these two amplifiers were modeled very well by both modeling engines (see modeling errors in Table 6.1), indicat-

ing that there were no changes in behaviour and therefore no need for alarming. Figure 6-9 shows the reverse signal of amplifier 711, the NNME model and the Demon alarms and bounds.

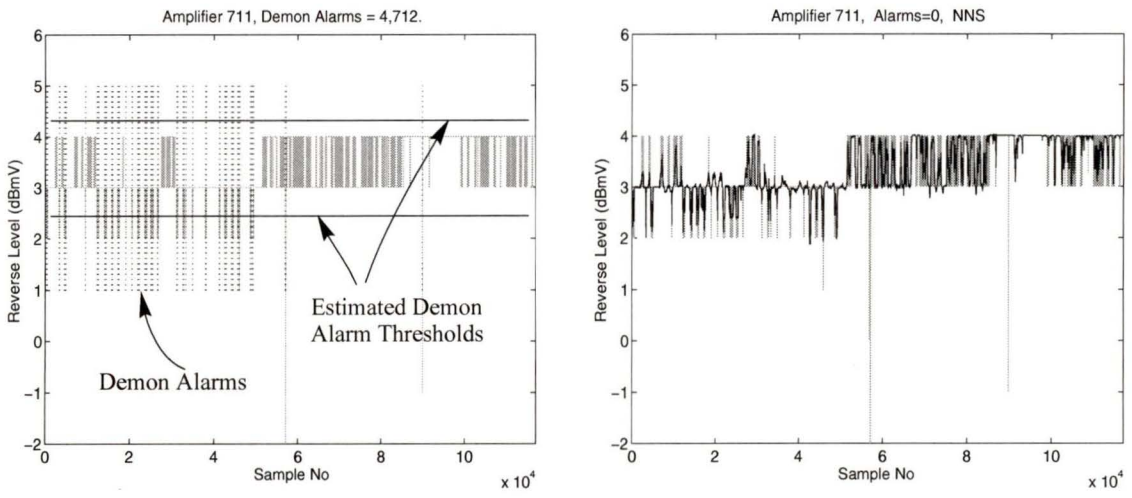

FIGURE 6-9.
a) The reverse pilot of amplifier 711 (gray trace) and the location of the Demon bounds and alarms. b) The reverse pilot (gray trace) and the NN modeling (dark trace).

In all cases but two, those of amplifiers 633 and 642, the number of alarms produced by the limit checking system is higher than that produced by the model-based systems. For these two amplifiers, as can be seen in Figure 6-8 (values can be found in Table 6.3), four alarms were produced by the model-based systems using an average bound width of 2.8 dBmV, while no alarms were produced by the limit checking system using a bound width of 7 dBmV. This is a typical example of the lack of sensitivity of the limit checking system. In both cases, limit checking uses very wide bounds (during that period at least), allowing real changes in behaviour to go undetected. The modeling of amplifier 642 using the Statistical Modeling Engine can be seen in Figure 6-10. The SME based system discovered 4 major changes in behaviour. Although the model was re-trained an equal number of times, the final rms error of the modeling remained relatively high (*rms=0.6*), an additional indication that during that period, the behaviour of the reverse pilot was indeed inconsistent.

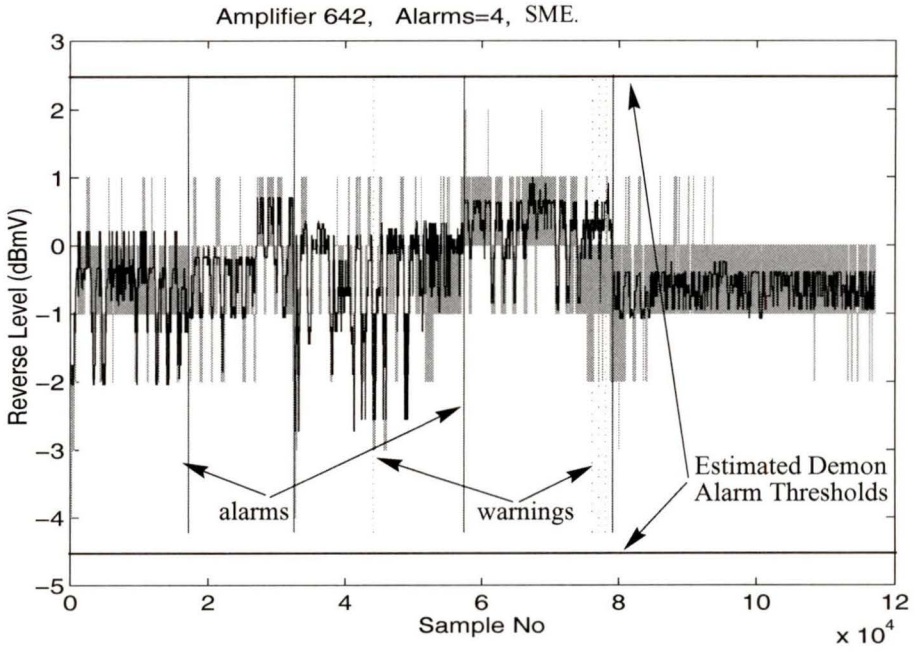

FIGURE 6-10.
The reverse pilot of amplifier 642 (grey trace) and the Statistical modeling (dark trace). The location of the 4 alarms are denoted by solid vertical lines. The horizontal lines indicate the position of the estimated Demon alarm thresholds, which are 7 dBmV apart.

In summary, it can be said that the results from the large scale experiment indicate that, the model-based fault detection system generally performs better than the limit checking system. The model-based system was capable of identifying and temporally locating changes in behaviour which generally were unnoticed by the limit checking system. It utilizes tighter bound widths, and in addition it produces a significantly lower number of alarms. As such, a model-based system may be considered more suitable for fault detection of the reverse pilot of cable television amplifiers.

6.4 Average Temperature

Ambient temperature plays a major role in the behaviour of the network. Changes in the temperature cause the load of the outgoing cable to change, which in turn is responsible for the variations of the reverse pilot. The reverse path, depending on the location of the polled amplifier, may extend for several kilometers, and the reverse signal must traverse many intervening cable segments and amplifiers before it reaches the head end

where it is measured. On the other hand, the temperature measurement that is obtained is that of the enclosure of the polled amplifier, which does not accurately reflect the temperature variations along the total reverse path (see Figure 6-11). As such, using the enclosure temperature of an amplifier to model its reverse pilot may not return the best model.

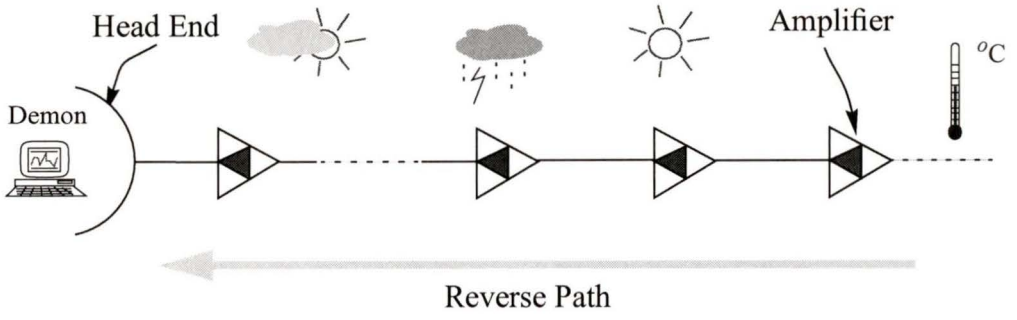

FIGURE 6-11.
The temperature of the enclosure of an amplifier, generally, does not accurately reflect the temperature variations along the reverse path towards the head end.

In order to obtain a more consistent estimation of the temperature variations over the length of the reverse path, we can use as temperature, the average of the amplifier enclosure temperatures of the amplifiers in the leg of the trunk starting with the one that is being polled and extending to the head-end.

Work done in [24], as well as the preliminary experiments presented in this section show that, indeed, the use of the average temperature for the modeling of the reverse pilot offers some improvement in the modeling error. For these experiments, the reverse pilot of amplifier 331 and 273 were modeled, using both the enclosure temperature and the average temperature. For practical reasons, the average temperature used was that deriving from the averaging of the enclosure temperatures of only three subsequent amplifiers: i.e. amplifiers 331, 136 and 135 for the modeling of amplifier 331, and 273, 271 and 124 for the modeling of 273.

In Figure 6-12(a) we can see a section of the temperatures of the three amplifiers 331, 136 and 135, while in Figure 6-12(b) the average temperature obtained and the enclosure temperature of amplifier 331 are plotted.

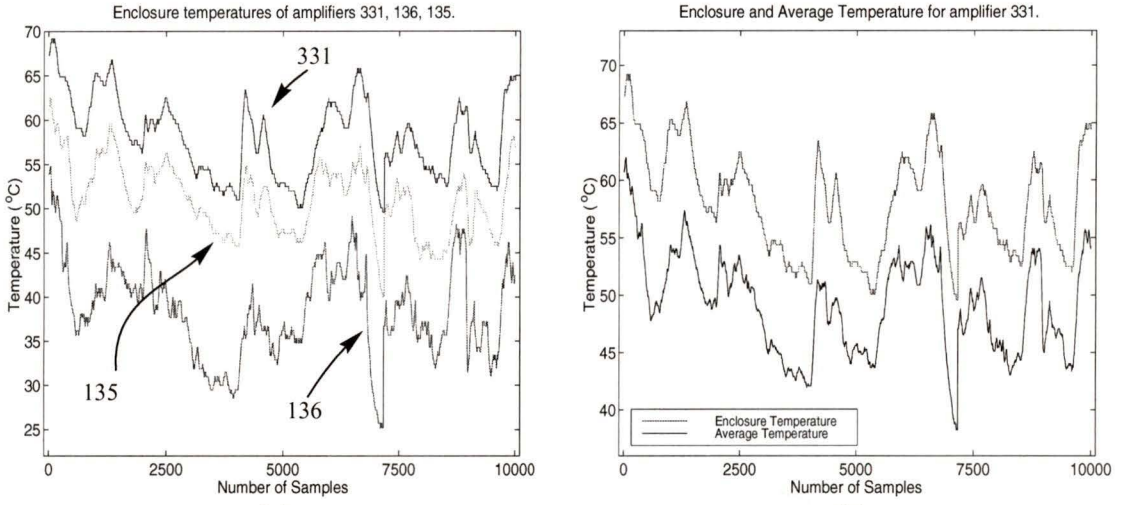

FIGURE 6-12.
(a) A section of the enclosure temperatures of the three amplifiers 331, 136, 135, (b) the enclosure temperature of 331 and the resulting average temperature.

The temperature sections shown in both subplots correspond to the first 10,000 temperature points used for the modeling of the reverse pilot of amplifier 331 (Figure 6-13). The modelings using both the enclosure temperature and the average temperature are shown in Figure 6-13(a) and Figure 6-13(b) respectively.

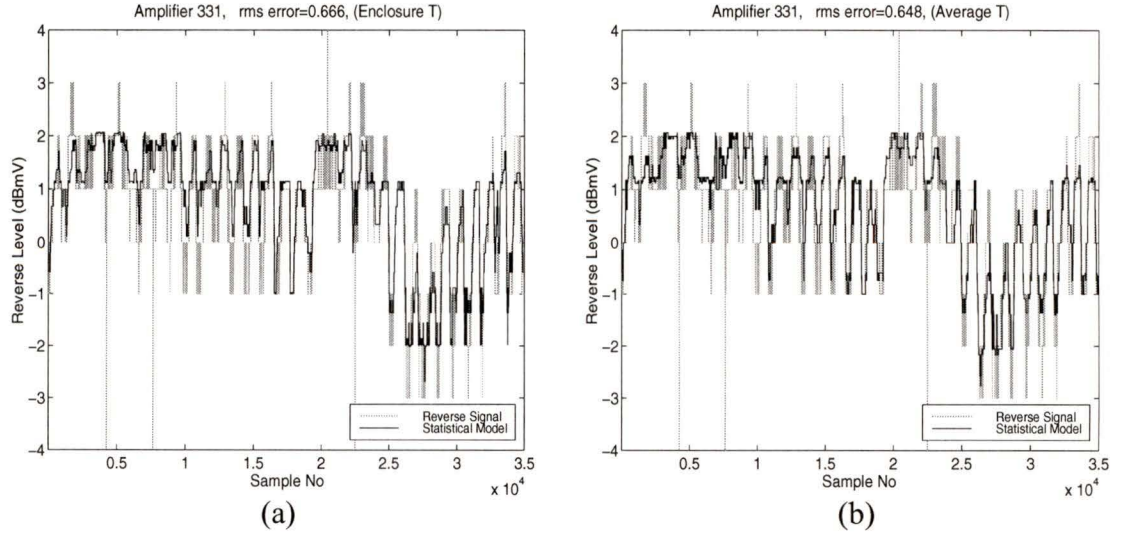

FIGURE 6-13.
The modeling of the reverse pilot for amplifier 331 using the enclosure temperature (a), and the average temperature (b)-notice the improvement in the modeling error.

The modeling rms error for the case where the average temperature was used is 0.64, and it is slightly lower than the error of the modeling using the enclosure temperature (0.66). A similar improvement in the modeling was achieved by using the average temperature in the modeling of amplifier 273. The modelings using the enclosure and average temperatures are shown in Figure 6-14(a) and Figure 6-14(b) respectively.

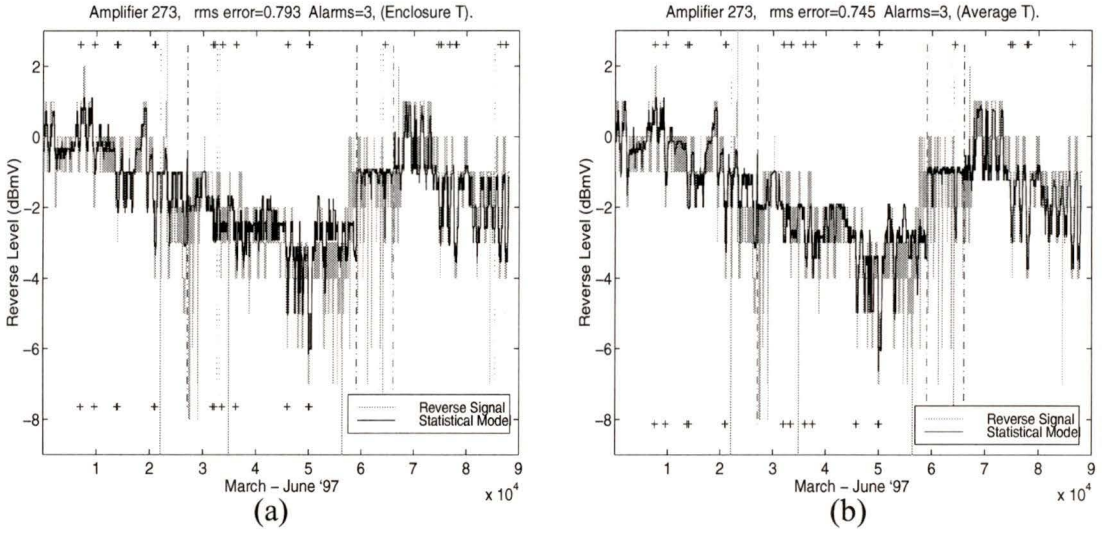

FIGURE 6-14.
Modeling of amplifier 273 using the enclosure (a) and average (b) temperatures. Using the two different models, the fault detection system alarms at the same locations.

In this case, the modeling using the average temperature (rms error = 0.74) is more accurate than the modeling using the enclosure temperature (rms error = 0.79). Observe also that in both cases the fault detection system generates the same number of alarms at the same locations, indicating changes in the behaviour of the reverse pilot.

The results of this preliminary experiment show that there can be a potential improvement in the modeling of the reverse pilot, by using the average temperature instead of the enclosure temperature. This average temperature reflects more accurately the temperature along the cable span between the modeled amplifier and the head-end. It should be noted however, that particularly for amplifiers located far from the head end, the calculation of the average temperature is not very easy, since information on the complete topology of

the amplifier network is needed, which generally is unknown[10]. Further large scale experiments need to be done to quantify the modeling improvement that results in by the use of the average temperature.

6.5 Real-Time Applicability

Obviously, for a fault detection technique to be useful in an operational fault detection system it is necessary for this technique to be applicable in real-time. The modeling engine-based fault detection methods presented in this thesis, were developed in terms of a batch processing context. All the experiments performed to set the parameters of the system and to test their final performance used historical data acquired from a operational cable plant. However, there are no processes used that cannot be applied in real-time and therefore there are no limitations in the real-time implementation of these techniques.

The training of the modeling engines can be done sample by sample in the case where a neural network is used, or after enough samples have been gathered in the case of the statistical modeling engine. The modeling of the reverse pilot and the behavioural change check also can be done on a per sample basis. As such, both developed techniques can be applied in real-time with minor modifications. These modifications are discussed in Appendix A, while their implementation is left as an area of future work.

6.6 Summary

Both modeling engines presented in Chapter 5, the NNME and the SME, are capable of modeling in a very similar way and generally sufficiently closely the reverse pilot signal of a well behaving amplifier. By using either of these two engines in a model-based fault detection system, it was possible to detect changes in the behaviour of the reverse pilot. In the majority of the cases, the two systems marked the same changes in behaviour and temporally close enough to verify the accuracy of the fault detection system. The developed model-based system generally utilizes tighter alarming bounds (on the average, two times tighter) than the limit checking fault detection system currently used. In addition, the model based approach produced many fewer alarms (in the order of 10^3).

---

10. Due to maintenances, repairs/replacement or amplifiers, and reconfigurations/expansion of the networks, the detailed topology is not always known.

The reliability of the novel model-based fault detection technique presented, depends heavily on the accurate modeling of the reverse pilot. Preliminary experiments show that, the modeling of the reverse pilot-temperature dependency of a given amplifier can be improved if instead of its enclosure temperature, an alternative measure of the temperature is used. This alternative temperature is the average of the enclosure temperatures of the amplifiers between the modeled amplifier and the head-end.

Chapter 7:

Conclusions and Future Work

7.1 Summary-Conclusions

Throughout this thesis a novel model-based fault detection technique suitable for signals with dependencies was presented. The implementation of this technique is based on the use of a modeling engine to model the monitored signal, and use the difference between the model and the actual signal to detect any changes in the signal's behaviour. These behavioural changes are believed to be associated with the onset of faults. The developed technique is general, requires minimum of *a priori* knowledge on the number and type of fault modalities, and can be retro-fitted into existing status monitoring systems.

The model-based system was calibrated specifically to be able to identify changes in the behaviour of the reverse pilot signal of cable television networks. For the modeling part, two modeling engines were developed which were able to model the dependency of the reverse pilot on the temperature. The first engine was based on the use of a feedforward neural network (NNME) while the second was based on the application of appropriate statistical analysis on both the reverse pilot and temperature signals (SME).

A large scale experiment was performed to test the modeling capabilities and the fault detection properties of the two systems: using data collected over a 3 month period from 50 randomly selected amplifiers from a operational cable plant, it was verified that the model-based fault detection system, using either modeling engine was able to discover the behavioural changes of the reverse pilot signal. The temporal nearness of the alarms produced by the two systems verified the accuracy of their fault detection capabilities.

The two novel systems were also compared directly to the limit checking fault detection system currently used in the cable plant where the test data were taken from. The comparison was based on the he number of alarms produced by each system, as well as their fault detection sensitivity. This comparison revealed that the fault detection properties of the model-based system are superior to those of the traditional limit checking

approach: the model-based system produced a considerably lower number of alarms while exhibiting a higher fault detection sensitivity. This higher sensitivity enable it to detect behavioural changes which the limit checking system was incapable to discover. The outcome of the comparison indicated that a model-based system is more suitable to be used for the fault detection of the reverse pilot of cable television amplifier networks.

In conclusion:

- Feedforward neural networks can learn the static non-linear dependency of the reverse pilot to the temperature. By training the network on a relatively small data set of two days, the NN can model accurately the reverse signal for an extended period of several months.
- Initial weight space explorations help the neural network trainings avoid high error saddle-points and local minima, and improve significantly the final training error, particularly for networks with low number of hidden neurons.
- Alternatively to using a neural network, statistical analysis techniques can be applied to extract the reverse pilot-temperature relationship from a small training set. By using this information, a modeling engine can model the reverse pilot closely.
- Using the difference between the reverse pilot signal and the model, it is possible to detect behavioural changes of the reverse pilot. Appropriate heuristics have been developed so that the a fault detection system issues alarms when severe changes in behaviour are detected, or alternatively, warnings when milder anomalies in the behaviour are identified.
- The developed fault detection systems were shown to have superior fault detection properties to those of the traditional limit checking approach.

It is important to note that the modeling and fault detection techniques which were developed in this work are general approaches and can be applied to a wide range of large scale engineering plants. As one of the primary goals of the work was to developed systems which can be implemented in operational engineering plants, special attention was given in their design and software implantation so that they can be applicable in real-time with minor modifications.

7.2 Future Work

Some of the areas of future work include:

- The implementation of the modifications for the real-time application of the modeling engines and fault detection system.[1]
- Techniques for the real-time estimation of the underlying reverse pilot signal (e.g. wavelet de-noising) to enable better modeling.
- Development of heuristics for the construction-selection of optimal training sets.[1]
- Optimization of the bound widths used by the fault detection system.
- Theoretical framework for determining the alarming and retraining methodologies.
- Large scale experiments for the calibration of the fault detection system, ideally in an environment in which information identifying the occurrence of true faults is available.
- Integration of the correlation detector into the fault detection system.
- The determination of the applicability of the modeling methods for other monitored signals of the cable plant (e.g. forward pilot signal, current signal).

---

1. Some suggestions for the real-time applicability of the modeling engines and the fault detection system are presented in Appendix A.

Bibliography

[1] Bernauer E., Demmou H., " Temporal Sequence Learning with Neural Networks for Process Fault Detection" *IEEE International Conferance on Systems, Man and Cybernetics*, 1993, Vol. 2, pp. 375-380.

[2] Bernieri A., Betta G., Pietrosanto A. and Sansone C."A Neural Network Approach to Instrument Fault Detection and Isolation", *IEEE Transactions on Instrumentation and Measurement*, Vol. 44, No. 3, pp. 747-750, June, 1995.

[3] Bernieri A., Betta G. and Liguori C. " On-Line Fault Detection and Diagnosis Obtained by Implementing Neural Algorithms on Digital Signal Processor", *IEEE Transactions on Instrumentation and Measurement*, Vol. 45, No. pp. 5, pp. 894-899, October, 1996.

[4] Caudill M. "Neural Network Primer Part I", *AI Expert*, Feb 1989.

[5] C-COR Electronics Inc., *Instruction Manual for Status Monitoring Systems*, Rev.0, May 1984.

[6] Chow Mo-Yuen, Sharpe R.N, Hung J.C "On the Application and Design of Artificial Neural Networks for Motor Fault Detection- Part I & Part II" *IEEE Transactions on Industrial Electronics,* Vol. 40, No2, pp. 181- 196, April 1993.

[7] Demuth H., Beale M. "Neural Network Toolbox-User's Guide" The MathWorks Inc., Jan. 1994.

[8] Dimopoulos N. J. "A Study of the Asymptotic Behaviour of Neural Networks" I*EEE Transactions on Circuits and Systems,* Vol. 36, No5, pp. 687-694, May 1989.

[9] Dimopoulos N. J. "Neural Networks-From Biological Origins to Silicon", *University of Victoria*, 1997.

[10] Doraiswami R. "Performance Monitoring in Expert Control Systems", *Automatica* Vol. 25, No6, pp. 799-811, 1989.

[11] Doraiswami R. "Performance Monitoring and Fault Detection Using a Linear Predictive Coding Algorithm", *Automatica,* Vol. 29, No4, pp. 1115-1120, 1993.

[12] Doraiswami R., Stevenson M., Didrich C., "Autonomous Control Systems: Monitoring Diagnosis and Tuning", *1993 IEEE International Conference on Systems, Man and Cybernetics,* Vol. 3, pp. 61.

[13] Dorocic J., "Asymptotically Stable Recurrent Neural Networks: Theory and Application" *M.A.Sc. Thesis, Dept. of Electrical And Computer Engineering, University of Victoria*, 1997.

[14] Frank P.M. "Fault Diagnosis in Dynamical Systems Using Analytical and Knowledge-based Redundancy. A Survey and Some New Results" *Automatica,* Vol. 26, No. 3, pp. 459-471, 1990.

[15] Frank P.M. "On Line Fault Detection in Uncertain Nonlinear Systems Using Diagnostic Observers, A Survey" *Intelligent J. System SCI*, 1994, Vol. 25, No. 12, pp. 2129-2154.

[16] Gent C.R. and Sheppard C.P. "Predicting time series by a fully connected neural network trained by backpropagation" *Computing & Control Engineering Journal*, May 1992.

[17] Gertler Janos, "Model-Based Failure Detection and Isolation in Complex Plants", *IEEE Control Systems Magazine*, Dec. 1988,

[18] Haykin Simon, "Neural Networks. A Comprehensive Foundation", *Macmillian Publishing Company*, 1994.

[19] Hornik K., Stinchcombe M., and White H., "Multilayer Feedforward Networks are Universal Approximators," Neural Networks, Vol. 2, 1989.

[20] Ingman D., Merlis Y., "Local minima escape using thermodynamical properties of neural networks," *Neural Networks*, Vol. 4, pp. 395-404, 1991.

[21] Isermann R., "Process Fault Detection Based on Modeling and Estimation Methods-A Survey", *Automatica,* Vol. 20, No. 4, pp. 387-404 1984.

[22] Isermann R., "Fault Detection of Machines via Parameter Estimation and Knowledge Processing. Tutorial Paper", *Automatica,* Vol. 29, No. 4, pp. 815-835 1993.

[23] Karunanithi N., Whitley P., "Using Neural Networks in Reliability Prediction", *IEEE Software*, July 1992.

[24] Kourounakis N.P., Neville S.W., Dimopoulos N.J., "Early Fault Detection in Cable Television Networks (The Case of The Reverse Pilot)", *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, Aug. 20, 1997, Vol. 2, pp. 551.

[25] O'Sullivan P.J., "Application of a New Technique for Modeling System Behavior" *ISA Symposium,* Edmonton, Alberta, May 2, 1991.

[26] Leigh J. R., "Neural Networks in Process Control", *Computing & Control Engineering Journal*, May 1992.

[27] Lipman R., P.,"An Introduction to Computing with Neural Networks", *IEEE ASSP Magazine,* Vol. 4, No.2, pp. 4-22, April 1987.

[28] Mirzai A.R. and Leigh J.R. "An Overview of the Applications of Neural Networks in Process Engineering", *Computing & Control Engineering Journal*, May 1992.

[29] Narendra K., S., Parthasarathy K., "Identification and Control of Dynamic Systems Using Neural Networks", *IEEE Transactions Neural Networks,* Vol. 1, No. 1, pp. 4-27, March 1990.

[30] Neville S. W. "A Prototype Expert System Based Diagnostic Tool for Cable Trunk Amplifier Networks", *M.A.Sc. Thesis, University of Victoria, 1992.*

[31] Neville S. W. "Early Fault Detection In Large Scale Engineering Plants", *Ph.D. Dissertation, University of Victoria, 1998.*

[32] Parten C., Seaks R. "Fault Diagnosis and Neural Networks" *IEEE International Conference on Systems, Man and Cybernetics*, 1991, Vol. 3, pp. 1517-1521.

[33] Patton R., Frank P., Clark R."Fault Diagnosis in Dynamic Systems, Theory and Applications", *Prentice-Hall N.Y.* 1989.

[34] Purucker M.C. "Neural Network Quarterbacking" *IEEE Potentials,* Aug 1996.

[35] Simon W. and Carter J. "Backpropagation Learning Equations From the Minimization of Recursive Error" *Proceedings of the IEEE International Conference on Systems Engineering*, pp. 155-160, 1989.

[36] Simpson P.K. "Foundations of Neural Networks" *Artificial Neural Networks Paradigms, Applications and Hardware implementations, Sanchez-Sinecio, IEEE PRESS* 1992.

[37] Simpson P.K. "Artificial neural systems: Foundations, Paradigms and Applications and Implementations" *New York, Pergamon Press,* 1990.

[38] Smith A. E., Dagli C. H. "An Empirical Analysis of Backpropagation Error Surface Initiation for Injection Modeling Process Control" *IEEE International Conference on Systems, Man and Cybernetics*, 1991, Vol. 3, pp. 1529-1534.

[39] Sorsa T., Koivo H., "Application of Artificial Neural Networks in Process Fault Diagnosis", *Automatica*, Vol. 29, pp. 843-849, 1993.

[40] Vemuri R. V., "Artificial Neural Networks: Theoretical Concepts", *IEEE Computer Society Press*, 1988.

[41] Vemuri R. V., "Artificial Neural Networks: Concepts and Control Applications" *IEEE Computer Society Press Tutorial*, 1992.

[42] Wasserman P, "Neural Network Computing: Theory and Practice" *New York: Van*

Nostrand Reinhold, 1989.

[43] Werbos P.J. "The Roots of Back Propagation", *John Wiley & Sons, INC., New York*, 1994.

[44] Widrow B., Lehr M., "30 Years of Adaptive Neural Networks: Perceptron, Madaline, and Backpropagation", *Proceedings of the IEEE*, Vol. 78, No. 9, pp. 1415-1442, Sept. 1990.

[45] Willis M.J, Montague G.A and Morris A.J. "Modelling of Industrial Processes Using Artificial Neural Networks", *Computing & Control Engineering Journa*l, May 1992.

[46] Wu Jian-Kang, "Neural Networks and Simulation Methods", *Marcel-Dekker Inc.*, 1994.

[47] Yamada R., Kami H., Tsukcumo J. and Temma T. "Handwritten numeral recognition by multy layered network with improved learning algorithm" *Proceedings of the International Joint Conference on Neural Networks*, Vol. II, pp. 256 - 266, 1989.

Appendix A:

Suggestions on the Real-Time Application and Improvement of the Developed Fault Detection System

A.1 Real-Time Applicability

Both model-based fault detection systems (NNME and SME) can be applied in real time with minor modifications. The modifications required for the real time implementation can be classified in two categories: modifications on the modeling process, and modifications in the fault detection process.

The statistical modeling can be applied in real time as it is without any modifications. On the other hand, the neural network modeling process has to be modified in such a way so that no information from the "future" is needed. One area where modifications need to be made is in the normalization process. Currently the normalization of the reverse pilot is done by normalizing *99%* of the signal in the space *[0.1 0.9]*, while the rest *0.1%* in the spaces *[0 0.1]* and *[0.9 1]*. This is done by calculating the number of active levels that are present in the signal. In real time this is not possible. One approach to overcome this difficulty is to normalize the reverse signal in a consistent manner[1], just like in the case of the temperature normalization (See "Normalization Function" on page 64.)

Alternatively, the normalization could all together be avoided, if another way could be found to produce the benefits of the normalization. The benefits of normalizing the data are twofold. First, through the non-uniform normalization, the influence of outliers in the training is diminished[2], and secondly there is a reduction in the training duration[3]. The

---

1. For example, the normalization could be done in such a manner so that the part of the signal that resides between *[-6, 6] dBmV* would be normalized in *[0.1 0.9]*, and the remaining in the intervals *[0, 0.1]* and *[0.9, 1]*.
2. The outliers belong to the *1%* of the signal which is normalized in the interval *[0 0.1]* or *[0.9 1]* which is *12.5%* of the *[0.1 0.9]* interval where the rest of the signal is normalized.
3. The use of un-normalized data will require extra training time for the internal adjustment of the weights to perform the mapping of the un-normalized input temperatures to the *[0, 1]* space of the sigmoid activation function, and back again to the *[-10, 10]* space of the reverse pilot.

loss of these benefits could possibly be compensated for by using other techniques to reduce the influence of the outliers to the training, and by allowing longer training durations.

Finally, as noted in the previous chapter, the training of the neural network can be considerably long. This however, is not a problem for the real time implementation, since for the specific plant this system was designed for, the data acquisition process is significantly slower than the modeling process.

For the fault detection part of the system, two are the areas where modifications need to be made. First, the alarm threshold setting heuristics need to be changed. Currently, the alarm thresholds used for a certain reverse pilot data section are set dynamically as a function of the number of active levels that are present in that section. For the real time application, appropriate heuristics need to be derived so that the alarm thresholds can be set by relying only on past information. An alternative solution is to use constant error thresholds (i.e. preset to a certain value of X dBmV). This would become possible with the improvement of the modeling accuracy.

Secondly, the temperature check and reverse alarm routines need to be modified so that they perform the "checkings" every sample instead of every thousand samples. In addition, proper heuristics need to be developed to avoid numerous re-trainings after each single new temperature is encountered. Instead, the modeling engine should retrain after a predetermined number of new temperatures has been collected.

A.1.1 Improvements

For the novel model-based fault detection systems presented in this thesis, the most significant improvement in their overall performance will come from the improvement in the modeling accuracy of the modeling engines used. In Chapter 6 (Section 6.4), a method (temperature averaging) which seems to improve the modeling accuracy of the reverse pilot was presented. In this chapter, in Section A.1.2 the potential modeling improvement that would result in by the selection or generation of appropriate training sets is discussed.

A.1.2 Training Set

The use of a "good" training set is critical for the accurate training of the modeling engines and hence for the modeling of the reverse pilot. One way of improving the "quality" of the training set is do derive heuristics that decide if the training set is adequately "good" to train on. If the training set is not good enough (e.g mixed correlation) then a new training set could be created by filtering out the bad features and inconsistencies or by entirely replacing the training set with a new and more appropriate one.

Another way that indirectly the modeling accuracy of the reverse pilot can be improved is by reducing the size of the training set. The use of reduced size training sets could make a difference in the modeling process using the neural network modeling engine. With smaller training sets longer training durations could be allowed resulting in lower modeling errors. By using statistical analysis techniques, it is possible to extract the redundant features of the trainings set and reduce its size; however, care must be taken to preserve the underlying behavioural statistics.

Also a potential improvement could come by using a de-noised estimate of the reverse pilot, obtained for example through wavelet estimation, to train the modeling engine. It is possible that a de-noised and hence continuous training set will result better modeling and alleviate the false alarms that are caused by the large quantization step of the reverse pilot.

Appendix B:

Correlation Detector

The correlation detector is implemented by the function CORRELATION, which calculates a sampled cross-correlation as the inner product of two normalized zero mean vectors:

function [corr,corr_indx]=CORRELATION(R,T);

%---

A segment of Reverse Pilot and Temperature data (win_len=300) is normalized in [-1,1], and farther, the mean value of that segment is subtracted. Then the temperature and Reverse Pilot segments are multiplied element by element, and the results are summed. This sum is the correlation (*corr*) for that segment of T and R. This process is repeated after sliding *win_len* by fifty (50) data pairs. Positive sums denote positive correlation and negative sums denoted negative correlation.

%---

```
n = min(length(R),length(T));                    % the minimum lenght
win_len=300;                                     % the window length
sld_win=50;                                      % sliding window step
Rm=[]; Tm=[];
Rcmp=[]; Tcmp=[]; corr=[];
for i=0 : sld_win:n-win_len-1,
    MaxR=max(R(i+1:i+win_len));                  % find maximum of R
    MinR=min(R(i+1:i+win_len));                  % find minimum of R
    x= -(MaxR+MinR)/2 ;
    N=(MaxR-MinR)/2;
    if N~=0,                                     %normilize between -1 : 1
            R_n =(R(i+1:i+win_len)+x)/N;
    else    R_n =1;
    end
    MaxT=max(T(i+1:i+win_len));
    MinT=min(T(i+1:i+win_len));
    x= -(MaxT+MinT)/2 ;
    N=(MaxT-MinT)/2;
    if N~=0
```

```
            T_n =(T(i+1:i+win_len)+x)/N;
    else    T_n =1;
    end
    Rm=mean(R_n);                                   % find the mean in the window
    Tm=mean(T_n);

    Rcmp=R_n-Rm;
    Tcmp=T_n-Tm;
    corr=[corr ;(sum(Rcmp.*(Tcmp)))/win_len]; % correlation vector
end
plot(sld_win*(0:length(corr)-1)+(win_len/2),corr,'y'),title('correlation'),xlabel(' X sld_win'),hold on,
```

Vita

Surname: Kourounakis Given Name: Nicolaos

Place of Birth: Montreal, Quebec, Canada.

Education Institutions Attended:

Aristotle university of Thessaloniki	1988 to 1995
Université de Lille	1993
University of Victoria	1995 to 1998

Degrees Awarded:

Ptyhion in Physics	University of Thessaloniki	1995

Honours and Awards

IKI-Erasmus Scholarship	1993

Publications:

Kourounakis N.P., S.W. Neville and N.J. Dimopoulos, "Early Fault Detection in Cable Television Networks (the Case of the Reverse Pilot)", *1997 IEEE, Pacific Rim Conference in Communication, Computers and Signal Processing (PACRIM '97)*, Conference Proceedings, Vol. 2 pp. 551.

Partial Copyright License

I hereby grant the right to lend my thesis to users of the University of Victoria Library, and to make single copies only for such users or in response to a request from the Library of any other University, or similar institution, on behalf or for one of its users. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by me or a member of the University designated by me. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Title of Thesis:

Improved Fault Detection in Cable Television Networks

Author: 

Nicolaos P. Kourounakis

March 11, 1998